MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21b/2016/01.05.13.10-TDI

UM FRAMEWORK PARA VALIDAÇÃO AUTOMÁTICA DE MODELOS APLICADO AO SUBSISTEMA DE ENERGIA DE UM PICOSSATÉLITE

Italo Pinto Rodrigues

Dissertação de Mestrado do Curso de Pós-Graduação em Engenharia e Tecnologia Espaciais/Gerenciamento de Sistemas Espaciais, orientada pela Dra. Ana Maria Ambrosio, aprovada em 21 de dezembro de 2015.

URL do documento original:
<http://urlib.net/8JMKD3MGP3W34P/3KT8D7H>

INPE
São José dos Campos
2015

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, TECNOLOGIA E INOVAÇÃO
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21b/2016/01.05.13.10-TDI

UM FRAMEWORK PARA VALIDAÇÃO AUTOMÁTICA DE MODELOS APLICADO AO SUBSISTEMA DE ENERGIA DE UM PICOSSATÉLITE

Italo Pinto Rodrigues

Dissertação de Mestrado do Curso de Pós-Graduação em Engenharia e Tecnologia Espaciais/Gerenciamento de Sistemas Espaciais, orientada pela Dra. Ana Maria Ambrosio, aprovada em 21 de dezembro de 2015.

URL do documento original:
<http://urlib.net/8JMKD3MGP3W34P/3KT8D7H>

INPE
São José dos Campos
2015

Dados Internacionais de Catalogação na Publicação (CIP)

Rodrigues, Italo Pinto.

R618f Um framework para validação automática de modelos aplicado ao subsistema de energia de um picossatélite / Italo Pinto Rodrigues. – São José dos Campos : INPE, 2015.

xxvi + 143 p. ; (sid.inpe.br/mtc-m21b/2016/01.05.13.10-TDI)

Dissertação (Mestrado em Engenharia e Tecnologia Espaciais/Gerenciamento de Sistemas Espaciais) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2015.

Orientadora : Dra. Ana Maria Ambrosio.

1. Execução automática de testes. 2. Pico e nanossatélites. 3. Validação de sistemas espaciais. 4. Verificação de sistemas espaciais. 5. Hardware-in-the-loop. I.Título.

CDU 629.7.01:629.78

Aluno (a): *Italo Pinto Rodrigues*
Título: "UM FRAMEWORK PARA VALIDAÇÃO AUTOMÁTICA DE MODELOS APLICADO SO SUBSISTEMA DE ENERGIA DE UM PICOSSATÉLITE **".**

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

Engenharia e Tecnologia Espaciais/Gerenciamento de Sistemas Espaciais

Dra. Maria de Fátima Mattiello-Francisco

Presidente / INPE / São José dos Campos - SP

Dra. Ana Maria Ambrosio

Orientador(a) / INPE / São José dos Campos - SP

Dr. Walter Abrahão dos Santos

Membro da Banca / INPE / São José dos Campos - SP

Dra. Eliane Martins

Convidado(a) / UNICAMP / Campinas - SP

Este trabalho foi aprovado por:

( ) maioria simples

(x) unanimidade

São José dos Campos, 21 de dezembro de 2015

"Onde existe vontade, sempre há solução".

Walter Bishop
em *"Fringe"*, 2012

A meus amados pais Marco e Elaine, e irmãos Tainah e Iago

AGRADECIMENTOS

Agradeço, primeiramente, a Deus, por tudo.

A minha família, pelo apoio e incentivo, e por entenderem minha ausência. Principalmente, a meus pais, que não mediram esforços em me ajudar nos momentos de dificuldade e por todo amor que dedicam a mim, e a minha irmã Tainah, e meu irmão Iago, pelo carinho e por sempre me receberem com um sorriso inspirador.

A minha orientadora, por quem eu tenho grande admiração, Dra. Ana Maria Ambrosio, pela sua disponibilidade na orientação deste trabalho e pela generosidade de compartilhar sua experiência, conhecimento e dicas preciosas para minha formação. Sobretudo, por ser tão cordial, acessível, motivadora e otimista.

Ao Christopher Shneider Cerqueira, pelo papel fundamental na realização deste trabalho, das ideias iniciais até a conclusão, me auxiliando nas questões práticas e resolução de problemas, com muita paciência, senso crítico e dedicação. Sobretudo, por ter se tornado um amigo, com que eu pude ter discussões filosóficas que muito me acrescentaram.

Ao Instituto Nacional de Pesquisas Espaciais (INPE) pela oportunidade de cursar o Mestrado. E à Capes pela bolsa de estudos.

Aos professores do curso ETE/CSE, com os quais eu tive a oportunidade de aprender e ter o primeiro contato com a área espacial.

À equipe responsável pelo satélite Tancredo 1, por compartilhar algumas informações do satélite, sobretudo o Sr. Auro Tikami, pelos esclarecimentos sobre o subsistema de energia elétrica e o Sr. Cândido Osvaldo de Moura, por ter idealizado este projeto inspirador.

Às secretárias da pós graduação, especialmente, à Edleusa e à Tuane, pela paciência e por estarem sempre dispostas a ajudar com as questões burocráticas.

Aos amigos que fiz no INPE, sobretudo, à Juliana Padilha Joria, com quem eu tive a alegria de dividir a sala 97, e ao Jaime Orduy Rodriguez, por ser uma pessoa admirável e um amigo que levarei para toda vida. À Roberta Porto, Joyce Lara, Paulo Barbosa, Alan de Brito, Camila Cossetin, Adriana Koumrouyan e Roy Soler, por terem feito os meus dias no INPE mais descontraídos.

As minhas amigas e companheiras de apartamento, Lorena Alves e Priscylla Oliveira, por serem amáveis e por terem se tornado uma extensão da minha família.

Por fim, a todos aqueles que contribuíram para a realização deste trabalho.

RESUMO

A Engenharia Espacial vem presenciando o avanço de satélites miniaturizados, principalmente os pico e nanossatélites, uma vez que, essas classes de satélites permitem o desenvolvimento rápido e com baixo custo de projeto e atende a uma gama de diferentes aplicações científicas e educacionais. A demanda crescente de nanossatélites impulsiona a necessidade de um processo de verificação bem definido, de modo a garantir que o projeto atenda às especificações. Estatísticas mostram que apenas 46% dos pico e nanossatélites lançados entre 2000 e 2015 tiveram sucesso. Um dos sistemas críticos, que está diretamente ligado ao sucesso de uma missão é o subsistema de energia elétrica. Devido a suas características construtivas, este subsistema apresenta falhas relacionadas à manutenção da energia, por causa do limite da área dos painéis solares e do tamanho da bateria. Na Engenharia Espacial, as atividades de verificação e validação são apoiadas pelas simulações realizadas nas diferentes fases do ciclo de vida que permitem antecipação de problemas. Os simuladores proporcionam aos sistemas em teste um ambiente sintético podendo ser compostos com modelos virtuais e/ou com modelos físicos. Dada a carência de métodos e procedimentos necessários para aumentar o número de missões de pico e nanossatélite bem sucedidas, esta dissertação propõe um *framework* para verificação de requisitos e validação de modelos, combinando simulações com execução automática de testes. O *framework* prevê três configurações para a execução de testes funcionais, uma configuração com modelos puramente virtuais, outra com modelos virtuais com modelo de interface física e uma configuração com *hardware-in-the-loop*. O framework cobre ainda: (i) definição de regras para criação de modelos proporcionando reutilização dos mesmos, (ii) combinação dos modelos com funções de teste, facilitando a repetibilidade dos testes, e (iii) execução de simulação e testes com captura automática dos resultados. Como estudo de caso utilizou-se o subsistema de energia elétrica do Tancredo 1, um picossatélite brasileiro em desenvolvimento com apoio do INPE. Os programas de computador desenvolvidos, os modelos do subsistema, os requisitos e casos de teste, bem como os resultados da simulação e dos testes são ilustrados com a aplicação do framework para um caso real.

Palavras-chave: Execução automática de Testes. Simulação. Validação de Sistemas Espaciais. Verificação de Sistemas Espaciais. Pico e nanossatélites. Subsistema de Energia Elétrica. Hardware-in-the-loop.

A FRAMEWORK FOR AUTOMATED MODEL VALIDATION APPLIED TO PICOSATELLITE ELECTRIC POWER SUBSYSTEM

ABSTRACT

The Space Engineering is witnessing the advance of miniaturized satellites, especially the pico and nanossatélites, since these satellites classes enable the fast development with low cost project, serving a range of different applications such as: scientific and educational. The growing demand for nanosatellites drives the need for a well-defined verification process in order to ensure that the project meets its requirements and specifications. Statistics show that only 46% of pico and nanosatellite launched between 2000 and 2015 have been successful. One of the critical systems, which is directly linked to the success of a mission is the power supply subsystem. Due to their construction, the electrical power supply subsystem shows failures related to energy maintenance, due to the limited area of the solar panels and small batteries. In Space Engineering, verification and validation activities are supported by simulations at differents life cycle phases that allow anticipating problems. The Simulators provide synthetic environment to test systems and that can be composed of virtual models and/or physical models. Due to the deficiency of methods and procedures necessary to increase the number of pico and nanosatellites missions, this dissertation proposes a framework for requirements verification and model validation combining simulations and automated tests. The framework provides three setup tests possibilities: only virtual models, virtual models with physical interface model and hardware-in-the-loop. The framework still covers: (i) definition of rules for modeling, providing reuse, (ii) combination of models and test functions, allowing test repeatability and (iii) execution of simulation and testing with automated results collection. As a case study we used the electrical power subsystem of Tancredo 1, a Brazilian picosatellite in development with INPE's support. The developed computer programs, subsystem models, requirements, test cases, and execution results are illustrated with the application of the framework to real case.

Keywords: Automated Functional Testing. Simulation. Space Systems Validation. Space Systems Verification. Pico and nanosatellite. Electrical Power Subsystem. Hardware-in-the-loop.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE ABREVIATURAS E SIGLAS

AC	–	Corrente Alternada (*Alternating Current*)
AD	–	Analógico-Digital
AEB	–	Agência Espacial Brasileira
AIT	–	*Assembly, Integration and Test*
AOCS	–	*Attitude and Orbit Control Subsystem*
COTS	–	*Commercial off-the-shelf*
DAE	–	Divisão de Aeronomia
DET	–	*Direct Energy Transfer*
DC	–	Corrente Contínua (*Direct Current*)
DJF	–	*Design Justification File*
DOD	–	*Depth-of-discharge*
DoD	–	*Department of Defense*
DSM	–	*Design Structure Matrix*
ECSS	–	*European Cooperation for Space Standardization*
EPS	–	*Electrical Power Subsystem*
FEE	–	*Front-End Equipment*
FVT	–	*Functional Verification Testbench*
GEO	–	*Geostationary Earth Orbit*
GSE	–	*Ground Support Equipment*
HIL	–	*Hardware-in-the-loop*
Hz	–	Hertz
I/O	–	Entrada e Saída (*Input* e *Output*)
INPE	–	Instituto Nacional de Pesquisas Espaciais
JAXA	–	*Japan Aerospace eXploration Agency*
LEO	–	*Low Earth Orbit*
MEF	–	Máquina de Estados Finitos
M&S	–	Modelagem e Simulação
NASA	–	*National Aeronautics and Space Administration*
OBC	–	*On Board Computer*
OBDH	–	*On Board Data Handling*
PCB	–	*Printed Circuit Boards*
PCDU	–	*Power Control and Distribuition Unit*
PDR	–	*Preliminary Design Review*
PPT	–	*Peack-Power Tracker*
PRR	–	*Preliminary Requirement Review*
RF	–	Rádio Frequência
UHF	–	*Ultra High Frequency*
V&V	–	Verificação e Validação

SUMÁRIO

Pág.

1 INTRODUÇÃO

De acordo com a norma ECSS (2009c), o ciclo de vida de um projeto na área espacial consta de sete fases: (0) Análise de missão/identificação das necessidades; (A) Análise de viabilidade; (B) Definição preliminar do projeto; (C) Definição detalhada; (D) Produção e qualificação; (E) Operação; e, finalmente, (F) Descarte.

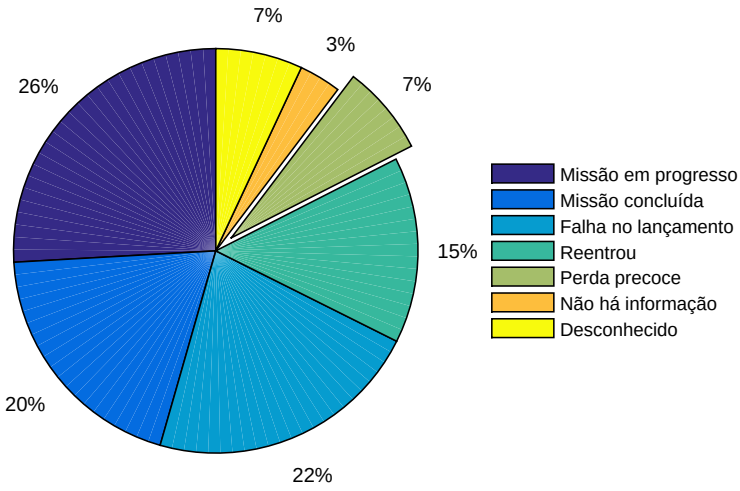
Em cada uma das fases são realizadas diversas atividades de verificação, cujo objetivo é garantir que o projeto atenda os requisitos para os níveis especificados (ECSS, 2010a). Uma técnica importante que apoia as atividades de verificação é a simulação, que permite medir o desempenho e/ou avaliar as funções do artefato sob verificação (EICKHOFF, 2009). Além disso, de acordo com a ECSS (2010a), as atividades de simulação podem desempenhar um papel fundamental no que tange os seguintes aspectos:

- Análise, definição e validação de requisitos de sistemas e técnicos.
- Validação do projeto (elétrico, térmico, mecânico, operacional) com relação ao atendimento dos requisitos de alto nível.
- Verificação e validação de *software*.
- Desenvolvimento de equipamentos de suporte e procedimentos de teste.
- Apoio às atividades de teste de subsistemas de satélite e de unidades.
- Previsão do desempenho do sistema.
- Treinamento da equipe de operação do satélite.
- Desenvolvimento e validação de procedimentos operacionais.
- Identificação de problemas causados por falhas ou anomalias.

As atividades de verificação são aplicadas a todos os tipos de satélites, desde os satélites considerados convencionais até os miniaturizados. Atualmente, principalmente na academia, os satélites de pequeno porte, ou ainda, os pico e nanossatélites, têm desempenhado um importante papel no setor espacial. Essas classes de satélites se tornaram uma opção às universidades e organizações, devido ao seu rápido desenvolvimento associado a custos e riscos menores, permitindo aprendizado rápido e formação de recursos humanos, bem como a validação de tecnologias no espaço (MEHRPARVAR et al., 2014; BÜRGER, 2014; SCHOLZ; JUANG, 2015).

Com o crescente número de pico e nanossatélites produzidos, observa-se também, falhas em um grande número deles. Swartwout (2015) mostra que entre 2000 e 2015, pelo menos 7% dos *CubeSats* lançados ao espaço foram perdidos antes de completarem sua missão, conforme mostrado no gráfico da Figura 1.1.

Figura 1.1 - Estado das Missões de *CubeSat*.



Estado das Missões de *CubeSat*, entre 2000 e 2015, destacando os satélites que foram perdidos antes de completarem sua missão.

Fonte: Adaptada de Swartwout (2015).

Essa estatística exclui os satélites perdidos por falhas no veículo lançador e os satélites que reentraram na atmosfera.

Em aspectos gerais, entre 2000 e 2012, o subsistema de energia elétrica foi responsável por 17% das falhas em *CubeSats*, tais como, baterias e painéis solares não conectados adequadamente ao barramento e geração insuficiente de energia (SWARTWOUT, 2013). Já no caso de satélites geoestacionários lançados ao espaço, no período entre 1998 e 2002, um em cada quatro destes satélites apresentou problema no subsistema de energia (PATEL, 2004), o que indica que as atividades de verificação, para o subsistema de energia elétrica do satélite, desempenham um papel fundamental no desenvolvimento de um satélite.

O subsistema de energia elétrica de um satélite pode ter diversas configurações, quanto à regulagem, distribuição, tipo de controle e tipos componentes para sua fabricação.

Neste cenário, torna-se imprescindível um processo de verificação para garantir que o subsistema atenda aos requisitos. Porém, de acordo com estudos apresentados em Bürger (2014), os projetos de pico e nanossatélites, normalmente, não aplicam massivamente testes funcionais como método de verificação de requisitos. Somente os testes ambientais e alguns testes funcionais são realizados a fim de atender as exigências impostas pela agência lançadora do satélite.

Na revisão sobre verificação do subsistema de energia de pico e nanossatélites (ver Capítulo 3) observa-se que as atividades de modelagem e simulação não são comumente aplicadas ao desenvolvimento desta classe de satélites. Apenas Corpino e Stesina (2014) utilizam um simulador para verificação funcional, todavia, os testes são executados manualmente.

Frente a este cenário, esta dissertação propõe um *framework* para execução automática de testes no subsistema de energia elétrica de pico e nanossatélites, reforçando e facilitando a verificação funcional, a fim de evitar que ela seja deixada de lado, em detrimento a outras prioridades do projeto.

1.1 Objetivo

Esta dissertação tem por objetivo propor um *framework* para validação de um subsistema virtual de energia elétrica para pico e nanossatélites, bem como a verificação de seus requisitos funcionais. O *framework* deve permitir a verificação do subsistema de energia, por meio de testes executados automaticamente em um ambiente baseado em modelos e simulação. O desafio na elaboração do *framework* é combinar simulação e execução de teste, além disso, explorar a simulação como ferramenta de verificação e validação.

Alinhado à necessidade de verificação do subsistema de energia, a utilização de um simulador nas fases iniciais de desenvolvimento pode endereçar diversos benefícios, tais como, antecipação às falhas, balanço de potência e redução de retrabalho ao longo do projeto.

Alguns objetivos secundários desta dissertação são:

a) Definir regras de modelagem para descrever os equipamentos de um sub-

sistema de energia elétrica de forma virtual;

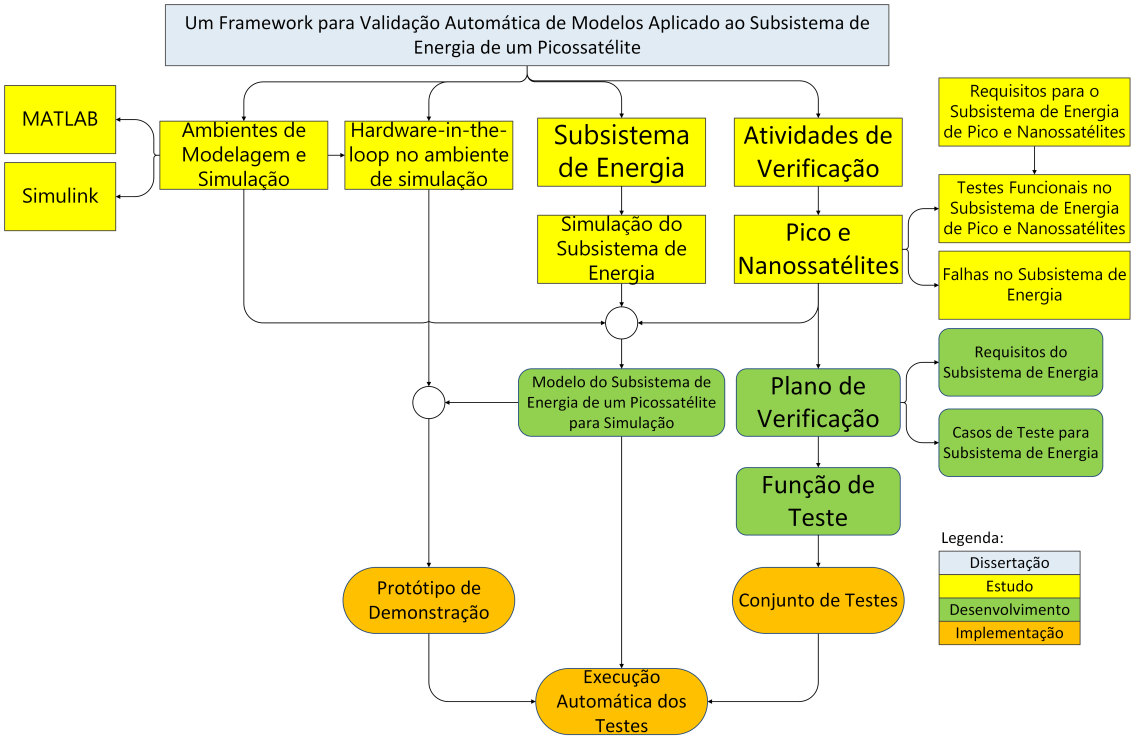
b) Demonstrar o uso de um equipamento de interface entre o ambiente de simulação virtual e o modelo físico do subsistema, definido em *software*;

c) Definir uma estratégia de organização e execução de uma rodada de simulação, com acoplamento automático de testes.

1.2 Metodologia

As atividades (ou áreas de conhecimento abordadas) para a realização desta Dissertação foram divididas em: pesquisa, desenvolvimento e demonstração de resultado. Uma visão geral das atividades desenvolvidas no contexto deste trabalho pode ser observada na Figura 1.2.

Figura 1.2 - Atividades da Dissertação.



A figura apresenta as principais atividades realizadas nesta dissertação.
Fonte: Produção do autor.

Inicialmente, realizou-se um embasamento teórico e revisão bibliográfica dos temas abordados na dissertação, como pode ser observado nas atividades identificadas em "Pesquisa" da Figura 1.2.

O embasamento teórico contempla o estudo do subsistema de energia, os aspectos que tangem a verificação de um subsistema, os conceitos de modelagem e simulação (M&S), incluindo os ambientes de M&S e o uso de *hardware*.

Uma revisão bibliográfica em verificação de pico e nanossatélites, e as principais causas de falha no subsistema de energia, foi então realizada.

As atividades de "Desenvolvimento", que contemplam: (i) implementação dos modelos para simulação do subsistema de energia elétrica, (ii) levantamento dos requisitos típicos do subsistema, (iii) criação do plano de verificação, que inclui os requisitos do subsistema, os casos de teste, e indicação de como realizar a verificação do subsistema. Cabe observar que, os casos de teste foram tratados tanto em sua forma abstrata como concreta, traduzidos para *scripts* executáveis. Nesta dissertação os *scripts* são denominados de funções de teste, as quais serão incorporadas ao modelo do subsistema para execução de teste.

As últimas atividades contam da obtenção dos resultados para demonstração da viabilidade do *framework*, dessa forma, verificando os requisitos selecionados, bem como, a validação dos modelos implementados.

1.3 Organização do Texto

O texto desta dissertação está organizado da seguinte forma:

- Capítulo 2 - FUNDAMENTAÇÃO TEÓRICA: apresenta os conceitos fundamentais para o entendimento do trabalho.
- Capítulo 3 - SUBSISTEMA DE ENERGIA ELÉTRICA: traz os conceitos relacionados ao subsistema de energia de um satélite e uma revisão sobre o subsistema de energia de satélites miniaturizados, enfatizando algumas falhas e aspectos de verificação do subsistema.
- Capítulo 4 - *FRAMEWORK* PARA EXECUÇÃO VALIDAÇÃO AUTOMÁTICA DE MODELOS: apresenta o *framework* proposto para verificação funcional do subsistema de energia de um satélite.
- Capítulo 5 - *APLICAÇÃO DO FRAMEWORK PARA O SUBSISTEMA*

DE ENERGIA ELÉTRICA DE PICO E NANOSSATÉLITES: mostra a aplicação do *framework* para o subsistema de energia elétrica de um satélite.

- Capítulo 6 - CONCLUSÃO: discute a solução desta dissertação, as vantagens e os pontos a serem melhorados neste *framework*.
- APÊNDICE A - ESTUDO SOBRE O SATÉLITE TANCREDO 1: descreve o picossatélite Tancredo 1.
- APÊNDICE B - MODELO DE REFERÊNCIA DO EPS: apresenta o modelo dinâmico do EPS, implementado no *Simulink*, utilizado como referência.
- APÊNDICE C - EXEMPLO DO PLANO DE VERIFICAÇÃO: apresenta o conteúdo de um Plano de Verificação.
- APÊNDICE D - PUBLICAÇÕES: lista as publicações realizadas durante o desenvolvimento desta dissertação.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentados alguns conceitos importantes relacionados ao tema da Dissertação.

2.1 Satélites Artificiais

O satélite é um objeto construído para orbitar algo ou realizar uma viagem no espaço (ECSS, 2012a). Satélites artificiais[1] são sistemas construídos pelo homem, que ficam na órbita da terra ou de qualquer outro planeta. Um satélite artificial é organizado em subsistemas, conforme ilustra o diagrama da Figura 2.1.

Figura 2.1 - Diagrama simplificado de um Satélite.

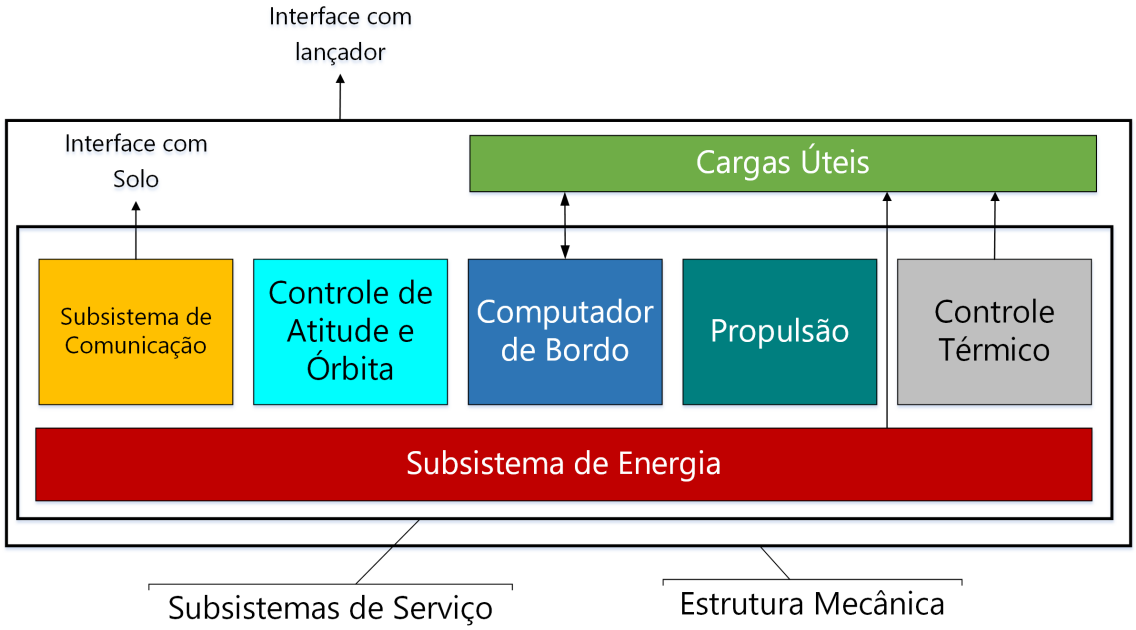


Diagrama de blocos simplificado de um satélite composto por: cargas úteis, subsistemas de serviço, interface com veículo lançador e interface com os sistemas de solo.

Fonte: Adaptada de Wertz e Larson (1999), Fortescue et al. (2004) e Azevedo (2014).

De maneira geral, para Wertz e Larson (1999), os satélites são compostos pelos seguintes subsistemas: comunicação, controle de órbita e atitude (AOCS, do inglês

[1]Disponível em: http://www.inpe.br/acessoainformacao/node/405. Acesso em: 29 out. 2015

Attitude Orbit Control Subsystem), computador de bordo (OBDH, do inglês *On Board Data Handling*), propulsão, controle térmico, energia e estrutura. Na Tabela 2.1, encontra-se uma breve descrição destes subsistemas.

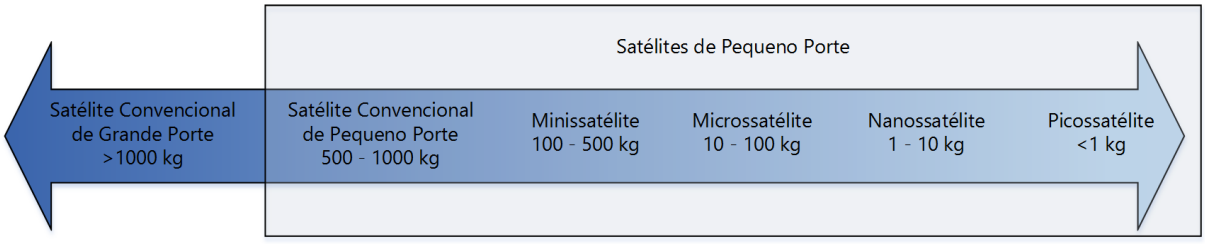
Tabela 2.1 - Descrição dos Subsistemas.

Subsistema	**Descrição**
Propulsão	Fornece o impulso para a correção da órbita, atitude e para o gerenciamento do momento angular.
AOCS	Responsável pela determinação e controle de atitude, posição orbital e apontamentos.
Estrutura	Fornece estrutura de suporte, interface e partes móveis.
Comunicação	Provê a comunicação entre o satélite e o segmento solo, ou também outro veículo espacial. Também é responsável pelo rastreio do satélite.
OBDH	Processa e distribui comandos e processa, armazena e formata dados.
Térmico	Mantém os equipamentos operando dentro dos limites de temperatura permitidos.
Energia Elétrica	Gera, armazena, regula e distribui energia elétrica.

Fonte: Adaptada de Wertz e Larson (1999).

Os satélites podem ser classificados de acordo com a sua massa em: grande porte, pequeno porte, minissatélite, microssatélite, nanossatélite e picossatélite, como pode ser observado na Figura 2.2. (FORTESCUE et al., 2004)

Figura 2.2 - Classificação de satélites.



Classe de satélites de acordo com sua massa.
Fonte: Adaptada de Fortescue et al. (2004).

2.1.1 Satélites Miniaturizados: Nano e Picossatélites

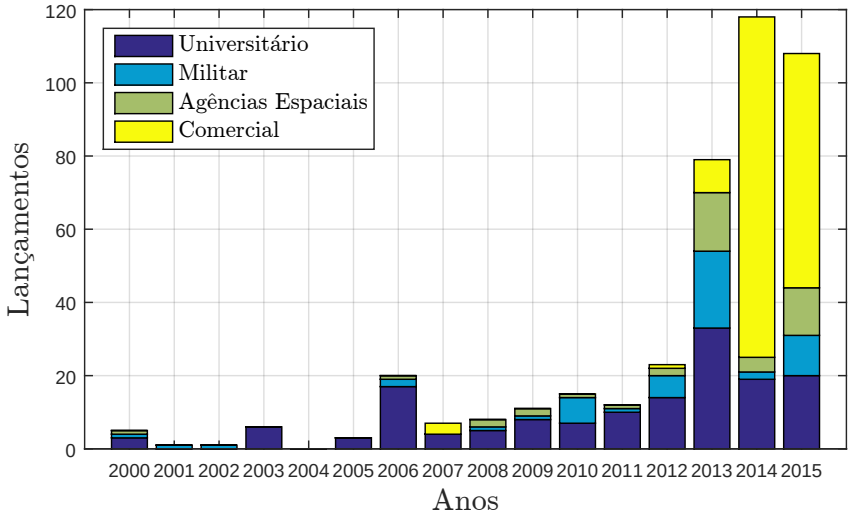
O fator principal para o desenvolvimento de pico e nanossatélites satélites é a redução de custos com o lançamento, projeto, construção, testes, operação e interface com o segmento solo, bem como, a possibilidade de testar novas tecnologias em ambiente espacial. (FORTESCUE et al., 2004)

O desenvolvimento de satélites miniaturizados tornou-se possível graças aos avanços na diminuição dos componentes eletrônicos, o aumento de sua capacidade e o baixo consumo de energia. (HEIDT et al., 2000)

Dentre os satélites miniaturizados, o padrão do *CubeSats*, foi projetado para ser reutilizado intensamente. Este padrão foi criado em 1999, através da colaboração entre os professores Jordi Puig-Suari, da *California Polytechnic State University* (*Cal Poly*) e Bob Twiggs, da *Stanford University*. O *CubeSat* possui 10 cm de aresta (HEIDT et al., 2000; MEHRPARVAR et al., 2014) e possui uma interface padrão para o lançamento, a plataforma P-POD (*Poly Picosatellite Orbital Deployer*) (NASON et al., 2002; MEHRPARVAR et al., 2014).

De acordo com Scholz e Juang (2015) e Swartwout (2015), o número de lançamentos de nano e picossatélites já ultrapassa 400 unidades, como pode ser observado na Figura 2.3.

Figura 2.3 - Pico e Nanossatélites lançados.



O gráfico apresenta a quantidade de nano e picossatélites lançados desde o ano 2000.
Fonte: Adaptada de Scholz e Juang (2015), Swartwout (2011) e Swartwout (2015).

Dentre as iniciativas brasileiras de desenvolvimento de *CubeSats*, têm-se: o NanoSatC-Br1 (COSTA et al., 2014), o AESP-14 (BÜRGER et al., 2015) e o Serpens[2] já lançados, enquanto que o ConaSat[3] (CARVALHO et al., 2013), o ITASat (GENTINA et al., 2009) e NanoSatC-Br2 (PIAZ et al., 2015) ainda estão em desenvolvimento. O primeiro *CubeSat* brasileiro foi o NanoSatC-Br1, lançado em 2014 e ilustrado na Figura 2.4 (FRANKE et al., 2014).

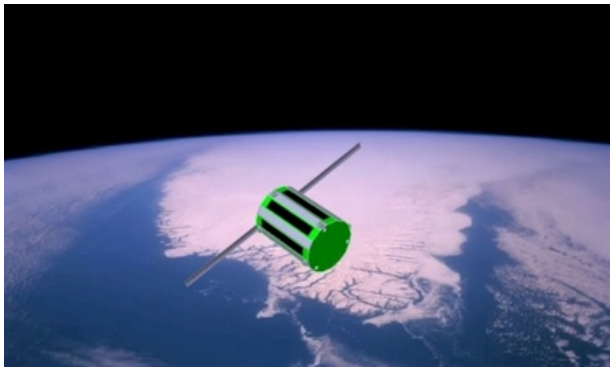
Figura 2.4 - *CubeSat* NanoSatC-BR1.

Em destaque, foto da versão final do primeiro *CubeSat* brasileiro.
Fonte: Franke et al. (2014).

Na categoria dos picossatélites, tem-se a plataforma desenvolvida pela *InterOrbital Systems*, conhecida como *TubeSat* (WOELLERT et al., 2011), cujo modelo é ilustrado na Figura 2.5.

[2]Disponível em: `http://goo.gl/lvHx8R`. Acesso em 20 nov. de 2015
[3]Constelação de Nanossatélites para Coleta de Dados Ambientais

Figura 2.5 - *TubeSat*.



Modelo artístico de um *TubeSat*.

Fonte: Sítio eletrônico da INTERORBITAL SYSTEMS (2015).

No Brasil há, pelo menos duas iniciativas de desenvolvimento de *TubeSats*: o *TubeSat* da Universidade de São Paulo[4] e o Tancredo 1 (SANTOS et al., 2011), que está sendo desenvolvido pela Escola Municipal Presidente Tancredo de Almeida Neves, de Ubatuba, em parceria com o INPE. O subsistema de energia elétrica do Tancredo 1 é utilizado no estudo de caso desta dissertação. Mais detalhes deste satélite são apresentados no Apêndice A.

2.2 Verificação e Validação de Sistemas Espaciais

A validação demonstra que "o sistema certo foi feito", enquanto que a verificação demonstra que "o sistema foi feito certo". (NASA, 2007)

De acordo com ECSS (2009a), as atividades de verificação demonstram, através de um processo dedicado, que o produto atende as especificações. Os objetivos deste processo são:

- Garantir que o produto está de acordo com o projeto, está livre de defeitos de fabricação e aceitável para uso;
- Confirmar a integridade e desempenho do produto em determinadas fases do ciclo de vida;
- Confirmar que todo o sistema é capaz de cumprir os requisitos da missão.

[4]Disponível em: http://www.interorbital.com/interorbital_06222015_014.htm. Acesso em: 11 nov. 2015

As atividades de verificação são realizadas durante todo o ciclo de vida de desenvolvimento de um produto espacial. Assim, é fundamental documentar todo o processo de verificação para implementação e acompanhamento do projeto. Entre os documentos têm-se (ECSS, 2009a; ECSS, 2009b; ECSS, 2012b):

- Plano de Montagem, Integração e teste: é o plano mestre para o processo de montagem, integração e teste (AIT - *Assembly Integration and Test*). Ele descreve o processo de AIT completo e demonstra quais requisitos são verificados por inspeção e teste. Ele contém as atividades de AIT, as ferramentas de verificação (GSE e instalações), a documentação envolvida, a gestão e a programação das atividades.

- Documento de Controle de Verificação: apresenta uma lista dos requisitos que devem ser verificados, os métodos selecionados de acordo com o estágio e os níveis definidos.

- Especificação de Teste: descreve detalhadamente os requisitos de teste aplicáveis a qualquer grande atividade de teste. Define, também, o objetivo do teste, o método para testar, o produto em teste, o GSE necessário, os instrumentos e a precisão de medição, a sequência de teste, a documentação necessária, os critérios de passou/falhou (*pass/fail*), os participantes e o cronograma de testes.

- Procedimento de Teste: fornece as instruções para a realização dos testes, a saber, descrições, recursos, restrições e passo a passo para execução.

- Relatório de Teste: descreve o resultado da execução do teste, a avaliação dos resultados e as conclusões em relação aos requisitos de teste.

- Relatório de Análise: descreve, para cada análise, os pressupostos relevantes, os métodos e técnicas utilizadas e os e resultados.

- Relatório de Revisão de Projeto: descreve as atividades de verificação realizadas por revisão de documentação.

- Relatório de Inspeção: descreve cada atividade de verificação realizada para inspecionar *hardware* e *software*.

- Relatório de Verificação: é preparado quando mais de um método de verificação definido é utilizado para verificar um requisito ou um conjunto específico de requisitos.

- Plano de Verificação (*Verification Plan* - VP): pode conter a abordagem de verificação, a filosofia de modelo, a matriz de produto, as estratégias de verificação

dos requisitos, ou seja, a inter-relação entre os diferentes métodos, níveis e estágios de verificação, os métodos, as ferramentas e a metodologia de controle da verificação. (ECSS, 2009a). O VP foi utilizado, no contexto desta Dissertação, para documentar os requisitos e relacionar os testes a serem executados no subsistema.

2.2.1 Plano de Verificação

O plano de verificação é um artefato fundamental no processo de verificação e validação de um sistema espacial.

A Tabela 2.2 apresenta a descrição do conteúdo do plano de verificação.

Tabela 2.2 - Conteúdo do Plano de Verificação.

Item	**Descrição**
Abordagem de Verificação	É definida nas fases iniciais de desenvolvimento através da análise dos requisitos a serem verificados. Leva em consideração: (i) restrições do projeto; (ii) status de qualificação das soluções candidatas; (iii) disponibilidade e maturidade das ferramentas de verificação; (iv) metodologias de verificação; e (v) custo.
Métodos de Verificação	Define o método pelo qual a verificação pode ser executada, como: teste, análise, revisão de projeto e inspeção.
Níveis de Verificação	Os níveis dependem da complexidade e características do projeto. Usualmente, para um produto espacial, a verificação é realizada nos seguintes níveis: equipamento, subsistema, elemento, segmento e sistema.
Etapas de Verificação	O processo de verificação é executado em etapas subsequentes ao longo do ciclo de vida do projeto. Estas etapas são divididas em: qualificação, aceitação, pré−lançamento e órbita.
Filosofia de Modelos	Define os modelos (estrutural, engenharia, qualificação, etc) contra os quais os testes são executados.
Ferramentas de Verificação	Ferramentas que são utilizadas para executar as atividades de verificação. Elas devem ser mencionadas neste documento para que sua aquisição seja feita e sua utilização seja planejada.

Fonte: Adaptada de ECSS (2009a).

2.2.1.1 Etapas de Verificação

A verificação é realizada através da seleção de etapas apropriadas para o projeto. A Tabela 2.3 descreve o que a verificação deve demonstrar nas etapas de um projeto espacial.

Tabela 2.3 - Etapas da Verificação.

Etapa	**Descrição**
Qualificação	O projeto, incluindo as tolerâncias, cumpre os requisitos aplicáveis.
Aceitação	O produto está livre de erros de fabricação e está pronto para uso operacional.
Pré-lançamento	A verificação deve demonstrar que o produto está configurado para as atividades de lançamento e operações iniciais.
Órbita	Não houve degradação durante o lançamento, nas órbitas iniciais e antes de iniciar sua operação.

Fonte: Adaptada de ECSS (2009a).

2.2.1.2 Filosofia de Modelo

A filosofia de modelos define o número ótimo e as características dos modelos físicos, necessários para atingir determinado nível de confiança durante a verificação do produto. A filosofia de modelo é definida através de um processo iterativo que combina restrições programáticas, estratégias de verificação e os programas de integração e testes, levando em consideração o estado de desenvolvimento da solução escolhida para o projeto. A Figura 2.6 ilustra os parâmetros que devem ser considerados. (ECSS, 2010b)

A Tabela 2.4 apresenta a descrição dos modelos que podem ser utilizados durante o processo de verificação e o nível em que são aplicados.

Tabela 2.4 - Tipos de Modelo.

Modelo	**Descrição**	**Nível (s)**
Mock-up (MU)	São utilizados em apoio à definição de projeto para toda análise de arquitetura, configuração e avaliação do projeto, otimização do *layout* e avaliação dos procedimentos operacionais.	Sistema/elemento
Modelo de desenvolvimento (DM)	São utilizados para desenvolvimento de um novo conceito ou quando se faz necessário um *redesign*.	Todos os níveis

(Continua)

Tabela 2.4 – *Continuação.*

Modelo	**Descrição**	**Nível (s)**
Modelo estrutural	É totalmente representativo do produto final, no que diz respeito aos aspectos estruturais. É utilizado para qualificação do projeto estrutural.	Subsistema
Modelo térmico	É totalmente representativo das propriedades térmicas do produto final. É utilizado para qualificação do projeto térmico.	Subsistema
Modelo elétrico e funcional	Representa o produto final, em termos de suas funções, tanto nos aspectos elétricos, quanto nos de *software*. São utilizados para testes funcionais e de interface, e também, para investigações em modo de falha. Parte pode ser simulada por *software* e o *hardware* pode ser construído com peças comerciais.	Todos os níveis
Modelo de engenharia	Representa o modelo de voo em termos de forma, funções e tamanho, porém sem utilizar peças de alta confiabilidade e, geralmente, sem redundância total. Este modelo é utilizado para qualificação funcional e demonstração de sobrevivência às falhas.	Todos os níveis
Modelo de qualificação	Reflete, completamente, o produto final em todos os aspectos. É usado para testes completos de qualificação funcional e ambiental. Normalmente utilizado para equipamentos e subsistemas recém-projetados.	Equipamento/subsistema
Modelo de Voo	É o produto final de voo. É submetido a testes de aceitação ambiental e funcional.	Todos os níveis
Modelos virtuais e híbridos	No domínio funcional e elétrico, modelos de simulação são utilizados para o desenvolvimento e verificação. Estes modelos podem existir tanto na configuração *software* puro (simulador) ou em uma combinação de *software* e componentes de *hardware*.	Todos os níveis

(Continua)

Tabela 2.4 – *Continuação.*

Modelo	Descrição	Nível (s)
Modelos do segmento Solo	São dedicados para uma finalidade específica no processo de verificação do segmento solo, como por exemplo, a validação do processamento de telecomandos e telemetrias.	Segmento solo

Fonte: Adaptada de ECSS (2009a).

Os vários níveis de equipamento, *breadboards* e modelos de desenvolvimento podem ser utilizados para identificar problemas nas etapas inicias de desenvolvimento. Enquanto que na etapa de qualificação é necessário um modelo dedicado. Em termos de verificação de sistema pode-se reutilizar os modelos de engenharia que têm alta fidelidade, para evitar impactos nos custos, ao invés de utilizar componentes de alta confiabilidade como exigido nos modelos de qualificação e voo.

A filosofia de modelo influencia fortemente o cronograma do projeto, ou seja, o uso de um modelo em níveis diferentes pode aumentar a duração do cronograma, ao passo que, o uso paralelo de vários modelos pode encurtá-lo. A filosofia de modelos deve sempre buscar equilibrar o cronograma, o custo e os riscos. (ECSS, 2010b)

2.2.1.3 Ferramentas de Verificação

As ferramentas de verificação são aquelas utilizadas para apoiar a execução do processo de verificação. Algumas dessas ferramentas são: Equipamentos de Suporte (*Ground Support Equipment* - GSE); Sistemas para Validação de *Software*; Simuladores; *Software* para verificação por análise; Ferramentas de teste. (ECSS, 2009a)

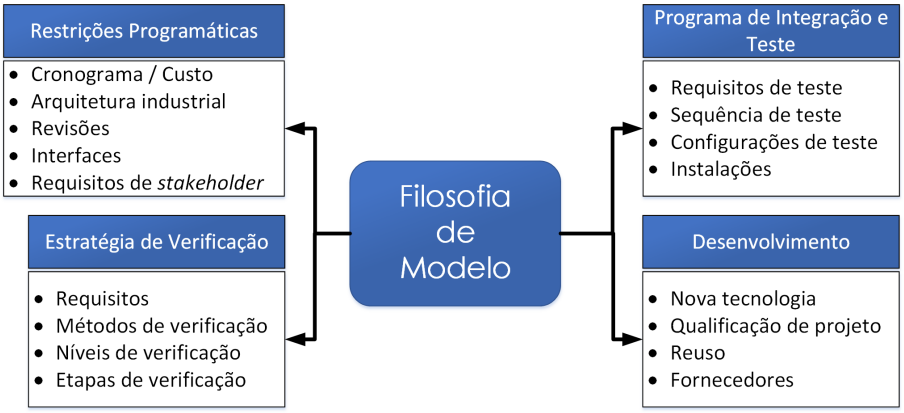
2.3 Modelagem & Simulação

Tradicionalmente, em projetos espaciais, a atividade de Modelagem & Simulação (M&S) é considerada como uma disciplina de suporte, aplicada a diversas áreas do projeto e em diferentes fases do ciclo de vida, da concepção à operação. O uso de M&S ao longo do ciclo de vida de um projeto na área espacial pode gerar benefícios significativos, tais como, redução de riscos e custos (ECSS, 2010a).

Para o DoD (2011) os termos modelagem e simulação são definidos da seguinte maneira:

- Modelagem: Aplicação de uma metodologia padronizada, rigorosa e estruturada para criar e validar a representação física, matemática ou lógica de um sistema, entidade, fenômeno ou processo.

Figura 2.6 - Filosofia de Modelos.



Parâmetros que devem ser considerados para definição da filosofia de modelos.
Fonte: Adaptada de ECSS (2010b).

- Simulação: Método para executar um modelo ao longo do tempo. Durante a simulação eventos podem ser inseridos pelo usuário, por um *script*, *hardware* externo ou outra simulação. (ECSS, 2010a)

- Modelo: É a descrição ou representação feita a partir da modelagem, e pode ser físico, matemático, lógico ou computacional.

2.3.1 Simuladores de Satélite ao Longo do Ciclo de Vida

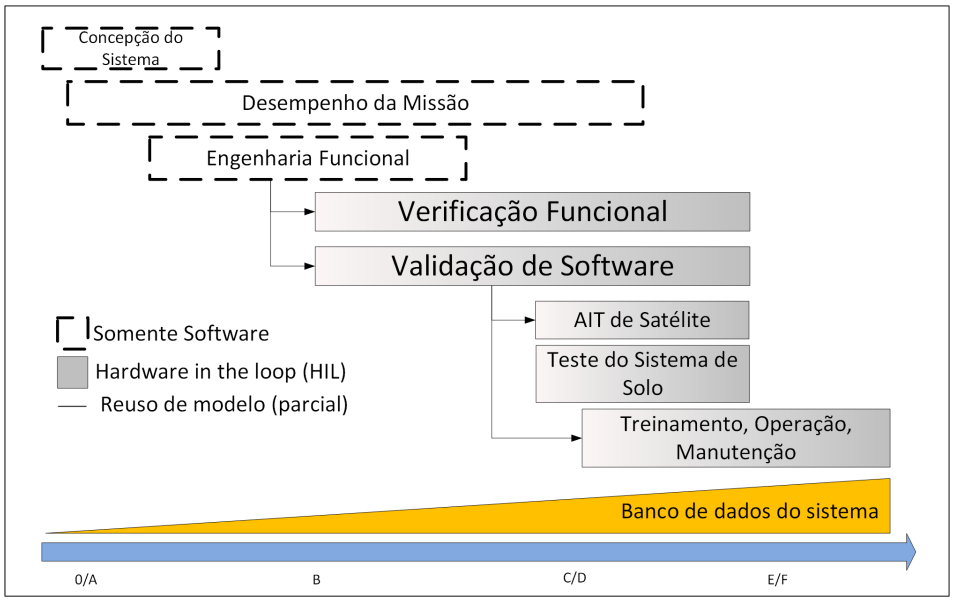
De acordo com NASA (2007), dividir o ciclo de vida em fases permite que vários produtos de um projeto possam ser desenvolvidos gradualmente e vão amadurecendo desde os conceitos iniciais até sua operação.

Os simuladores que podem ser utilizados durante o ciclo de vida de um satélite estão divididos nos seguintes grupos: análise e concepção de missão, qualificação e aceitação do veículo espacial, qualificação e teste (montagem, integração e teste) do sistema de solo e operacionais (ver Figura 2.7). O conjunto dos simuladores que compõem estes grupos constitui o **modelo virtual do sistema**, o que reflete as funções e comportamento completo do sistema para apoiar as atividades de análise e verificação. Assim, é possível reusar os modelos nas fases seguintes do ciclo de vida. O modelo virtual do sistema deve ser considerado na filosofia de modelos e, no decorrer das fases ele pode assumir diferentes configurações (ECSS, 2010a).

A Figura 2.7 também indica que alguns modelos podem ser reutilizados durante o ciclo

de vida de um projeto e que o banco de dados do sistema cresce ao longo do tempo.

Figura 2.7 - Simuladores ao Longo do Ciclo de Vida.



Simuladores nas fases do ciclo de vida de um produto espacial.
Fonte: Adaptada de ECSS (2010a).

Dentre as categorias de simuladores, destaca-se o simulador para verificação funcional, descrito a seguir, tendo em vista que, os conceitos deste simulador são utilizados para o desenvolvimento do *framework* apresentado neste trabalho.

2.3.2 Simulador para Verificação Funcional

De acordo com Eickhoff (2009) e ECSS (2010a), os simuladores podem ser empregados para auxiliar o processo de V&V durante o ciclo de vida de um produto espacial. Eickhoff (2009) afirma que, o desenvolvimento e a verificação de um artefato espacial são realizados de forma gradual, dessa forma, têm-se as seguintes fases:

- (i) modelagem do sistema em uma linguagem independente de *hardware* (*algorithm in the loop*);
- (ii) algoritmos codificados na linguagem do *hardware* (*software in the loop*);
- (iii) código embarcado em um *hardware* representativo do equipamento real

(*controller in the loop*); e por fim,

- (iv) código final embarcado no *hardware* para controlar o sistema real (*hardware-in-the-loop)*.

No contexto desta Dissertação, as fases citadas, apoiam o *framework* no que diz respeitos às configurações de teste utilizadas para realizar a validação dos modelos. As configurações de teste são apresentadas no Capítulo 4.

De acordo com a ECSS (2010a), entre as fases B e C[5] (Projeto Preliminar Projeto Detalhado, respectivamente) do ciclo de vida de desenvolvimento de um produto espacial desenvolve-se o Simulador para Verificação Funcional (*Functional Verification TestBench* - FVT), Figura 2.8.

No FVT a simulação foca os elementos críticos permitindo análises de desempenho no sistema e testes no subsistema.

Figura 2.8 - Simulador para Verificação Funcional.

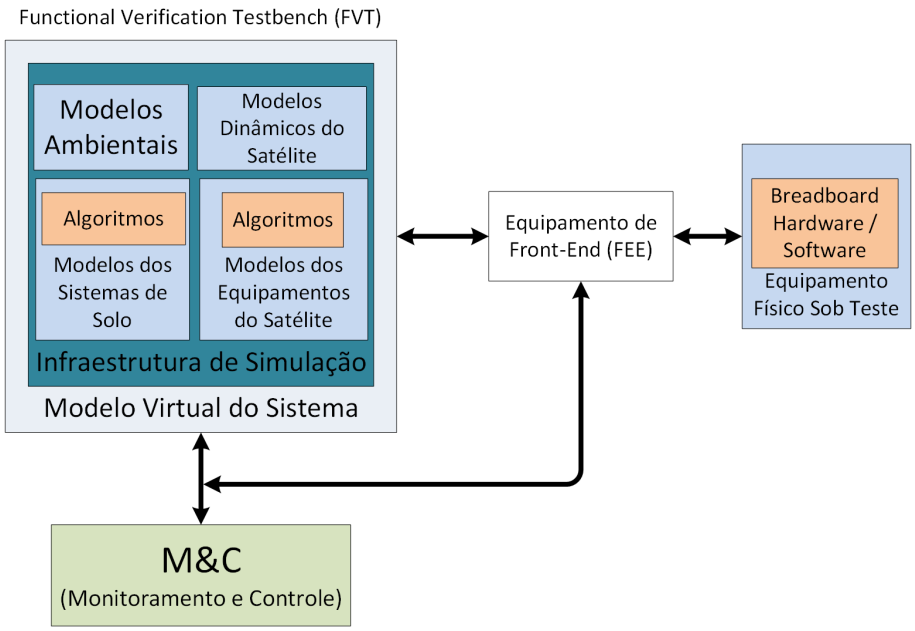


Diagrama do FVT.
Fonte: Adaptada de ECSS (2010a).

[5]As revisões entre as fases B e C são: PRR (*Preliminary Requirements Review*) e PDR (*Preliminary Design Review*). As revisões são realizadas para avaliar a situação global do projeto em relação às metas e exigências (ECSS, 2009c)

O FVT é composto, basicamente, por (ECSS, 2010a):

- *Virtual System Model* (Modelo Virtual do Sistema): simula as funções e o comportamento do sistema a ser construído e contém a infraestrutura de simulação e os modelos do sistema.
- *Simulation Infrastructure* (Infraestrutura de Simulação): interfaceia os modelos e o sistema de monitoramento e controle;
- Modelos: (i) modelos ambientais (*environmental models*), (ii) modelos dinâmicos do satélite (*spacecraft dynamics models*), (iii) modelos dos equipamentos do satélite (*spacecraft equipments models*) e (iv) modelos do sistema de solo (*ground system model*);
- M&C (monitoramento e controle): é a interface entre o usuário e o simulador;
- *Front-End Equipment* (FEE) (Equipamento de interface): interface entre o simulador e o *hardware*;
- *Physical Equipment Under Test* (Equipamento Físico Sob Teste): é um equipamento real que está interligado ao simulador, e é estimulado de acordo com o que ocorrer na simulação virtual.

O FVT pode adotar duas configurações, a primeira contém somente modelos computacionais (virtuais), chamada de *software only* (somente *software*), a segunda contém um *hardware* na malha de simulação (*hardware-in-the-loop*).

A simulação é realizada pela parte computacional, que estimula e coleta respostas do *hardware* em teste, através de um equipamento de interface (FEE). As respostas são retornadas à simulação computacional que realiza as comparações e verifica as funções (EICKHOFF, 2009).

O sistema ou subsistema em teste evolui da pura representação lógico-numérica, até que cada item crítico seja substituído pelo *hardware*, podendo ser somente um protótipo ou o equipamento final. Uma das vantagens desta solução é o fluxo contínuo de verificação, sem intervalos causados por indisponibilidade de *hardware*. Entretanto, cabe observar a existência de um nível de incerteza, que sempre permanece, mesmo que a simulação apresente bons resultados em relação ao produto final. (ECSS, 2010b)

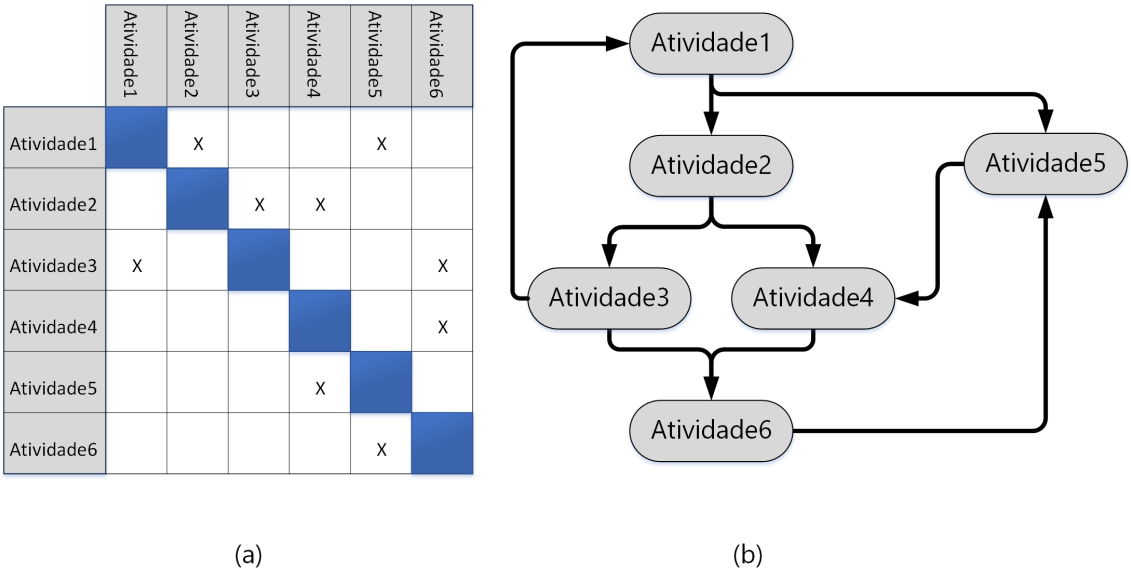
Dentre as principais atividades a serem realizadas com o apoio do FVT destacam-se: (i) desenvolvimento, teste e integração dos modelos funcionais e das interfaces específicas do elemento sob teste; (ii) desenvolvimento/adaptação dos modelos de dinâmica, órbita e

ambiental; (iv) desenvolvimento de cenários de teste (ECSS, 2010a). No caso desta Dissertação, foram acrescentados facilidades de validação de modelos lógicos representando o subsistema de energia elétrica.

2.4 Matriz de Estrutura de Projeto

A Matriz de Estrutura de Projeto, do inglês *Design Structure Matrix* (DSM), ilustrada na Figura 2.9a, através de uma representação simples, compacta e visual, fornece um método que captura a relação entre os componentes de um sistema complexo, suas interações, interdependência e interface (QIAO; RYAN, 2015; DSM, 2009a). A Figura 2.9b representa a leitura feita a partir da DSM da Figura 2.9a.

Figura 2.9 - DSM Simples.



(a) Representação de uma DSM; (b) Interpretação da DSM (a) em fluxo de dados.
Fonte: Adaptada de DSM (2009b).

Pode-se destacar o uso da DSM em aplicações de engenharia, tais como, projeto de uma aeronave (MARTINS, 2010), análise de uma missão espacial (DENG; YANG, 2014; QIAO; RYAN, 2015) e desenvolvimento de *software* (HWANG, 2014).

2.5 Testes de *Software*

De acordo com Hayes (2012) a automatização dos testes, no desenvolvimento de *software*, favorece o cumprimento dos prazos e das metas de qualidade. Se os testes são executados automaticamente, os benefícios são: a repetibilidade que reduz tempo e custos e melhoria da produtividade.

Entretanto, a automação dos testes não pode ser implementada se não existir um processo bem definido (COSTA, 2004). A execução de teste requer um conjunto de casos de teste

Os casos de teste constam de uma especificação completa e independente de ações requeridas para atingir um propósito específico de teste. Os casos de teste podem ser definidos como uma sequência de entradas e saídas esperadas do subsistema. Os casos de teste podem ser classificados em:

- Abstrato: o termo "abstrato" refere-se ao fato de que o caso de teste é gerado independente de plataforma, dessa forma, faz-se necessário traduzir o caso de teste para torná-lo executável.
- Executável: este tipo de caso de teste pode ser aplicado diretamente ao sistema em teste, ou seja, define um conjunto de instruções interpretáveis pelo computador que representam as ações que devem ser executadas.

Após a execução dos casos de teste têm-se o veredicto, que pode ser: (i) passou (a saída atende o requisito), (ii) falhou (a saída não atende o requisito) ou (iii) inconclusivo (a saída não permite dizer se o requisito foi atendido ou não).

Os casos de teste podem ser derivados a partir de requisitos ou de outros modelos. Muitas pesquisas usam modelos do tipo máquinas de estados finitos (MEFs) como fonte para geração de casos de teste, uma vez que, as MEFs representam um portamento previsível e são largamente utilizadas para modelar sistemas reativos. (MARTINS et al., 2004; AMBROSIO et al., 2005; PINHEIRO et al., 2014).

Há muitas pesquisas em teste de software, algumas referências que influenciaram a pesquisa dessa dissertação foram: Copeland (2004), Almeida (2004), Delamaro et al. (2007), Naik e Tripathy (2011), Narayani (2012), Rios et al. (2012), Corsetti (2014).

3 SUBSISTEMA DE ENERGIA ELÉTRICA

Este capítulo apresenta aspectos gerais relacionados ao Subsistema de Energia Elétrica de um satélite.

3.1 Visão Geral

Um dos subsistemas que compõe um satélite é o Subsistema de Energia Elétrica (do inglês, *Electrical Power Subsystem* - EPS). Este subsistema deve fornecer, armazenar, distribuir e controlar a energia elétrica em um satélite. Além disso, o EPS deve Wertz e Larson (1999):

- Fornecer conversores de energia AC (corrente alternada) e barramento DC (corrente contínua) regulado;
- Prover capacidades de comando e telemetria para análise da saúde e *status* do EPS, para fins de controle pela estação de solo ou sistema autônomo;
- Proteger as cargas do satélite contra falhas dentro do EPS;
- Suprimir os transitórios de tensão no barramento.

O EPS, ilustrado na Figura 3.1, tanto para satélites convencionais (WERTZ; LARSON, 1999) como para pico e nanossatélites (ULRICH et al., 2009; BURT, 2011a), é composto, basicamente, pelos seguintes elementos:

- Painel Solar: transforma energia solar em elétrica para o satélite. As células fotovoltaicas são largamente utilizadas e convertem a radiação solar em energia elétrica.
- Armazenador de energia: A forma mais utilizada para armazenar energia é através de baterias. Elas fornecem a demanda de energia nos períodos de eclipse.
- Distribuidor de Energia e Protetor de Falhas: garante que a energia seja distribuída às cargas do satélite. É constituído, basicamente, pelos cabos de distribuição de energia, elementos comutadores para ligar e desligar as cargas e sistema proteção contra falhas.
- Regulador e Controle: A regulagem da energia se divide em 3 categorias principais: controle do painel solar, regulagem da tensão de barramento e carregamento da bateria. A regulagem é realizada por um controlador que responde a um erro de sinal no barramento de tensão.

Figura 3.1 - Diagrama do EPS.

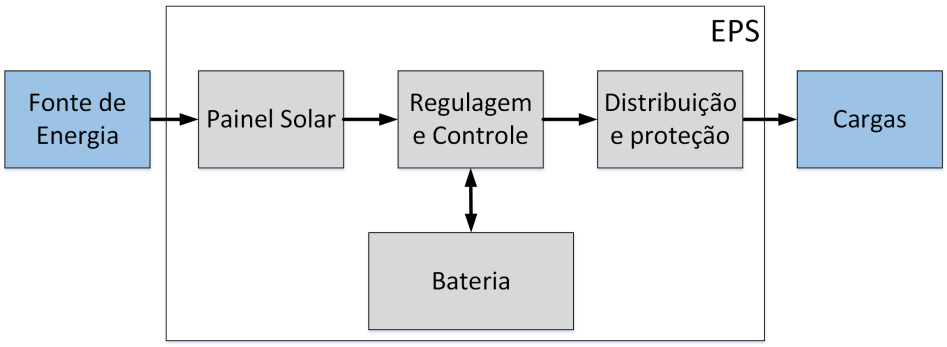


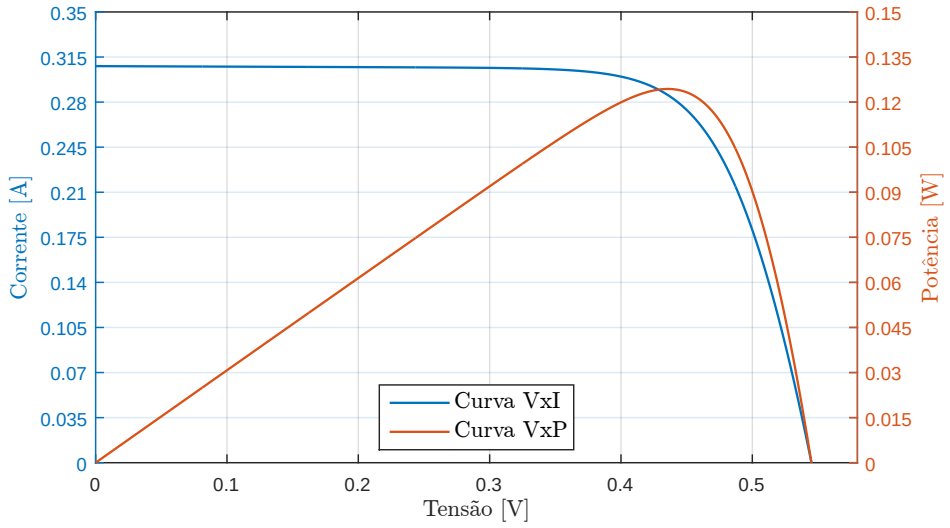
Diagrama de blocos simplificado do EPS de um satélite.
Fonte: Adaptada de Wertz e Larson (1999).

3.1.1 Painel Solar

O painel solar é composto por células solares. A quantidade de células é determinada pela tensão de barramento (WERTZ; LARSON, 1999). Os painéis solares possuem uma característica não linear entre suas grandezas físicas mais importantes, a tensão (V) e a corrente (I), como pode ser observada na Figura 3.2. Além disso, fatores como a temperatura e nível de iluminação, influenciam a característica da curva, dessa forma, a temperatura tende a deslocar a curva para esquerda, enquanto que o aumento do ângulo de incidência dos raios solares, distância do painel ao sol e radiação, tende a achatar a curva (FORTESCUE et al., 2004; MAGALHÃES, 2005).

Também na Figura 3.2, pode-se observar a potência (P) fornecida pelo painel solar. O ponto máximo da curva "V x P" representa a máxima potência produzida pelos painéis solares.

Figura 3.2 - Curvas características do Painel Solar.



A curva em azul representa a relação tensão (V) e corrente (I) dos painéis solares, enquanto que a curva em vermelho representa a potência (P) fornecida pelos painéis solares.

Fonte: Adaptada de Fortescue et al. (2004) e Magalhães (2005).

3.1.2 Bateria

A bateria, além de fornecer energia durante o período que não há radiação solar, ou seja, no eclipse, pode fornecer energia durante os períodos em que o satélite está iluminado, de forma a auxiliar os painéis solares a atender a demanda de energia solicitada pelas cargas do satélite. (FORTESCUE et al., 2004)

As baterias podem ser primárias, aquelas que não podem ser recarregadas, ou secundárias, as que permitem serem recarregadas. O número de de ciclos de carga e descarga da bateria é determinado pelos parâmetros orbitais e, principalmente, pela altitude do satélite, alternando os períodos de Sol e sombra. (WERTZ; LARSON, 1999)

No projeto de um EPS deve-se levar em consideração os seguintes aspectos da bateria:

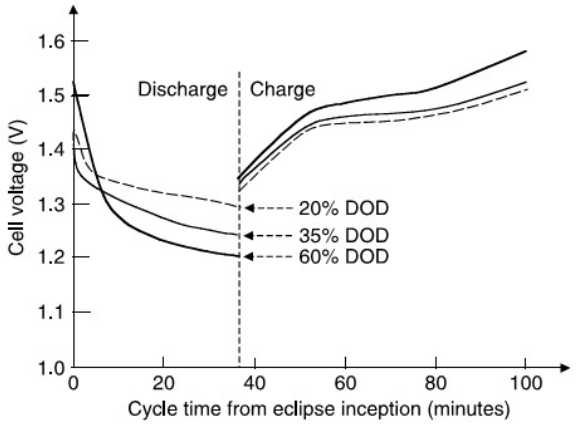
- Físico: tamanho, peso, configuração, ambiente dinâmico e estático.
- Elétrico: tensão, corrente de carregamento, ciclo de trabalho, tempo de armazenamento, profundidade de descarga.
- Programático: custo, vida útil, missão, confiabilidade, manutenibilidade, produtibilidade.

Um fator importante, no que diz respeito à vida útil das baterias é a profundidade de descarga (*Depth-of-discharge* - DOD) (WERTZ; LARSON, 1999). Este parâmetro quantifica a porcentagem da capacidade total da bateria que é perdida durante o período de descarga, ou seja, é a relação entre a quantidade de carga retirada da bateria durante o tempo de descarga e a capacidade de carga total da bateria quando esta se encontra totalmente carregada (WERTZ; LARSON, 1999; FREIRE, 2009).

Segundo Fortescue et al. (2004) os tipos mais comuns de baterias utilizadas em satélites convencionais são: (i) LEO: zinco-prata (*silver-zync* - AgZn) e níquel-cádmio (*nickel-cadmium* - NiCd); (ii) GEO: níquel-hidrogênio (*nickel-hydrogen* - NiH_2).

A Figura 3.3 apresenta a curva de carga e descarga de uma bateria de um satélite de órbita baixa.

Figura 3.3 - Ciclos de Carga e Descarga de uma Bateria.



A figura apresenta ciclos de carga e descarga de uma bateria.
Fonte: Patel (2004).

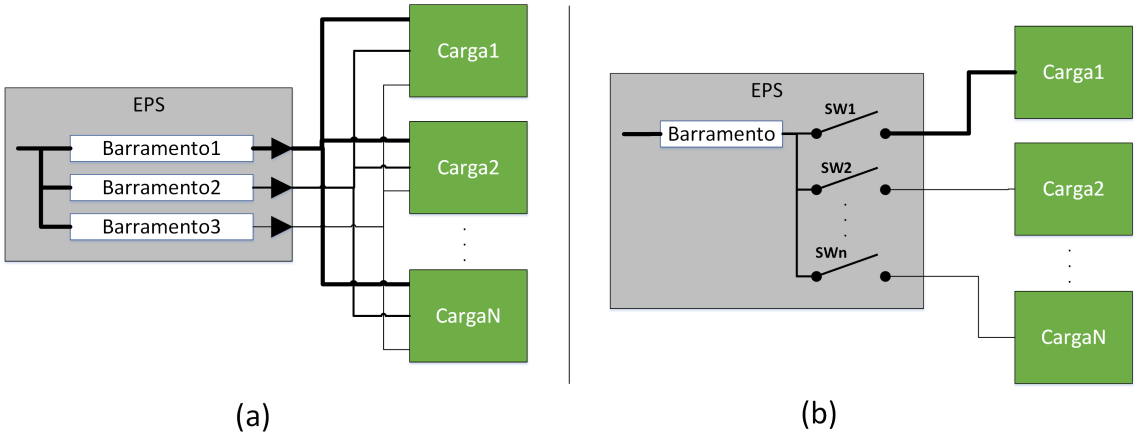
3.1.3 Gerenciamento, Controle e Distribuição de Energia

De acordo com Wertz e Larson (1999), a unidade de distribuição de energia (*Power Distribution Unit* - PDU) em um satélite consiste basicamente de: cabeamento, proteção contra falhas e comutadores para ligar e desligar as cargas do satélite.

A distribuição pode ser centralizada ou distribuída, dependendo da posição dos conversores. Na arquitetura distribuída, cada subsistema é responsável por regular sua carga (BURT, 2011a). Já na arquitetura centralizada, a energia para as cargas do satélite é regulada no

barramento principal. A Figura 3.4 ilustra os tipos de distribuição.

Figura 3.4 - Arquitetura EPS.



A figura apresenta dois tipos de arquitetura para EPS, em relação à unidade de distribuição de energia: (a) arquitetura centralizada e (b) arquitetura distribuída.

Fonte: Adaptada de Burt (2011b).

Há duas técnicas para controlar a energia gerada. A primeira delas é por transferência direta de potência (*direct energy transfer* - DET), na qual a energia flui dos painéis solares para as cargas, sem nenhum elemento intermediário conectado em série. A segunda é pelo rastreamento de potência (*peack-power tracker* - PPT), em que há um bloco regulador entre os painéis solares e as cargas, permitindo que o ponto de potência máximo do gerador solar possa ser rastreado. (MAGALHÃES, 2005)

A unidade de condicionamento de potência (PCU) depende essencialmente da topologia de barramento regulado ou barramento não regulado (FREIRE, 2009; TORRES, 2014). A PCU pode conter, pelo menos:

- *Battery Charge Regulator* (BCR): responsável por fornecer uma fonte constante de corrente para a carga da bateria com proteção e monitoramento adequado.

- *Battery Discharge Regulator* (BDR): converte a tensão da bateria na tensão do barramento com uma tensão regulada apropriada às cargas. Responsável pelo controle de descarga da bateria, tendo a função de manter o barramento dentro do range fixo de tensão especificado.

- *Shunt*: mantém a tensão do barramento principal do satélite num valor fixo. Sua função é controlar diretamente e com confiabilidade a potência disponibilizada pelo painel solar. Sendo assim, a demanda de energia do satélite deve ser suprida e o excesso de energia dissipado em forma de calor.

O barramento não regulado simplifica o subsistema de energia. Assim, o EPS disponibiliza no barramento a potência dentro de uma faixa variável de tensão. O barramento quase regulado, executa a regulação de tensão do barramento somente durante a carga da bateria. Já no barramento regulado o EPS disponibiliza uma tensão fixa no barramento e faz-se necessário o uso de reguladores para controlar a carga e descarga da bateria (WERTZ; LARSON, 1999).

Os dispositivos para controle de carga (BCR) e descarga (BDR) da bateria visam asseguram a operação da bateria dentro de uma faixa especificada, de modo a garantir que a vida útil esperada seja alcançada. (FILHO, 1984)

Em alguns casos, a PDU e a PCU são vistas como um único sistema, chamado de PCDU (*Power Control and Distribuition Unit*), a qual gerencia e monitora o comportamento do EPS (FORTESCUE et al., 2004).

3.2 Subsistema de Energia em Satélites Miniaturizados

Esta seção apresenta uma revisão bibliográfica para elucidar os aspectos importantes que devem ser levados em consideração durante sua concepção e, posteriormente, durante a de verificação de projeto de EPSs de satélites miniaturizados.

De acordo com Burt (2011a) há diversas soluções para o EPS de pico e nanossatélites, tais como, as soluções de prateleira (COTS[1], do inglês *Commercial off-The-Shelf*), e soluções únicas, concebidas diretamente para um projeto

O projeto ideal de um EPS é aquele que atende aos requisitos de uma missão específica, e que pode ser utilizado em outras missões, sem a necessidade de ser completamente redesenhado para cada nova missão. As arquiteturas distribuídas permitem uma maior flexibilidade no cumprimento dos requisitos de diferentes cargas úteis, dessa forma, elas possibilitam realizar um projeto modular, que garante uma maior reutilização. (BURT, 2011a; BURT, 2011b)

Os painéis solares e as baterias devem ser os primeiros componentes a serem analisados no projeto do EPS, uma vez que, eles determinam a arquitetura do subsistema (THIRION, 2009). Em relação aos painéis solares dos satélites ESMO e ESTCube (ULRICH et al., 2009; PAJUSALU et al., 2014) a configuração da conexão das células solares foi de suma

[1]Itens disponíveis para sempre comprados no mercado.

importância para atingir a tensão de barramento necessária para alimentação do satélite e carregamento da bateria.

A pesquisa realizada demonstrou que a grande maioria dos pico e nanossatélites utiliza baterias de Li-Ion, sendo alguns exemplos: ESMO (ULRICH et al., 2009), NanoSatC-BR1 (PIOVESAN et al., 2014), Tancredo 1 (TIKAMI et al., 2014), ESTCube (PAJUSALU et al., 2014) e AESP-14 (COSTA, 2015); uma vez que esse tipo de bateria possui boa relação de energia específica (Wh/kg) (WERTZ; LARSON, 1999; FORTESCUE et al., 2004; HOFFMANN, 2015). Enquanto as baterias de NiCd têm 39 Wh/kg, as baterias de Li-ion apresentam energia específica de 80 Wh/kg (FORTESCUE et al., 2004).

Em geral, para proteção das cargas contra curto-circuitos e sobre-corrente, são utilizados comutadores ESMO e limitadores de corrente (ULRICH et al., 2009; THIRION, 2009), Zambrano et al. (2013) e Costa (2015) citam o uso de sensores de corrente e tensão para monitoramento dessas grandezas no AESP-14 e uma chave eletrônica para proteção.

Outro aspecto importante que diz respeito a PCDU dos satélites miniaturizados, é que eles podem adotar uma unidade de rastreio de máxima potência e uma unidade para processar os comandos do OBDH e também, enviar algumas medidas. (ULRICH et al., 2009; PIOVESAN et al., 2014; PAJUSALU et al., 2014; THIRION, 2009)

O subsistema de energia elétrica dos satélites pesquisados adotam, majoritariamente, a configuração de barramento não regulado, como no caso do ESMO ESTCube-1 e OUFTI-1. (ULRICH et al., 2009; PAJUSALU et al., 2014; THIRION, 2009).

3.2.1 Falhas em EPS

Uma falha no EPS pode representar a perda do satélite e, consequentemente, a perda da missão (FORTESCUE et al., 2004). Na Tabela 3.1 são apresentados alguns casos de satélites miniaturizados que apresentaram falhas no subsistema de energia.

Tabela 3.1 - Falhas no EPS de Satélites Miniaturizados.

Satélite	Ano	Local da Falha	Descrição da Falha	Consequência
AAUsat	2003	Bateria	Perda na capacidade da bateria de armazenar carga, conduzindo o sistema a entrar em modo de contingência com frequência, o que impedia o EPS de fornecer energia para o OBC.	A situação degrada da bateria impediu o estabelecimento de comunicação. (ALMINDE et al., 2001)
Express	2005	Bateria	O problema foi causado por uma falha no subsistema de energia, impendido que as baterias pudessem ser recarregadas e, dessa forma, resultando o desligamento do satélite.	Perda da missão. (ULRICH et al., 2009; ALMINDE et al., 2005)
RAX	2010	Painel Solar	Uma degradação ocorrida nos painéis solares proveniente do curto-circuito entre as células solares casou interrupção na produção de energia.	Perda da missão. (CUTLER et al., 2011)
NanoSatc-BR1	2014	Bateria	As baterias não retém mais carga.	Satélite operando somente durante o período que recebe energia solar. (NANOSATC-BR, 2014)

Fonte: Produção do autor.

3.2.2 Atividades de Verificação de Pico e Nanossatélites

Para Patel (2004), durante todo o projeto do EPS, o engenheiro deve focar nos requisitos que o subsistema deve atender, não somente no ambiente operacional, mas também, na plataforma de lançamento e durante as órbitas iniciais. Dessa forma, é necessário verificar o subsistema para garantir que ele atenda todas as especificações. Nesta seção são apresentados alguns trabalhos relacionados às atividades de verificação de um EPS de pico e nanossatélites.

O trabalho de Bürger (2014) apresenta alguns testes elétricos funcionais realizados durante a etapa de AIT. Dentre as atividades de integração elétrica e testes funcionais realizadas antes e após os testes ambientais, destacam-se:

- *Check* funcional utilizado para verificação de comunicação entre os subsistemas e verificação funcional durante cada teste ambiental;
- Teste breve para verificação do estado do *CubeSat*, através do envio de telemetrias e o recebimento de telecomandos. O envio e o recebimento destes dados é feito por meio de cabos;
- Teste de simulação de voo. Verificação de todas as funções e desempenho para todos os modos de operação do *CubeSat*, via RF.

Ainda no contexto do AIT, Hoffmann (2015) propõe uma arquitetura para efetuar **testes operacionais** nos modos eletroeletrônicos presentes em um *CubeSat*.

Bizarria et al. (2014) propõe um modelo para **simular a energia produzida pelos painéis solares**, afim de executar os testes de carga e descarga das baterias de um *CubeSat*. O sistema de teste é composto por:

- Computador: que realiza a interface entre o usuário e a fonte de tensão. Por este computador são realizados ajustes nos parâmetros da simulação.
- Fonte de tensão: recebe os sinais do computador, de modo a definir um intervalo com níveis específicos de tensão, que é uma função característica do fornecimento de corrente elétrica das células fotovoltaicas.
- Elemento em teste.

No desenvolvimento do EPS do *CubeSat* OUFTI-1, Thirion (2009) cita que foram utilizados três modelos, descritos a seguir:

- Modelo protótipo em *protoboard*[2]: permitiu conhecer o comportamento dos componentes eletrônicos e testá-los de forma independente. Diversos protótipos foram feitos durante o desenvolvimento, como por exemplo, o sistema de dissipação e os conversores DC/DC;
- Modelo de engenharia: correspondeu ao modelo de voo e foi utilizado para realização de testes. Este modelo incluiu mais pinos para coleta de medições, em relação ao modelo de voo;

[2]Placa que oferece diversos pontos para conexão rápida de componentes eletrônicos.

- Modelo de voo: modelo que foi enviado ao espaço.

O artigo de Pajusalu et al. (2014) apresenta os testes realizados no EPS do *CubeSat* ESTCube e conta que todos os componentes foram testados antes da integração. Os testes da bateria incluíram ciclagem de carga e descarga e testes de termo-vácuo. Além disso, foram realizados testes funcionais nos reguladores de tensão, que fornecem energia às linhas de tensão reguladas, em condições normais e de pior caso. O teste completo do EPS foi realizado em um modelo *protoflight*, que é fisicamente igual ao modelo de voo (PAJUSALU et al., 2014; SLAVINSKIS et al., 2015). Um modelo de engenharia foi produzido para realização de testes funcionais, e também, para permitir o desenvolvimento do *software* de bordo, independente do cronograma de teste do modelo *protoflight*. O ESTCube, lançado em maio de 2013, após 2 anos em operação, encerrou suas atividades após degradação dos painéis solares[3].

Corpino e Stesina (2014) apresentam o método de verificação dos requisitos funcionais do *CubeSat* E-st@r, no qual utilizam simulação com *hardware-in-the-loop* (HIL), similarmente a proposta do framework. Dessa forma, os resultados se aproximam ainda mais do subsistema real, em ambiente operacional.

Todavia, Corpino e Stesina (2014) executam as simulações e coletam os resultados para, mais tarde, verificarem se os requisitos foram atendidos. No *framework* proposto nesta Dissertação, os testes para verificação funcional dos requisitos e validação dos modelos são executados automaticamente, durante a simulação.

O processo de verificação do CubeSat E-st@r se deu de acordo com as normas da ECSS (ECSS, 2009a; ECSS, 2010b; ECSS, 2012b), que contemplam a verificação ambiental e funcional, em diferentes níveis e em diferentes fases do projeto. Utilizando simulador HIL na fase de qualificação as seguintes funções de um *CubeSat* são verificadas: (i) transmissão e recepção de telemetrias e telecomandos; (ii) determinação e controle de atitude; e, (iii) carga e descarga das baterias. Além disso, os requisitos operacionais: (i) troca dos modos de operação do *CubeSat*, a partir de comandos de estação terrena; (ii) execução da sequência de ativação; e (iii) desdobramento da antena, podem ser demonstrados. (CORPINO; STESINA, 2014)

Em resumo Corpino e Stesina (2014) concluem que os testes ambientais realizados em *CubeSats* visam para demonstrar que os mesmos têm capacidade de sobreviver à fase de lançamento, sem comprometer o veículo lançador e outras cargas, que não há um esforço grande em relação à verificação dos requisitos funcionais e operacionais e não há diretrizes para estes tipos de testes, cabendo ao desenvolvedor a decisão de realizá-los ou não.

[3]Disponível em: `http://www.eas.ee/kosmos/en/estonian-space-office/news/article/389-estcube-1-ceased-working`. Acesso em: 20 nov. de 2015

A necessidade de aumentar a confiabilidade dos pico e nanossatélites é evidenciada no esforço para o estabelecimento de uma norma ISO (*International Organization for Standardization*) para verificação funcional de pico e nanossatélites, sem desprezar a natureza de baixo custo e produção rápida (CORPINO; STESINA, 2014), bem como na existência da norma ISO (ISO/CD 19683) que padroniza os testes ambientais para nanossatélites. (CHO et al., 2012)

3.3 Simulação do Subsistema de Energia Elétrica

Na concepção do EPS de um satélite podem ser adotadas diversas configurações. Dessa forma a simulação deste subsistema pode auxiliar na escolha da melhor configuração, na análise do balanço de potência, e também, na sua verificação funcional.

Um dos primeiros trabalhos sobre simulação do subsistema de energia de um satélite é o artigo de Bauer (1969), no qual é descrito um *software* de simulação computacional para análises elétricas e dos transientes de temperatura, já que as análises do desempenho de um EPS é uma tarefa complexa, pois envolve a interação com outros subsistemas, incluindo o subsistema de controle térmico, uma vez que a bateria dissipa calor de forma imprevisível.

No trabalho de Bauer (1969), as características de cada componente são representadas através de curvas "V x I", representadas no simulador através de tabelas. A curva do painel solar leva em consideração as informações da órbita, tempo, enquanto a curva da bateria leva em consideração características de profundidade de descarga e temperatura.

Filho (1984) descreve um simulador de desempenho de um EPS de um satélite de órbita baixa, pequeno e de baixa potência, desenvolvido em linguagem de programação *Fortran*. Ele destaca que os cálculos para o projeto de um EPS são complexos, uma vez que, seus componentes possuem características não-lineares. Por exemplo, as características de tensão e corrente dos painéis solares dependem do ângulo de iluminação e temperatura, enquanto na bateria, estas características dependem basicamente do estado de carga, corrente de carga e descarga e temperatura. Os modelos de referência utilizados na simulação incluem gerador solar, regulador *shunt*, baterias recarregáveis, dispositivos para controle de carga e descarga e elementos consumidores (cargas) (FILHO, 1984).

No artigo de Colombo et al. (1997) é apresentado um simulador de EPS para dimensionamento preliminar do subsistema, sendo que este simulador não leva em consideração características de massa e temperatura da bateria, desenvolvido em MATLAB/*Simulink*. Este simulador permite a análise de diversas arquiteturas de EPSs, em particular permite variar o número e a orientação dos painéis solares, atitude do satélite, regulagem da tensão de barramento (não regulado, semi regulado ou regulado) e o número de baterias. O comportamento do EPS pode ser simulado de duas maneiras:

- Balanço de energia - monitorar os principais parâmetros do subsistema: tensão e corrente de saída dos painéis solares, tensão e corrente do barramento, corrente, estado de carga e tensão nos terminais da bateria, além de verificar se a demanda de energia é suficientemente atendida.

- Qualidade da tensão - analisar o comportamento do transiente dos dispositivos eletrônicos de regulagem e controle da energia.

O diagrama de blocos completo deste simulador pode ser observado na Figura 3.5

Figura 3.5 - Diagrama de Blocos do Simulador para EPS.

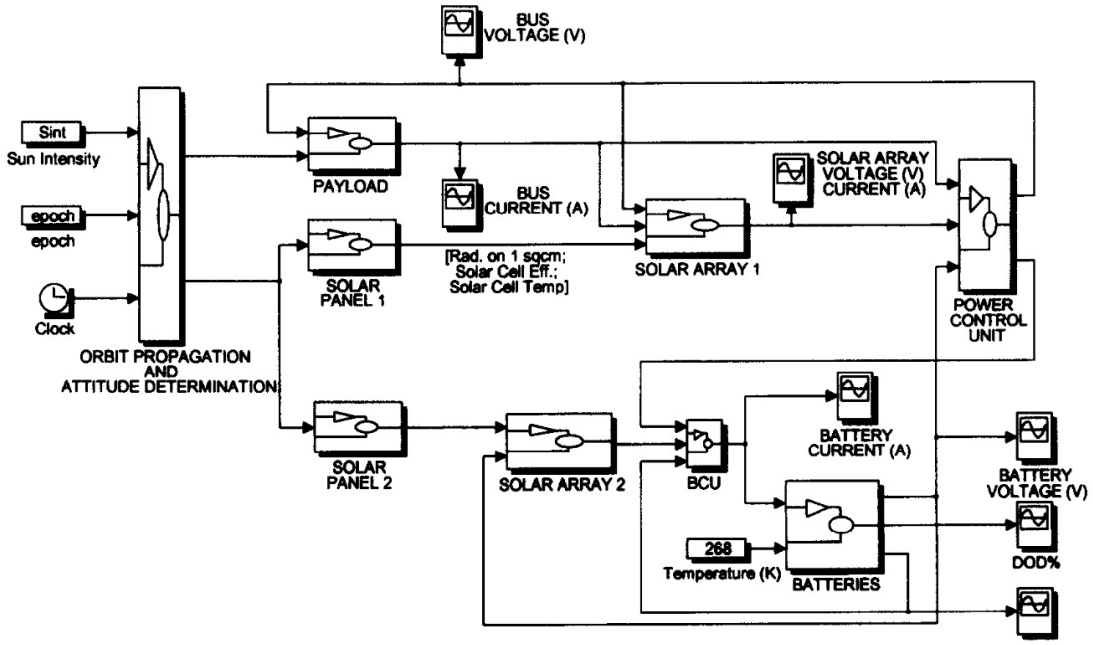


Diagrama de blocos de um simulador de EPS, com barramento de tensão não regulado.

Fonte: Colombo et al. (1997).

Jiang et al. (2003) apresentam uma bancada de teste virtual (VTB) que pode ser utilizada para projeto e testes do subsistema de potência de um satélite, desenvolvida em Simulink e ACSL (*Advanced Circuit Simulation Language*).

Zahran (2006) propõe o desenvolvimento de ferramentas para modelagem e simulação do EPS de um satélite, para análise do mesmo durante sua operação. Este simulador, deno-

minado SEPSMS (*Satellite Electrical Power Subsystem Modelling and Simulation*), leva em consideração os parâmetros orbitais (inclinação, ascensão reta do nodo ascendente e argumento do perigeu), iluminação dos painéis solares, cargas do satélite e baterias. Uma interface gráfica disponibilizada pelo MATLAB e mostra os resultados da simulação em três gráficos e um em painel. Nos primeiros gráficos concentram-se as informações relacionas à órbita do satélite, no segundo as informações relacionadas aos painéis solares (corrente de curto-circuito, corrente das cargas e capacidade da bateria em Ah) e no terceiro as informações de carga e descarga da bateria. No painel, podem-se realizar ajustes nos parâmetros orbitais, na perturbação da órbita, e também, no passo de simulação. (ZAHRAN, 2006)

Farid et al. (2008) apresenta um simulador do EPS de um satélite de sensoriamento remoto, de órbita baixa, utilizando o *Simulink*. O modelo matemático do EPS levou em consideração: as células solares e as células da bateria, incluindo a profundidade de descarga. O comportamento do EPS é guiado pela incidência de Sol. Se o satélite estiver em eclipse as baterias fornecem energia para o satélite. A bateria interrompe o fornecimento de energia em qualquer uma das três situações: (i) DOD é maior que $0,25$; (ii) a tensão da bateria é igual ao mínimo da tensão de descarga; ou (iii) um comando é enviado da estação terrena para interromper a descarga. Nestas situações as cargas menos significativas do satélite são desligadas, exceto as cargas críticas (EPS, AOCS, OBDH) que nunca são desligadas. Caso o satélite esteja iluminado pelo Sol três modos de operação são possíveis: (i) os painéis solares fornecem energia para o satélite enquanto carrega as baterias; (ii) tanto os painéis solares, quanto as baterias fornecem energia para o satélite; (iii) os painéis solares só fornecem energia às cargas do satélite, uma vez que, a bateria esteja totalmente carregada. (FARID et al., 2008)

O artigo de Kaya e Bayrakceken (2011) apresenta a modelagem e simulação de um EPS e mostra que a simulação de um modelo preciso do sistema real provê ajuda durante a etapa de projeto, no que diz respeito às mudanças ambientais e internas.

Corpino e Stesina (2014) realizaram uma simulação com HIL de um *CubeSat*, levando em consideração vários subsistemas, incluindo o EPS com painéis solares e sensores de temperatura. A simulação disponibiliza informações, tais como, posição do sol, atitude, condições térmicas, e as tensões e correntes de cada painel, as quais são transmitidas à PCDU física durante a simulação.

Dentre os simuladores de EPS estudados, observou-se que as ferramentas usadas na maioria dos trabalhos são: MATLAB/*Simulink*, *Scilab*, *AMESim*, *Modelica*, *Multisim* e *PSpice*.

O MATALB contém uma linguagem de alto nível e um ambiente interativo para desenvolvimento de projetos e resolução de problemas, em diversas áreas, tais como, sistemas

de controle, processamento de sinais e imagens e comunicação. Na suíte do MATLAB encontra-se o *Simulink*, que possibilita a modelagem gráfica em diagrama de blocos, o qual também permite gerar código automaticamente e embarcar num *target*[4], através do *Simulink Coder*. (MATHWORKS, 2015a; MATHWORKS, 2015b; WIKIPÉDIA, 2015)

Uma solução *open-source* ao MATLAB é o *Scilab*, que possui interface gráfica similar ao MATLAB e também permite a modelagem por diagrama de blocos. (MARTIN, 2009; LAMY, 2012; SCILAB, 2015)

Quando for importante modelar fluxos físicos as ferramentas AMESim e Modelica permitem representar sistemas físicos, através da notação de gráficos de ligação (PULECCHI et al., 2006; OLIVA, 2012; MODELICA, 2015). Enquanto que para realizar as análises de aspectos relacionados aos circuitos eletroeletrônicos as ferramentas *MultiSim* (NATIONAL INSTRUMENTS, 2015) e *PSpice* (OrCAD, 2015) são indicadas.

[4] *Hardware* específico que possui interface com a ferramenta.

4 *FRAMEWORK* PARA VALIDAÇÃO AUTOMÁTICA DE MODELOS

Neste capítulo é apresentado o *framework* para validação de modelos e execução automática de testes funcionais de um subsistema de pico e nanossatélites. Os elementos genéricos do *framework* são descritos neste capítulo, enquanto que o detalhamento do mesmo aplicado ao subsistema EPS é descrito no Capítulo 5.

O papel fundamental do *framework* e permitir que um usuário realize a verificação funcional do subsistema de energia elétrica de um pico ou nanossatélite, bem como a validação dos modelos virtuais e prototipados que representam este subsistema, já nas fases iniciais de desenvolvimento. O subsistema é inicialmente representado por modelos virtuais ou físicos. A entrada para o *framework* são os requisitos e a saída é um veredicto de teste. Uma visão geral do *framework* é apresentada na Figura 4.1, onde as atividades são representadas nos retângulos e os artefatos nas elipses.

O *framework* cobre as seguintes atividades: (i) criar a Matriz de Testes; (ii) criar o Plano de Verificação; (iii) criar os modelos que descrevem o funcionamento do subsistema a ser validado; (ii) criar a Matriz de Testes; (iii) criar o Plano de Verificação; (iv) definir a configuração de execução; (v) criar a Matriz de Sequência; (vi) criar a(s) função(ões) de teste; (vi) executar os testes; e (vii) analisar os resultados dos testes.

Um conjunto de artefatos é definido e usado no *framework*. O documento de requisito é considerado o principal artefato de entrada para o *framework*, e partir dele é possível realizar as atividades e gerar outros artefatos, os quais são descritos ao longo deste capítulo.

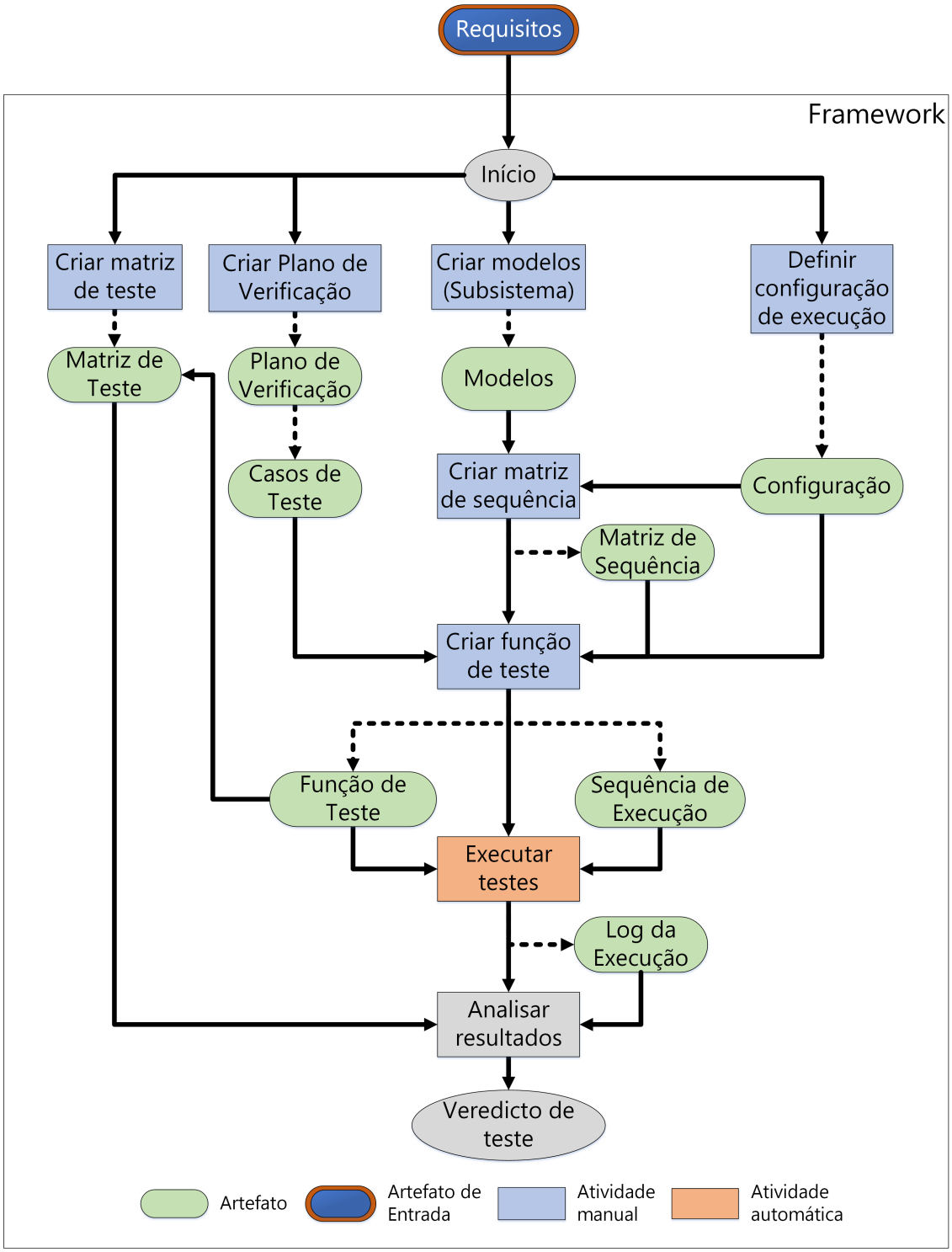
Diferentes profissionais estão envolvidos com o *framework*, seja na elaboração de artefatos, seja na execução de atividades. Os principais profissionais são:

- Engenheiro de Sistema ou Requisitos: é responsável pela elaboração dos requisitos;

- Engenheiro Especialista: responsável pela criação dos modelos do subsistema a ser validado e da Matriz de Sequência;

- Equipe de Verificação de Validação: responsável pela elaboração da Matriz de Teste e do Plano de Verificação, pela definição da sequência de execução, pela criação da Função de Teste e pelo veredicto de teste.

Figura 4.1 - Atividades e artefatos do *Framework*.



A relação entre as atividades do framework pode ser observada na figura.

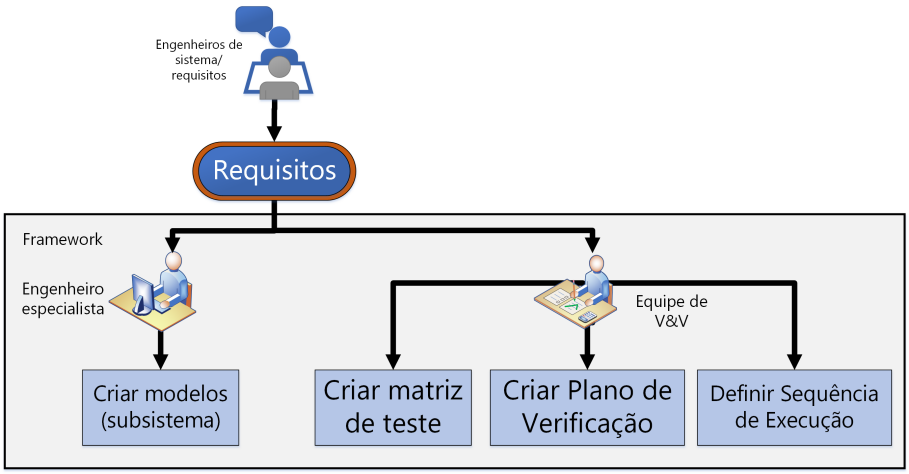
Fonte: Produção do autor.

4.1 Preparação Inicial

A entrada do *framework* consta dos requisitos do subsistema. Os requisitos são todas as considerações que determinam o que deve ser realizado, quão bem devem ser realizados e sob quais condições (LOUREIRO, 1999). Os requisitos são preparados pelos engenheiros de sistema com conhecimento em elaboração de requisitos, a partir da necessidade dos *stakeholders*. (NASA, 2007)

Uma vez que os requisitos do subsistema são estabelecidos, é possível criar os modelos que descrevem o subsistema, criar a Matriz de Teste, criar o Plano de Verificação, e finalmente, definir a sequência de execução, conforme ilustra a Figura 4.2. Algumas atividades podem ocorrer paralelamente, como Criar Modelos e Criar o PLano de Verificação, sempre que não há dependência direta entre elas.

Figura 4.2 - Preparação Inicial do *Framework*.



As atividades de preparação inicial do *framework* são disparadas a partir dos requisitos de entrada.

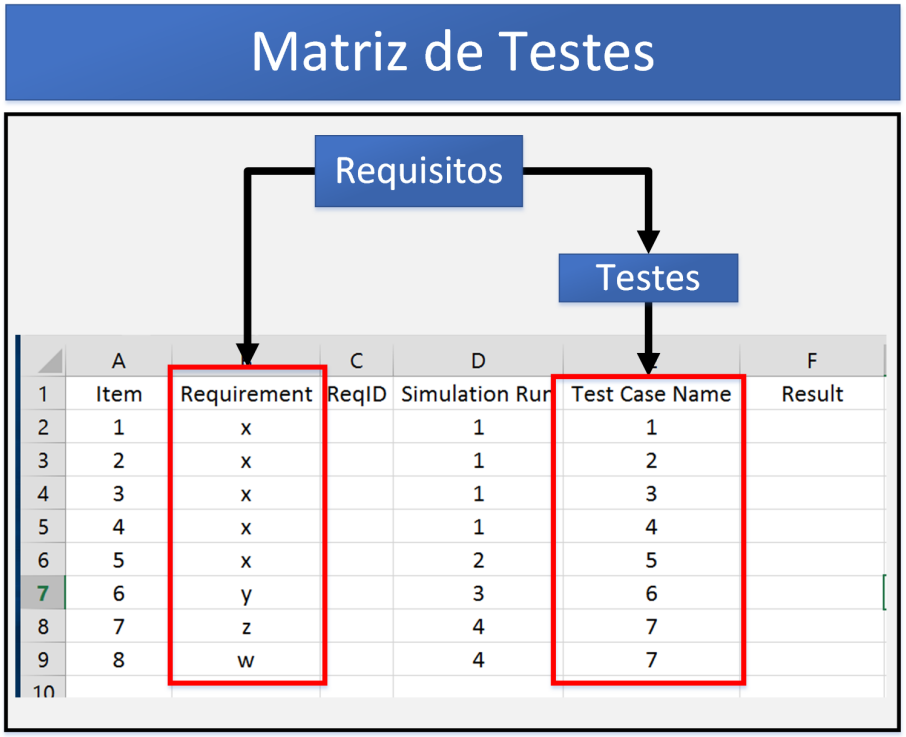
Fonte: Produção do autor.

4.1.1 Matriz de Teste

A Matriz de Teste, ilustrada na Figura 4.3, é o artefato que relaciona os requisitos e os casos de teste, de forma a permitir uma rastreabilidade desde os requisitos até os casos de teste gerados. A partir dos requisitos é possível identificar as saídas esperadas após a execução de cada caso de testes.

A Matriz de Teste deve ser elaborada pela equipe de verificação e validação e deve conter os itens descritos na Tabela 4.1 e seus relacionamentos.

Figura 4.3 - Matriz de Teste.



A figura apresenta o *layout* da Matriz de Teste adotado no *framework*.
Fonte: Produção do autor.

Tabela 4.1 - Descrição das Colunas da Matriz de Testes.

Item	**Descrição**
Item	Identifica a ordem dos requisitos.
Requirement (Requisito)	Lista os requisitos funcionais a serem verificados no *framework*.
ReqID	Identificação do requisito escrito na coluna "*Requirement*" e que pode estar associada a um ou mais casos de teste.
Simulation Run	Refere-se à simulação que executa determinado caso(s) de teste.

(Continua)

Tabela 4.1 – *Continuação.*

Item	Descrição
Test Case Name (Nome do Caso de Teste)	Nome da Função de Teste (derivada de um caso de teste), utilizada para aplicar o teste no subsistema, e que pode ser utilizada para verificar a um ou mais requisitos.
Result (Resultado)	Lista o resultado do teste. É preenchida automaticamente através da comparação entre a saída esperada (presente no *script* da Função de Teste, ver Seção 4.3), proveniente do requisito, e a saída obtida após a execução do teste. Os detalhes os resultados de uma rodada de simulação são apresentados na seção 4.4.

Produção do autor.

4.1.2 Plano de Verificação

O Plano de Verificação é o único artefato produzido no contexto do *framework* que é um documento.

Os requisitos também são entrada para elaboração do Plano de Verificação do subsistema, o qual deve ser elaborado pela Equipe de Verificação e Validação.

O Plano de Verificação deve conter os requisitos, uma descrição do subsistema, os casos de teste e os itens mencionados a seguir:

- (i) Método de verificação: o único método utilizado é o teste, executado automaticamente;
- (ii) Nível de Verificação: é possível realizar testes em nível de subsistema e componentes[1];
- (iii) Etapa de Verificação: indica que o *framework* é utilizado para as fases iniciais do ciclo de vida.
- (iv) Filosofia de Modelos: é possível executar os testes em modelos virtuais e modelos reais, dessa forma indica-se o modelo a ser utilizado no teste;
- (v) As ferramentas de verificação são: simulador para verificação funcional e placas para prototipação do subsistema a ser verificado.

No contexto do *framework*, os casos de teste são inicialmente gerados a partir dos requisitos e, mais tarde, convertidos em uma Função de Teste para que a verificação possa ser

[1]Componentes, neste caso faz referência aos equipamentos do subsistema.

automatizada o máximo possível. A Tabela 4.2 apresenta os itens que caracterizam um caso de teste no Plano de Verificação, com sua respectiva descrição.

Tabela 4.2 - Formato de um Caso de Teste.

Item	**Descrição**
Identificação do caso de teste	Identifica o caso de teste.
Descrição	Descreve o caso de teste, incluindo as saídas esperadas.
Configuração do Teste	Indica a configuração adotada para realizar a verificação.
Equipamento de *Front-end*	Dependendo da configuração, este campo indica a necessidade do equipamento de *front-end*.
Requisitos a serem verificados	Indica os requisitos a serem verificados pelo caso de teste descrito. Vale ressaltar que o mesmo caso de teste pode ser utilizado para mais de um requisito, e também que, para verificar um requisito, possa ser necessário mais de um caso de teste.
Sequência de Teste	Apresenta a sequência de teste e o que deve ser observado após a verificação.

Produção do autor.

Foi incluído no Apêndice C um Plano de Verificação didático, com objetivo de exemplificar o conteúdo do mesmo.

4.1.3 Modelo do Subsistema

O modelo do subsistema EPS deve ser fornecido pelo engenheiro especialista como um conjunto de modelos dos equipamentos. O modelo de cada equipamento do subsistema é entregue em separado (painel solar, bateria, PCDU), desta forma, o modelo do subsistema é considerado componentizado. A vantagem do modelo do subsistema estar componentizado, isto é, separado, é a facilidade de reutilizá-lo em outros projetos.

Os modelos são manipulados na linguagem de programação utilizada pelo *framework*. Preferencialmente, os modelos devem ser entregues à Equipe de Verificação e Validação na linguagem de programação considerada. Uma discussão sobre a linguagem de programação adotada no *framework* é dada no Capítulo 5.

Devido à componentização do modelo do EPS, todas as interfaces entre os componentes (ou modelos dos equipamentos) devem estar bem definidas, para que o modelo do subsistema seja "montado" a partir da Matriz de Sequência. Dessa forma, o funcionamento do subsistema, ou seja, a relação entre cada equipamento que compõe o subsistema e a interdependência entre os mesmos, de modo que tal sequência corresponda a ordem de

execução do subsistema estará definida na Matriz de Sequência (ver Seção 4.2).

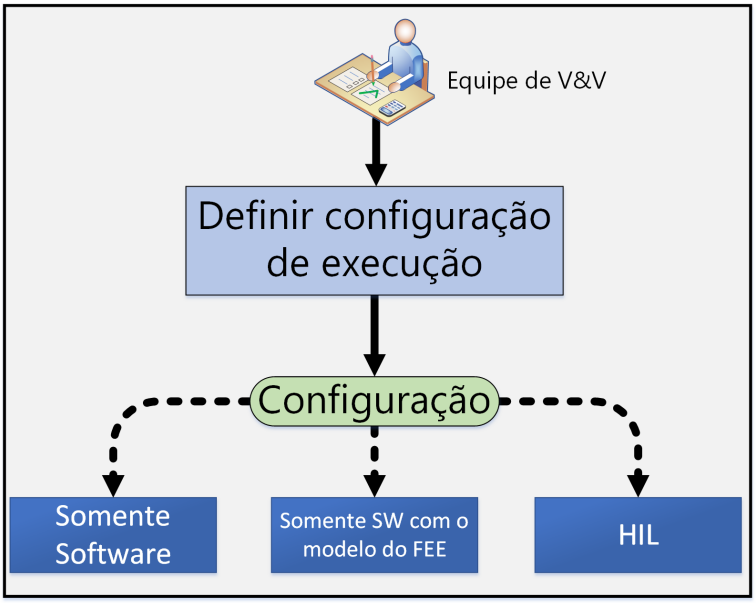
4.1.4 Configuração de Execução de Teste

A configuração diz respeito a verificação dos requisitos e validação dos modelos. Primeiramente considera-se somente os modelos dos equipamentos do subsistema em sua forma virtual (modelos virtuais) e, depois evolui-se até a inclusão de um *hardware* à malha de simulação (modelos híbridos).

O *framework* proposto nesta Dissertação acomoda três **configurações de teste**, nas quais se evolui da simulação realizada somente por *software* até a inclusão do *hardware*, sendo este último, um protótipo, ou ainda, um modelo mais representativo do modelo de voo, como por exemplo um modelo de engenharia do equipamento. A definição das Configurações do *framework* são baseadas na proposta de um simulador que segue os conceitos descritos pela ECSS (2010a), em particular o Simulador para Verificação Funcional (FVT).

As configurações devem ser definidas pela Equipe de Verificação e Validação, conforme ilustra a Figura 4.4. Assim, o subsistema pode ser verificado em uma ou em mais configurações, descritas a seguir.

Figura 4.4 - *Setup* de Execução.



Uma configuração pode ser considerada um *setup* de teste, de acordo com o jargão corrente de engenharia. As três configurações ou *setups* possíveis são: somente *software*, somente *software* com o modelo do FEE e *hardware-in-the-loop* (HIL).

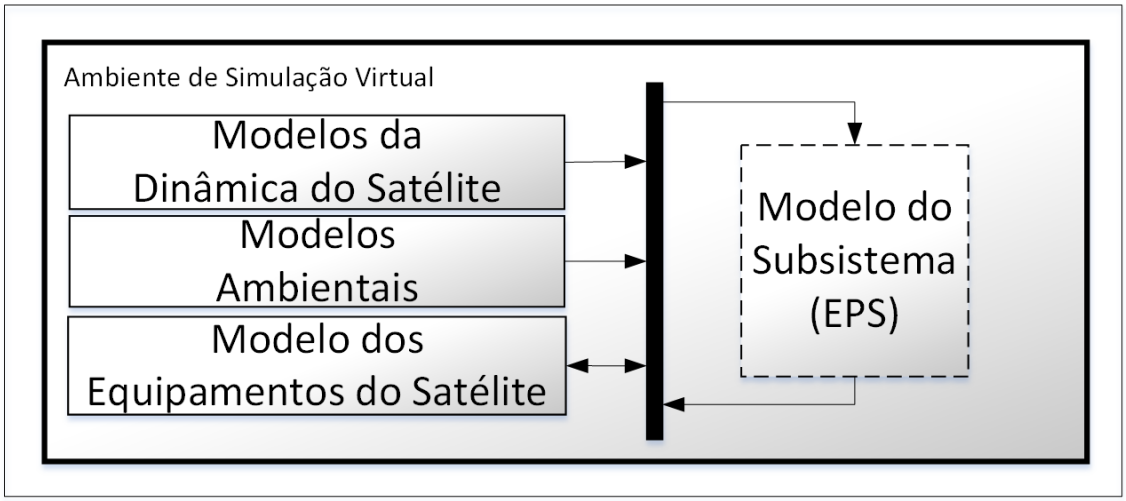
Fonte: Produção do autor.

4.1.4.1 Configuração - Somente *Software*

Nesta configuração realiza-se a verificação dos requisitos executando os testes funcionais nos modelos virtuais, durante a simulação do subsistema. Todas as informações transitam somente entre os modelos presentes no ambiente de simulação. Além dos modelos que compõem o subsistema, outros modelos como o Modelo de dinâmica Orbital, os Modelos Ambientais e os Modelos de Equipamentos do Satélite (relativos aos subsistemas que fazem interface com o modelo do EPS) devem fazer parte desta Configuração.

A Figura 4.5 ilustra a primeira configuração, que é executada somente em ambiente virtual.

Figura 4.5 - Configuração - Somente *Software*.



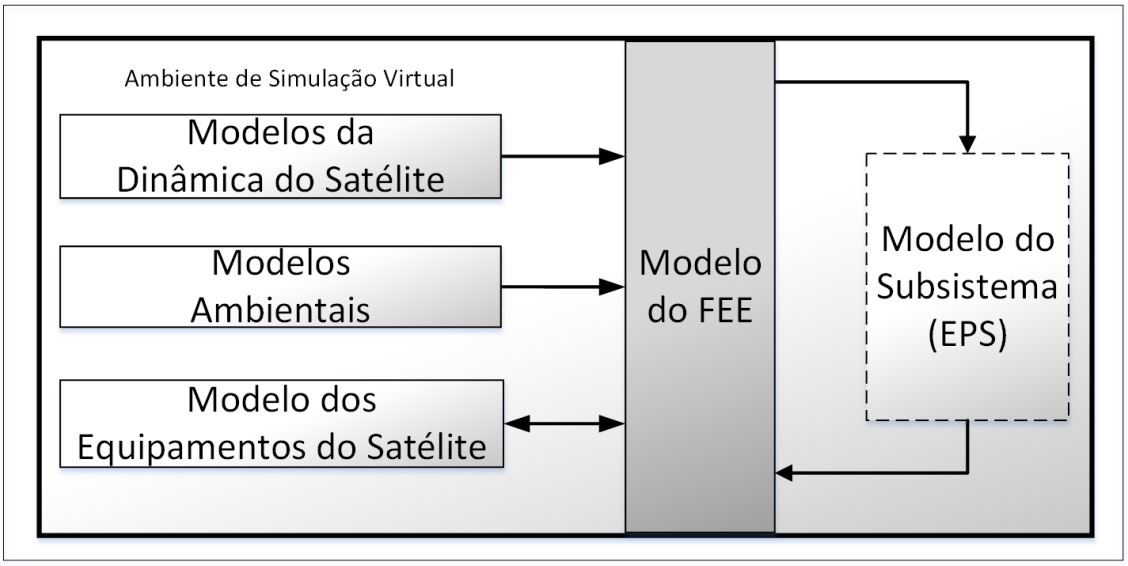
O modelo tracejado representa o subsistema em teste.

Fonte: Produção do autor.

4.1.4.2 Configuração - Somente *Software* com o Modelo do FEE

Nesta configuração, acrescenta-se aos modelos anteriores o modelo do *hardware* do equipamento de *front-end* (FEE), como pode ser observado na Figura 4.6. O FEE é o equipamento responsável por enviar as informações do ambiente virtual ao modelo físico do subsistema em teste. O objetivo desta Configuração é preparar e testar os requisitos de comunicação (protocolos e configuração elétrica) entre os modelos simulados e o modelo do subsistema, a partir do equipamento real de interface.

Figura 4.6 - Configuração - Somente *Software* com o Modelo do FEE.



O modelo tracejado representa o subsistema em teste.

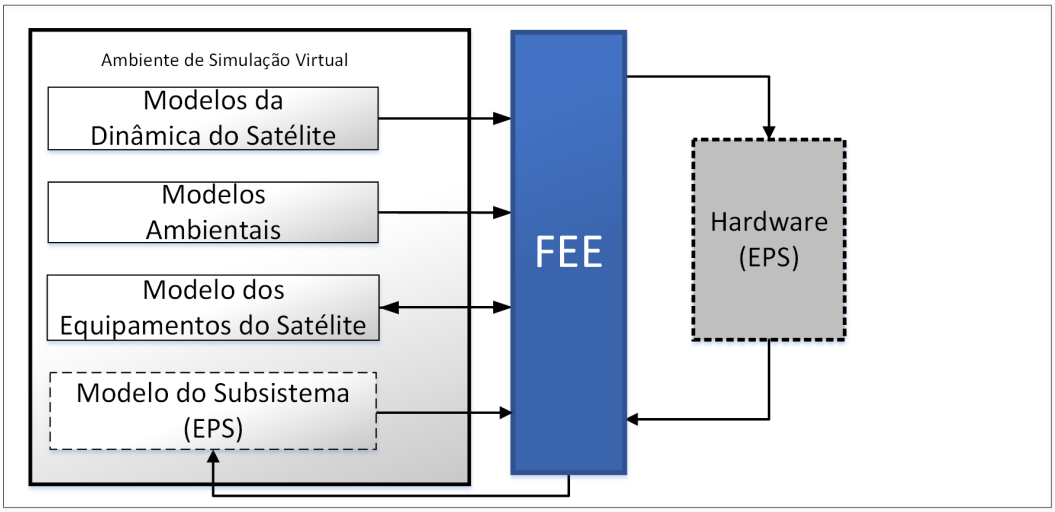
Fonte: Produção do autor.

4.1.4.3 Configuração - *Hardware-in-the-loop*

Finalmente, nesta configuração, substitui-se o modelo virtual do subsistema pelo modelo físico, para executar a verificação funcional. Observa-se na Figura 4.7 que o modelo do subsistema em teste é substituído gradualmente, dessa forma, alguns componentes são representados virtualmente, enquanto outros pelo *hardware*. Já na Figura 4.8 observa-se que o modelo virtual do subsistema em teste foi substituído integralmente pelo *hardware* que o representa. Vale ressaltar que, admite-se o como *hardware*, protótipos do subsistema e *hardwares* mais representativos, como, por exemplo, o modelo de engenharia, caso este modelo esteja disponível durante as fases iniciais do ciclo de desenvolvimento do satélite. Assim, é possível verificar as funções reais do subsistema em condições ambientais simuladas virtualmente. Esta configuração é caracterizada pelo uso de modelos híbridos.

O ambiente de simulação virtual se comunica com o *hardware* que representa o EPS a partir do equipamento de interface.

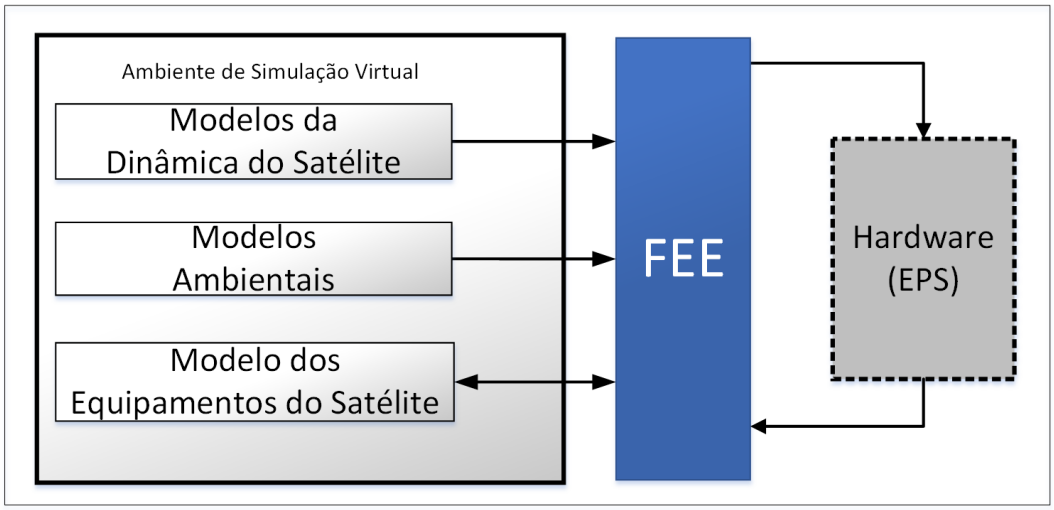
Figura 4.7 - Substituição Gradual dos Modelos Virtuais pelos Modelos Físicos.



A figura demonstra que o processo de substituição dos modelos virtuais pelos modelos físicos pode ser gradual.

Fonte: Produção do autor.

Figura 4.8 - Configuração - *Hardware-in-the-loop*.



A figura apresenta a combinação entre modelos virtuais e modelo físico. O modelo físico, tracejado, representa o subsistema em teste.

Fonte: Produção do autor.

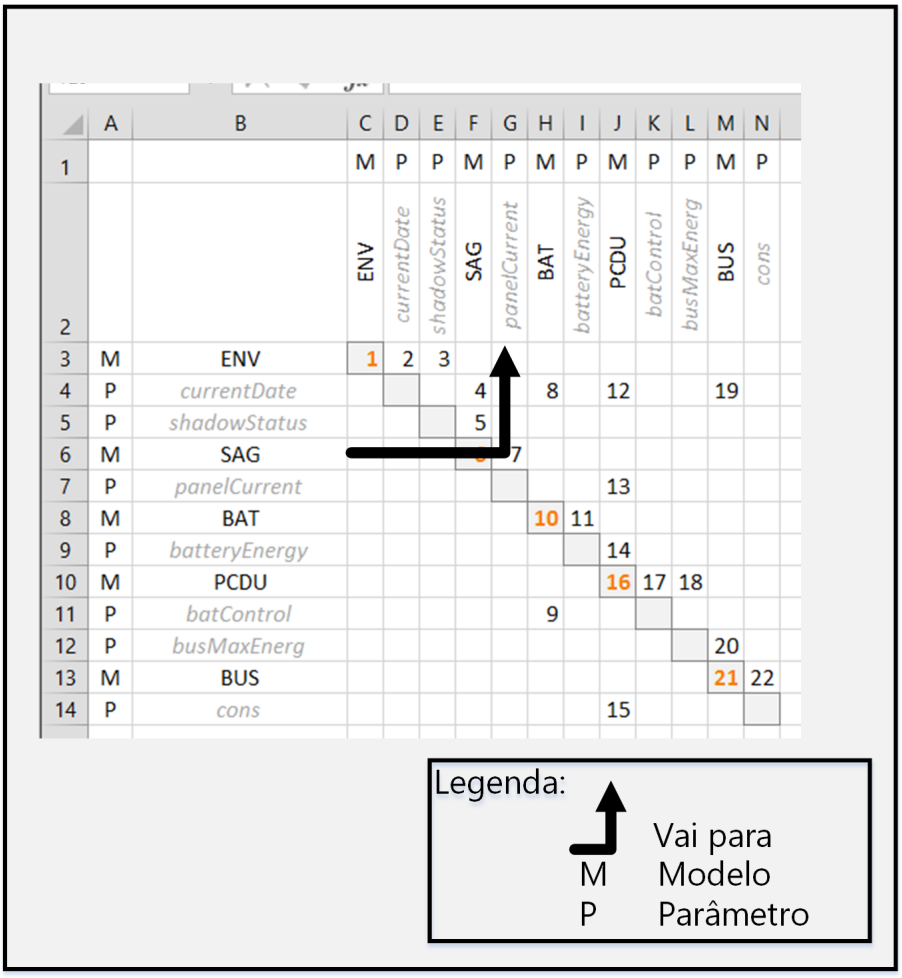
4.2 Matriz de Sequência

A Matriz de Sequência deve ser elaborada pelo Engenheiro Especialista para definir de forma bem estruturada a sequência ordenada de execução dos modelos.

Vale ressaltar que, conceitualmente, a Matriz de Sequência é semelhante à *Design Structure Matrix* (DSM), definida na Seção 2.4 do Capítulo 2.

A Matriz de Sequência relaciona a ordem das trocas de informações entre modelos (M) e seus parâmetros de interface (P). Nela, as linhas representam as saídas do modelo e as colunas, as entradas. Os numerais nas células indicam a ordem de execução, conforme ilustra a Figura 4.9. Quando houver um numeral na diagonal principal, onde há o encontro da linha e da coluna de um mesmo modelo ou parâmetro, significa que a execução ocorre dentro do modelo especificado. Os parâmetros de interface são as variáveis que transitam entre os modelos, ou seja, o modelo de um determinado componente realiza algum cálculo. Esta informação é necessária para que o próximo modelo execute sua função.

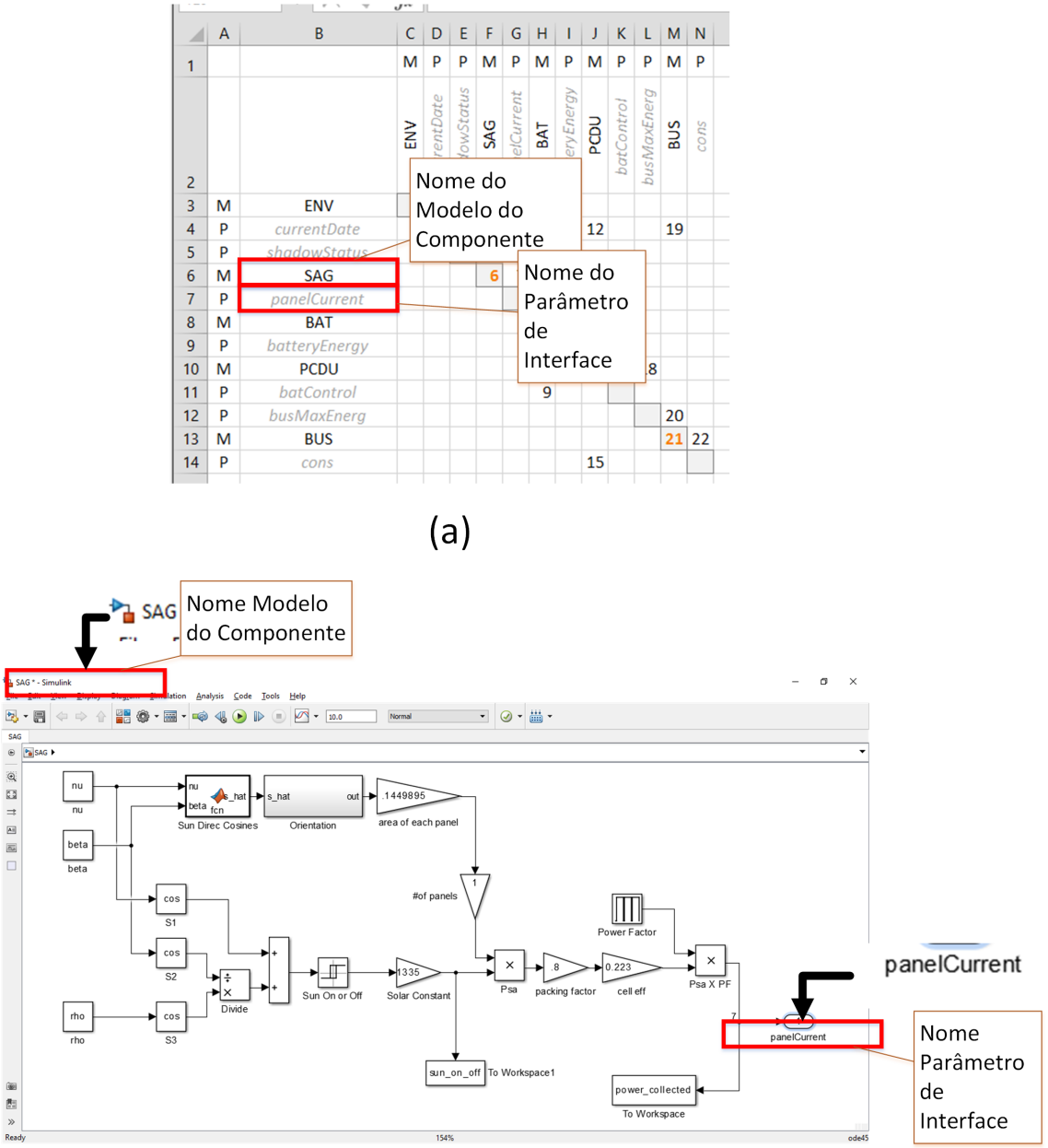
Figura 4.9 - Matriz de Sequência.



A figura apresenta o layout da Matriz de Sequência, adotado no *framework*.
Fonte: Produção do autor.

O nome dos modelos dos equipamentos do subsistema (referenciados aqui como componentes) na Matriz de Sequência e o nome dos parâmetros de interface destes modelos devem ser idênticos ao nome do arquivo que armazena o modelo e o nome dos parâmetros de interface definidos no modelo implementado, respectivamente, conforme ilustra a Figura 4.10.

Figura 4.10 - Padronização dos Nomes na Matriz de Sequência e no Modelo do Componente.

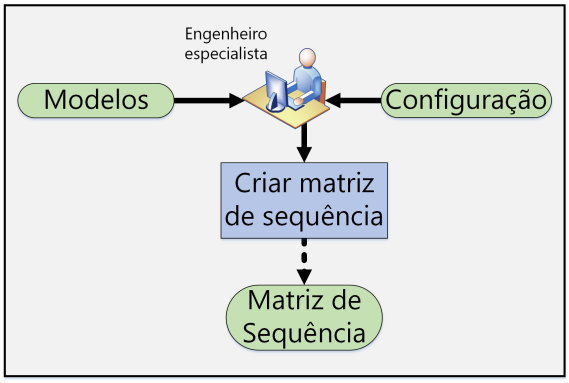


(a) Nome do modelo do componente e do parâmetro de interface na Matriz de Sequência; (b) Nomes do modelo do componente e do parâmetro de interface no arquivo do modelo.

Fonte: Produção do autor.

A etapa descrita nesta seção pode ser observada na Figura 4.11.

Figura 4.11 - Criação da Matriz de Sequência.



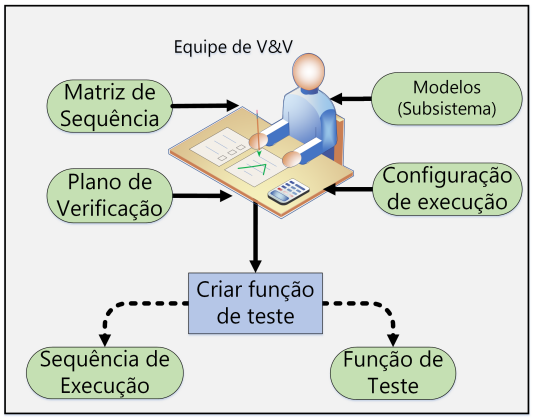
Tanto os modelos do subsistema, quanto as definições da Configuração (ou *setup* de execução) influenciam a Matriz de Sequência.

Fonte: Produção do autor.

4.3 Função de Teste e Sequência de Execução

As funções de teste e a Sequência de Execução caracterizam a finalização do *framework* para a execução dos testes. Estes artefatos dependem de todas as etapas descritas anteriormente e devem ser elaboradas pela Equipe de Verificação e Validação como pode ser observado na Figura 4.12.

Figura 4.12 - Criação da Função de Teste e Sequência de Execução.



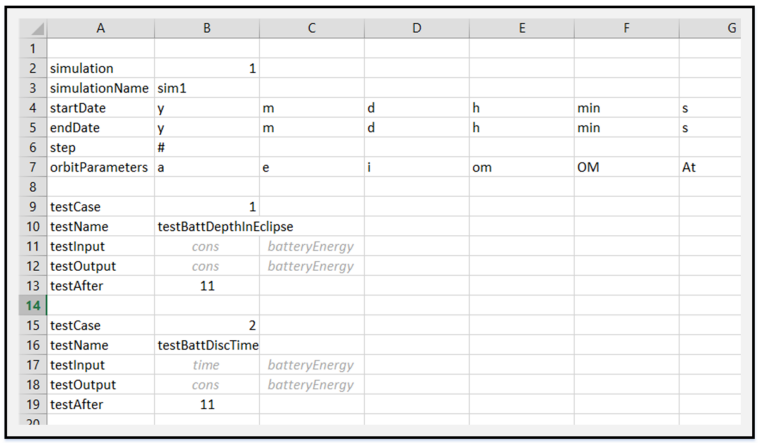
A criação das funções de teste e sequência de execução depende das atividades anteriores do *framework*.

Fonte: Produção do autor.

A Função de Teste representa a forma executável de um caso de teste. Na Matriz de Teste e no Plano de Verificação o caso de teste está na forma abstrata e deve ser traduzido para a linguagem de programação. Além da sua definição, o caso de teste deve excitar e coletar informações dos modelos, tanto dos virtuais quanto dos físicos durante sua execução. Assim, a execução de teste e a simulação são combinadas numa atividade de verificação.

São interpretados pelo *framework* os casos de teste elaborados em planilhas, seguindo o *template* ilustrado na Figura 4.13. A planilha do caso de teste, interpretável pelo simulador do *framework*. Na mesma planilha pode haver mais de um caso de teste.

Figura 4.13 - *Template* dos Casos de Teste.



	A	B	C	D	E	F	G
1							
2	simulation	1					
3	simulationName	sim1					
4	startDate	y	m	d	h	min	s
5	endDate	y	m	d	h	min	s
6	step	#					
7	orbitParameters	a	e	i	om	OM	At
8							
9	testCase	1					
10	testName	testBattDepthInEclipse					
11	testInput	cons	batteryEnergy				
12	testOutput	cons	batteryEnergy				
13	testAfter	11					
14							
15	testCase	2					
16	testName	testBattDiscTime					
17	testInput	time	batteryEnergy				
18	testOutput	cons	batteryEnergy				
19	testAfter	11					

Exemplo de um caso de teste que pode ser interpretado pelo *framework*.
Fonte: Produção do autor.

Os itens apresentados na Figura 4.13 são descritos na Tabela 4.3. Os parâmetros *startDate* e *endDate*, ditam o tempo total da simulação.

Tabela 4.3 - Formato de um Caso de Teste.

Item	**Descrição**
Simulation	Indica qual a simulação que está sendo realizada.
startDate	Indica a data inicial da simulação, no padrão: ano, mês, dia, hora, minuto e segundo.
endDate	Indica a data final simulação, no padrão: ano, mês, dia, hora, minuto e segundo.
step	Parâmetro que dita o passo de simulação.
orbitParameters	Elementos orbitais do satélite, sendo eles: (i) altitude (a); (ii) excentricidade (e); (iii) inclinação (i); (iv) argumento do perigeu (om); (v) ascensão reta do nodo ascendente (OM); e, (vi) anomalia verdadeira (At).
testCase	Indica qual caso de teste está sendo executado e se relaciona com os requisitos presentes na Matriz de Teste.
testName	Nome do caso de teste. Acrescenta um item à Matriz de Sequência, com o mesmo nome.
testInput	Indica o (s) parâmetro (s) de entrada para o caso de teste.

(Continua)

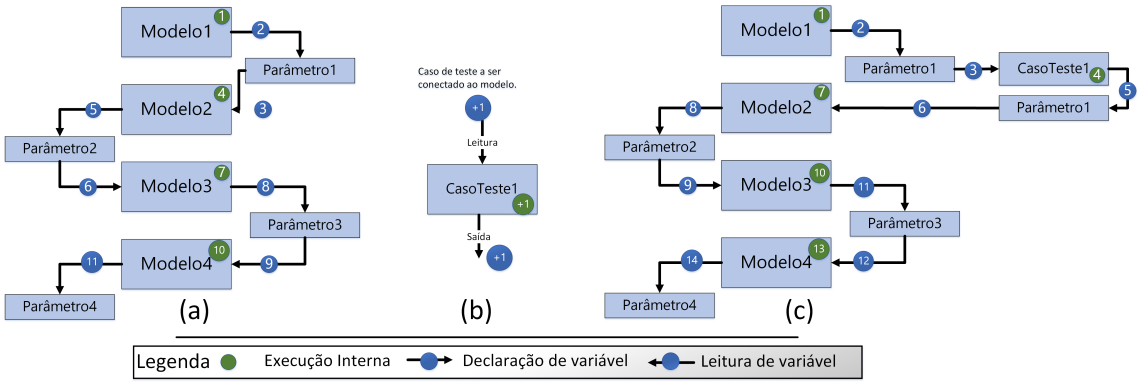
Tabela 4.3 – *Continuação.*

Item	**Descrição**
testOutput	Indica a saída do caso de teste.
testAfter	Indica a partir de qual posição a Matriz de Sequência deve ser alterada.

Produção do autor.

Um caso de teste executável deve fazer as leituras pertinentes para executar sua lógica interna, executar esta lógica e fornecer uma saída para o modelo do subsistema (declaração de variável/parâmetro). Dessa forma são contabilizadas pelo menos três atividades a mais, que não estavam previstas na Matriz de Sequência inicialmente elaborada (seção 4.2). Consequentemente, é necessário reconfigurar a ordem da matriz, após a inclusão do caso de teste, para possibilitar a execução dos modelos e do teste. Dependendo de onde o teste for executado acrescentam-se as atividades extras, em cada um dos valores que representam a ordem da sequência, a partir de onde o teste foi incluído, gerando e sequência de execução, conforme Figura 4.14.

Figura 4.14 - Alteração da Execução dos Modelos.

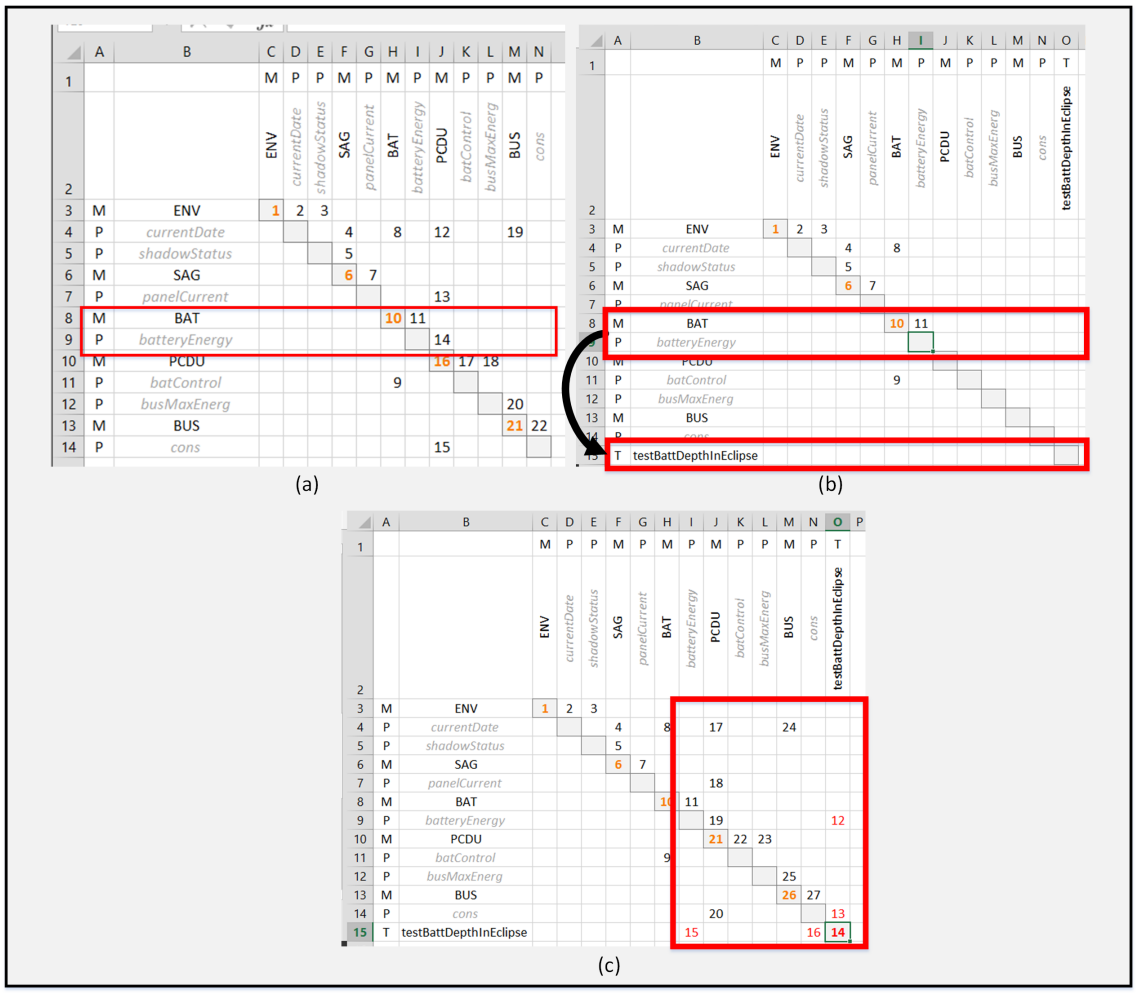


(a) Simulação sem execução de teste (como representada na Matriz de Sequência); (b) Caso de teste a ser conectado ao modelo para execução de teste (como descrito na seção 2.5); (c) Caso de Teste conectado ao modelo, alterando a ordem de execução dos modelos durante a simulação.

Fonte: Produção do autor.

Assim quando o simulador do *framework* faz a leitura do caso de teste, automaticamente, a Matriz de Sequência é alterada para a realização do teste, de acordo com o item "*after*". A Figura 4.15, apresenta a alteração realizada na Matriz de Sequência, em virtude do caso de teste.

Figura 4.15 - Matriz de Sequência Modificada para Execução do Teste.



(a) Matriz de Sequência original; (b) Inclusão do caso de teste na Matriz de Sequência; (c) Ordem de execução alterada, devido o teste a ser executado.

Fonte: Produção do autor.

Deve-se dizer que os casos de teste podem excitar e/ou coletar os dados das interfaces dos modelos, não sendo possível realizar alterações no comportamento do modelo.

4.4 *Log* de Execução

No framework é possível executar os testes automaticamente. Os resultados obtidos durante a simulação e execução dos testes são salvos num arquivo denominado *Log* de Execução. Este arquivo mantém o histórico dos resultados da simulação, dessa forma, a Equipe de V&V pode revisitar os arquivos, sempre que desejarem analisar alguma varável do

modelo.

Como citado anteriormente, a execução dos testes pode ser realizada em três Configurações diferentes, somente modelos virtuais, modelos virtuais e FEE e, com o *hardware* na malha. Independentemente da Configuração, ao final de uma rodada de simulação, todos os resultados são armazenados no *Log* de Execução.

Durante a execução, a coluna "*Result*" da Matriz de Testes será preenchida automaticamente. Uma vez que as funções de teste (ou *script* referente aos casos de teste) guardam a saída esperada, esta será comparada com a saída obtida. Na Matriz de Teste deve aparecer a informação de que o requisito foi verificado ou não, possibilitando realizar a análise dos resultados.

A partir das informações preenchidas na Matriz de Testes, a Equipe de V&V pode dar o veredicto de teste. Caso o requisito tenha sido verificado e o caso de teste passou, o veredicto será positivo, caso contrário, será necessário investigar o motivo pelo qual o requisito não foi verificado. Pode-se admitir dois erros: (i) o modelo do subsistema não corresponde à especificação; e/ou, (ii) erro durante a execução.

5 APLICAÇÃO DO *FRAMEWORK* PARA O SUBSISTEMA DE ENERGIA ELÉTRICA DE PICO E NANOSSATÉLITES

Neste capítulo, descreve-se a aplicação do *framework* para validação de modelos e execução automática de testes funcionais do Subsistema de Energia Elétrica de picossatélites. A aplicação do *framework* consiste na adaptação dos elementos descritos no Capítulo 4, na definição de regras de modelagem para descrever os equipamentos do Subsistema de Energia Elétrica, bem como na descrição das ferramentas utilizadas para implementação dos modelos e os *hardwares* para prototipação do subsistema. Utilizou-se como estudo de caso, o picossatélite *TubeSat* denominado Tancredo 1 (SANTOS et al., 2011), cuja descrição pode ser vista no Apêndice A.

5.1 Requisitos de Entrada

O *framework* para teste funcional do Subsistema de Energia Elétrica tem como entrada os requisitos de alto nível extraídos a partir da documentação do EPS do Tancredo 1, os quais são apresentados na Tabela 5.1, dessa forma, testes serão realizados de modo a verificar estes requisitos.

Tabela 5.1 - Requisitos a Serem Verificados.

Item	Requisito
1	O Subsistema de Energia Elétrica deve fornecer uma tensão de 5 V para o Barramento.
2	A Bateria deve ser recarregada toda vez que o Painel Solar receber energia solar.
3	O estado de carga da Bateria não deve ser menor que 3 Ah.

Produção do autor.

5.2 Modelos

Os modelos a serem configurados no contexto da aplicação do *framework* para validação do Subsistema de Energia Elétrica do picossatélite Tancredo 1 constam dos modelos do subsistema EPS que será avaliado e dos modelos ambientais e de Dinâmica Orbital (ver seção 4.1.4). Os modelos dos equipamentos do EPS considerados aqui foram: solar panel, battery, bus, PCDU, os quais são descritos na seção 5.2.2. Da parte ambiental foi considerado o modelo do propagador de órbta, apresentados em detalhe na seção 5.2.3.

Como visto na Seção 3.3 do Capítulo 3, o ambiente MATALB/*Simulink* fornece uma suíte completa para modelagem e simulação de sistemas. Além disso, é possível utilizar duas abordagens para escrever os programas no MATLAB: programação procedural e programação orientada a objeto (MATLAB, 2015b). Enquanto que o *Simulink* fornece uma

vasta biblioteca de objetos para modelagem gráfica.

Tanto o MATLAB, quanto o *Simulink* permitem a comunicação com diversos *hardwares*, tais como, processadores *Atmel*, *Arduino*, *Kinect* e *Raspberry Pi*, entre outros (CERQUEIRA et al., 2015; MATLAB, 2015a).

Dessa forma, alinhando-se à experiência do autor com a ferramenta (ALVES et al., 2013; RODRIGUES et al., 2013; RODRIGUES; AMBROSIO, 2014; CERQUEIRA et al., 2015; RODRIGUES; AMBROSIO, 2015), decidiu-se utilizar o MATLAB/Simulink como ferramenta de apoio à modelagem e simulação no contexto da especialização do *framework* para o Subsistema de Energia Elétrica.

5.2.1 Metamodelos

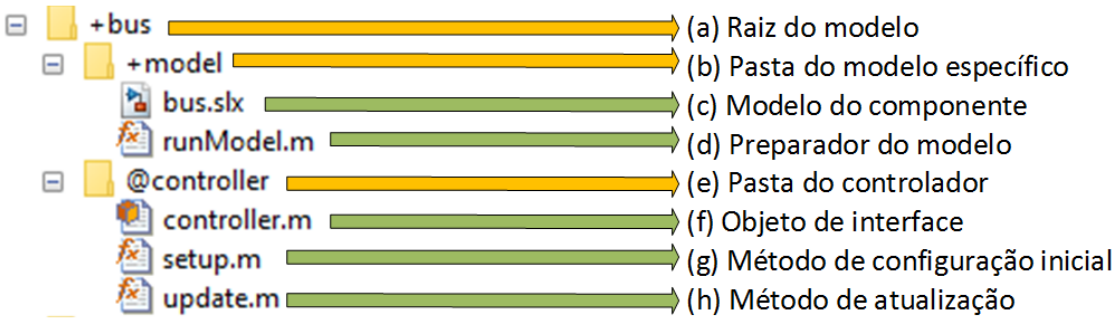
Os modelos dos componentes do subsistema fornecidos pelo engenheiro especialista, podem estar implementados em *scritp* do MATLAB, diagrama de blocos do *Simulink*, *look-up tables*, entre outros.

Este modelo é então, encapsulado em um objeto para ser manipulado no contexto do *framework*. Para tanto foram estabelecidos dois conjuntos de regras: (i) relacionada à estrutura de armazenamento dos arquivos que contém os modelos e (ii) relacionada ao conteúdo destes arquivos (modelos).

5.2.1.1 Estrutura de Arquivos

A estrutura de arquivos deve ter o formato apresentado na Figura 5.1. Os elementos da estrutura e as regras para formação da estrutura são definidas na Tabela 5.2.

Figura 5.1 - Exemplo de uma Estrutura de Arquivos.



A figura apresenta a estrutura de arquivos para o modelo do "bus".

Fonte: Produção do autor.

Tabela 5.2 - Estrutura de Pastas e Arquivos.

Item	**Descrição**	**Formato do Nome**	**Exemplo**
(a)	Raiz do Modelo - Esta pasta contém as pastas: (i) do modelo dos componentes do subsistema e (ii) do controlador do modelo.	$< + >< nomeDoComponente >$[1]	+bus
(b)	Pasta do Modelo Específico - Esta pasta contém o modelo implementado pelo Engenheiro Especialista.	$< + >< model >$	+model
(c)	Modelo do componente Específico - Arquivo do modelo implementado pelo engenheiro.	Formato livre	bus.slx[2]
(d)	Preparador do modelo - Este arquivo prepara os parâmetros de entrada do modelo, realiza uma chamada para execução do modelo (c) e coleta os resultados.	<runModel.m>	runModel.m
(e)	Pasta do controlador - Esta pasta contém os arquivos para descrever a interface e os métodos de inicialização e atualização.	$< @ >< controller >$[3]	@controller

(Continua)

Tabela 5.2 – *Continuação.*

Item	Descrição	Formato do Nome	Exemplo
(f)	Objeto de Interface - Este arquivo é a classe que descreve a interface do modelo com o framework, possui a descrição dos parâmetros e os métodos.	< *controller.m* >[4]	controller.m
(g)	Método de Configuração Inicial - Este método realiza as configurações iniciais necessárias.	< *setup.m* >	setup.m
(h)	Método de atualização - Este método chama o preparador (d), que faz a execução do modelo.	< *update.m* >	update.m

Produção do autor.

Os casos de teste a serem executados seguem a mesma estrutura apresentada na Tabela 5.2, adicionando-se apenas na pasta "@controller", o arquivo "collect.m", cujo resultado de sua execução completa a coluna "*Result*" da Matriz de Teste.

5.2.1.2 Estrutura dos Modelos e Funções de Teste

Os arquivos, tanto dos modelos dos componentes do subsistema (*model*), quanto das funções de teste (*test*), são estruturados por um conjunto de parâmetros e de métodos. Assim, um modelo e uma função de teste são vistas como um Objeto ao qual estão associados parâmetros e métodos, conforme ilustram as Figura 5.2 e 5.3.

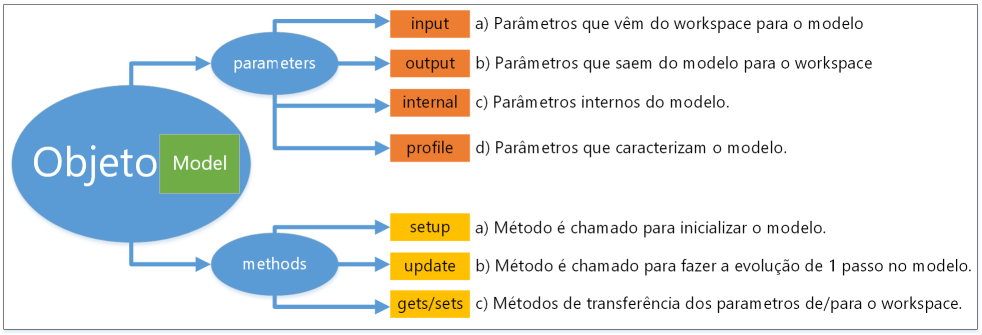
[1]No item (a), o valor "nomeDoComponente" deve ter o mesmo nome do componente, que também está presente na Matriz de Sequência.

[2]Formato ".slx" diz respeito aos modelos implementados em *Simulink*.

[3]No MATALAB faz-se necessário o uso da arroba quando se deseja usar métodos fora do arquivo da classe.

[4]Deve ter o mesmo nome da pasta mencionada no item (e).

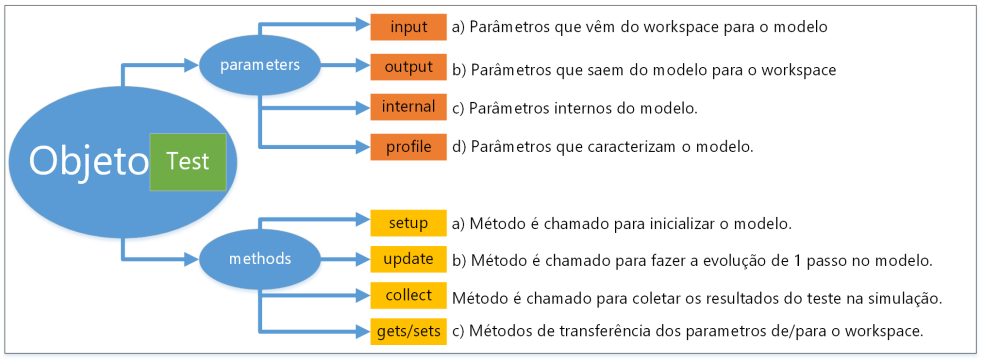
Figura 5.2 - Metamodelo para o Modelo.



A figura apresenta a estrutura do arquivo dos modelos dos componentes do subsistema.

Fonte: Produção do autor.

Figura 5.3 - Metamodelo para o Teste.



A figura apresenta a estrutura do arquivo de uma Função de Teste.

Fonte: Produção do autor.

Os parâmetros são categorizados em:

a) *input*: valor a ser obtido pelo modelo ou pela Função de Teste para execução de sua lógica interna. No caso de uma implementação em MATLAB, este valor é lido no *workspace*[5];

[5]É o local no MATLAB utilizado para gerenciar as variáveis.

b) *output*: valor de saída para outro modelo ou Função de Teste resultante da execução lógica interna deste. No caso de uso do MATLAB este valor é declarado no *workspace*;

c) *internal*: variáveis que não são trocadas entre os modelos. Não são declaradas no *workspace*;

d) *profile*: conjunto de parâmetros que caracterizam o modelo.

Os métodos incluem:

a) *setup*: responsável pela inicialização dos modelos/Função de Teste;

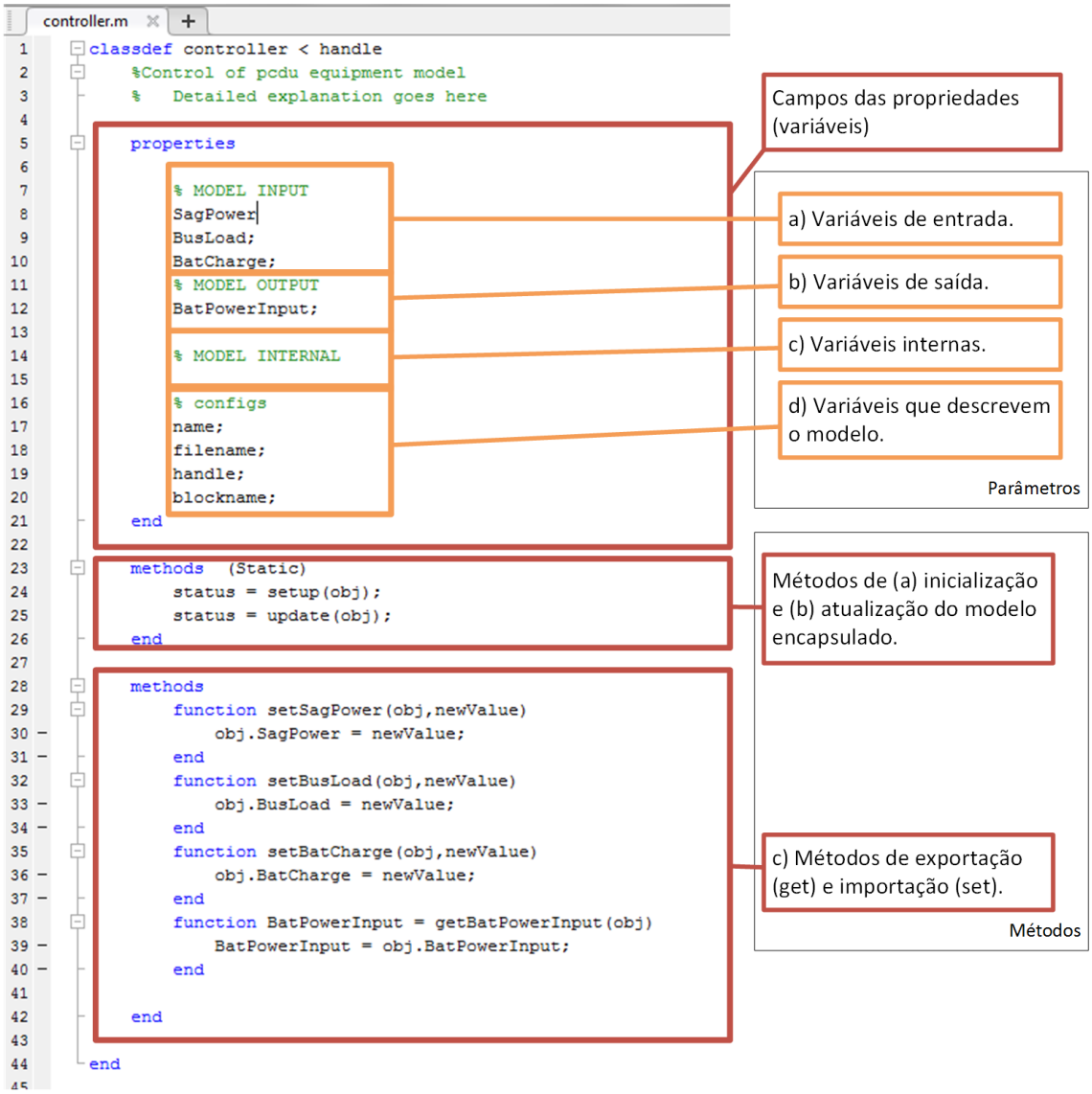
b) *update*: realiza a atualização do modelo/Função de Teste, frente ao passo da simulação;

c) *gets*/*sets*: transferem os parâmetros de/para o outros modelos/funções de teste. No caso do MATLAB, esta transferência é através do *workspace*.

A diferença entre a estrutura dos modelos e a estrutura dos testes é que os testes possuem um método a mais denominado "*collect*", responsável por coletar e armazenar os resultados dos testes, obtidos durante a simulação. Este método é importante para a análise dos resultados, conforme descrito na Seção 4.4 do Capítulo 4.

A Figura 5.4 ilustra aplicação da estrutura proposta ao modelo do equipamento PCDU do subsistema EPS, destacando as variáveis e os métodos.

Figura 5.4 - Controlador do modelo PCDU.



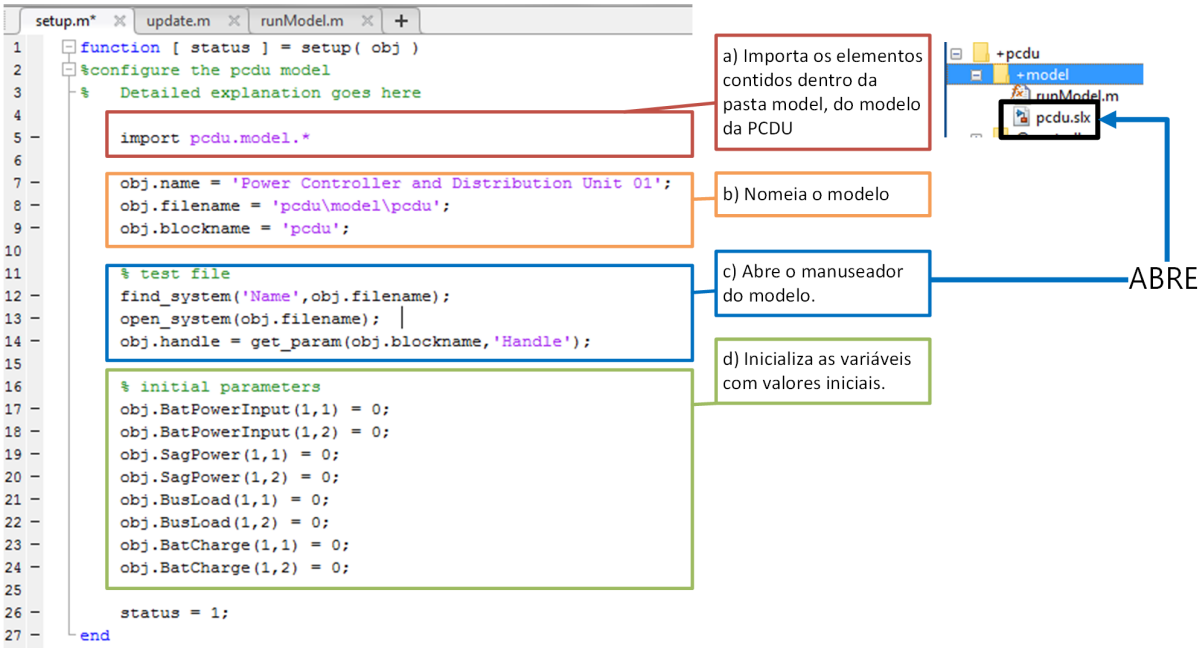
A figura apresenta o Controlador do Modelo da PCDU, seguindo as regras definidas no *framework*.

Fonte: Produção do autor.

O método `setup`, ilustrado na linha 24 da Figura 5.4, inicializa o modelo ou a Função de Teste em questão. Dessa forma, "`status = setup(obj);`" executa a função apresen-

tada na Figura 5.5, onde "obj" é o modelo do equipamento PCDU.

Figura 5.5 - O método "setup.m" do modelo do PCDU.



A figura apresenta o método "setup.m" do modelo do equipamento PCDU.
Fonte: Produção do autor.

O método "setup" é organizado em 4 partes principais, destacadas nos blocos da Figura 5.5, a saber:

a) Importa todos os arquivos contidos na pasta "+model", da PCDU;

b) Atribui um nome ao modelo;

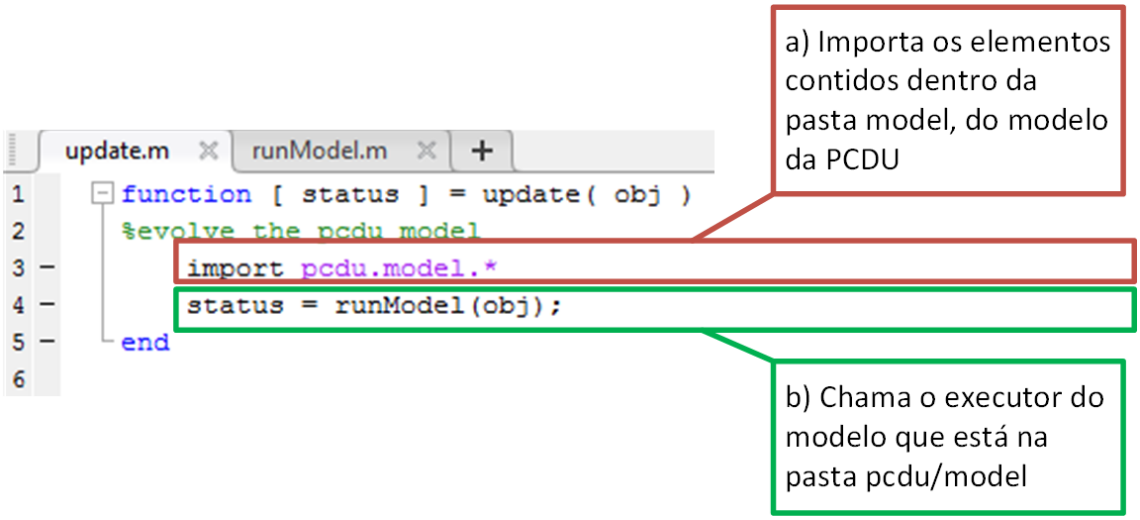
c) Abre o arquivo que contém o modelo executável do componente do subsistema, neste caso, "pcdu.slx".

d) Atribui às variáveis os valores iniciais da simulação.

Todos o métodos retornam `status = 1;`, indicando sucesso na execução. Em trabalhos futuros, poderiam ser analisadas as condições de erro, retornando diferentes valores de "status".

O método “update”, ilustrado na linha 25 da Figura 5.4, é responsável pela atualização do modelo ou da Função de Teste. Dessa forma, a declaração “`status = update(obj);`” executa a função apresentada na Figura 5.6, onde “`obj`” é o modelo do equipamento PCDU.

Figura 5.6 - O método “update.m” da PCDU.



A figura apresenta o método “update.m” da PCDU.
Fonte: Produção do autor.

Os blocos destacados na Figura 5.6, indicam que:

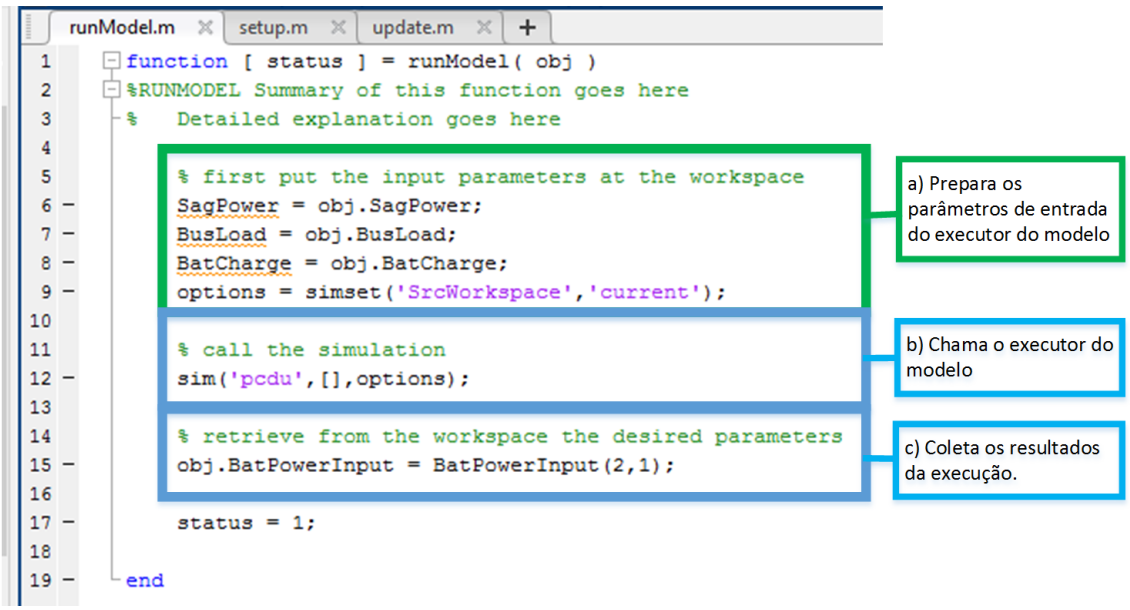
a) o comando “`import`” aqui, também importa todos os arquivos contidos dentro da pasta “+model”, do modelo da PCDU;

b) a linha “`status = runModel(obj)`” chama a rotina, responsável por inicializar a execução do modelo. Esta rotina está ilustrada na Figura 5.7

O modelo executável do subsistema é armazenado na pasta “+pcdu/+model”, a fim de manter uma pasta com um controlador genérico e suas informações (parâmetros e métodos). Mas como esta solução impede a chamada do modelo do componente diretamente pelo método “update”, para executar o modelo fez-se necessário o uso de uma função auxiliar “runModel”.

A Figura 5.7 mostra o método “runModel.m” que dispara a execução do modelo da PCDU.

Figura 5.7 - Exemplo do método “runModel.m” da PCDU.



A figura apresenta o método “runModel.m” da PCDU.
Fonte: Produção do autor.

O método “runModel.m” é organizado em três partes conforme os blocos destacados na Figura 5.7, os quais são explicados a seguir:

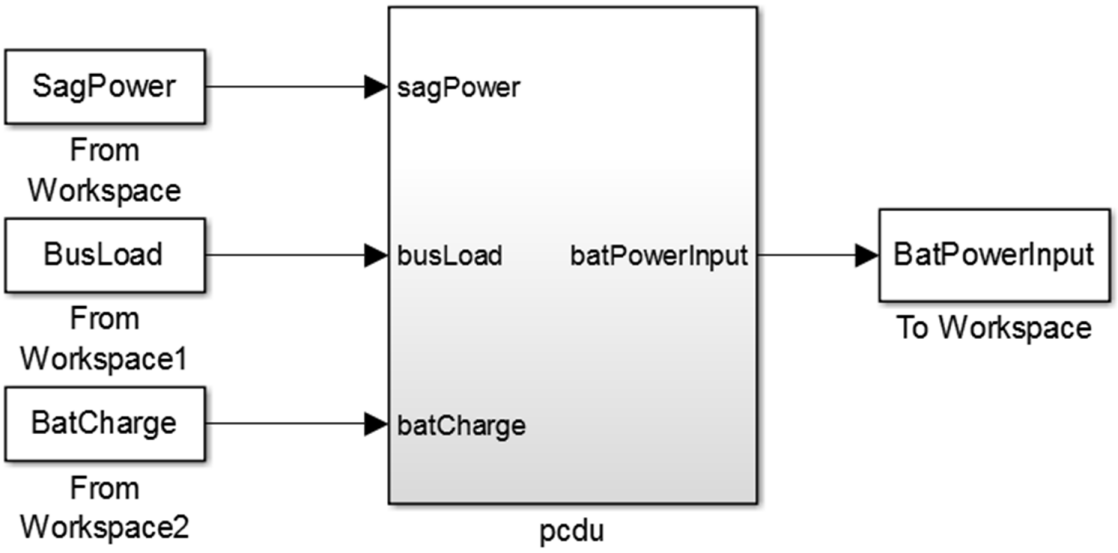
a) Prepara os parâmetros de entrada declarando, no *workspace*, os objetos que fazem interface com o modelo a ser executado e, por fim, salva o *workspace* dentro de `options` . Salvar o *workspace* é fundamental, pois os valores presentes nele são carregados no modelo do componente do subsistema, a fim de torná-lo executável;

b) Chama o executor do modelo. A linha onde lê-se “`sim('pcdu',[],options);`” executa o modelo do componente do subsistema, neste caso implementado em *Simulink*;

c) Coleta e salva os resultados da execução da simulação, que estão no *workspace* do objeto, neste caso, `obj.BatPowerInput`.

A Figura 5.8 ilustra o modelo executável do equipamento PCDU.

Figura 5.8 - Modelo Executável da PCDU.



A figura apresenta um diagrama de blocos do modelo executável da PCDU, implementado em *Simulink*.

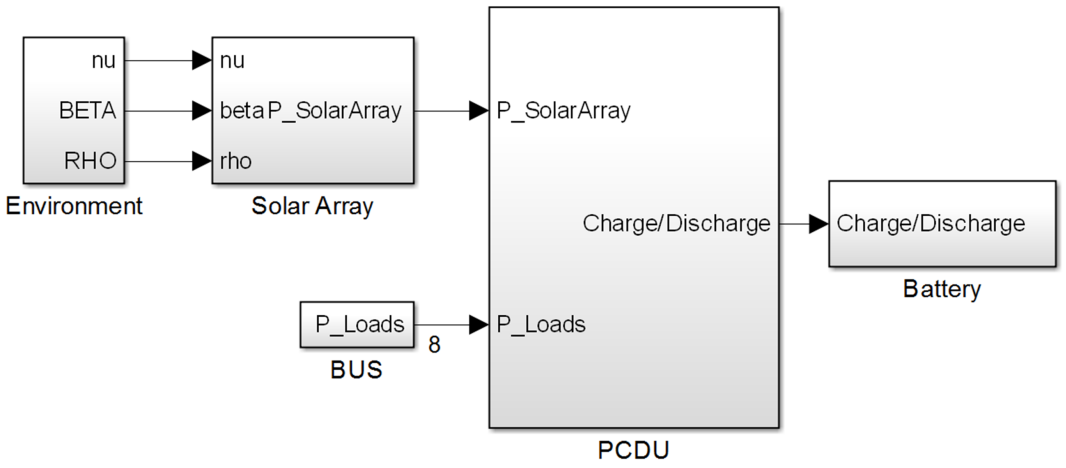
Fonte: Produção do autor.

5.2.2 Modelo do Subsistema EPS

Conforme supracitado no decorrer do texto, o modelo de cada componente do subsistema deve ser implementado separadamente. Tomando-se como referência o modelo descrito por Melone (2009), apresentado com mais detalhes no Apêndice B, esta seção apresenta a maneira com a qual os modelos executáveis dos componentes do subsistema devem ser implementados.

A Figura 5.9 apresenta o modelo completo do EPS.

Figura 5.9 - Visão geral do Modelo do EPS.



Visão geral do modelo do EPS do Tancredo 1, implementado em *Simulink*.
Fonte: Adaptada de Melone (2009).

O modelo do EPS é composto pelo modelo dos equipamentos: solar panel, battery, bus e PCDU. A Figura 5.10 ilustra estes modelos e suas interfaces.

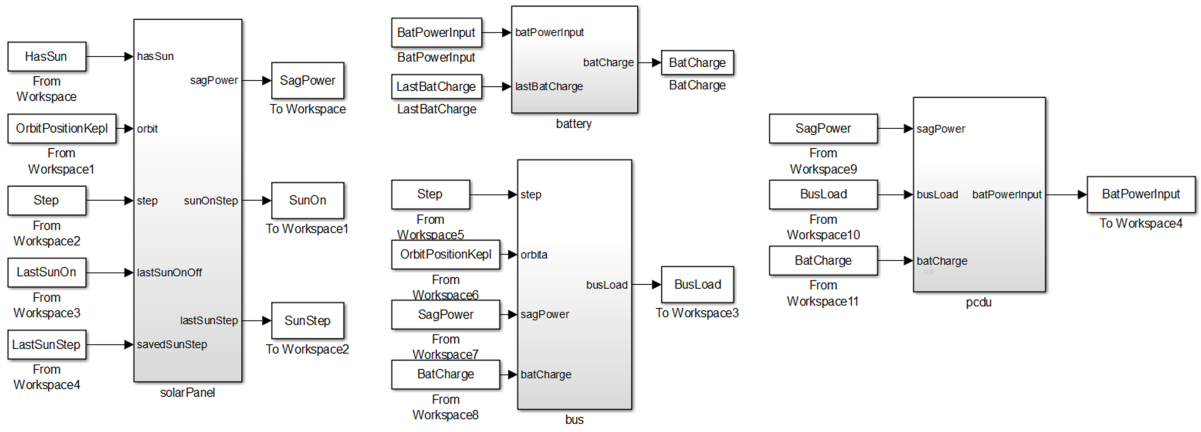
O modelo cuja entrada depende da saída de outro modelo, deve possuir o bloco "*From Workspace*" em sua interface. Consequentemente, o componente que fornece uma saída utilizada como entrada para outro modelo deve possuir o bloco "*To Workspace*", em sua interface. A Figura 5.11 apresenta o modelo da PCDU que depende de um parâmetro calculado pelo modelo da bateria, assim, a bateria deve declarar a variável ("*To Workspace*") e a PCDU lê-la ("*From Workspace*").

5.2.3 Modelo do Propagador de Órbita

O modelo propagador de órbita executa a propagação de órbita do satélite e fornece sua posição em um dado instante de tempo. A posição indicada pelo propagador de órbita, conforme ilustra a Figura 5.12, é de suma importância para o EPS, uma vez que, esta posição define se o satélite está em eclipse ou não.

As entradas para que o modelo propagador de órbita evolua com o tempo são: os parâmetros orbitais, a data inicial e final da simulação e o passo da simulação. Os parâmetros orbitais são listados a seguir: (i) Altitude [km]; (ii) Excentricidade; (iii) Inclinação [°]; (iv) Argumento do Perigeu [°]; (v) Ascensão Reta do Nodo Descendente [°]; e, (vi) Anomalia Verdadeira [°].

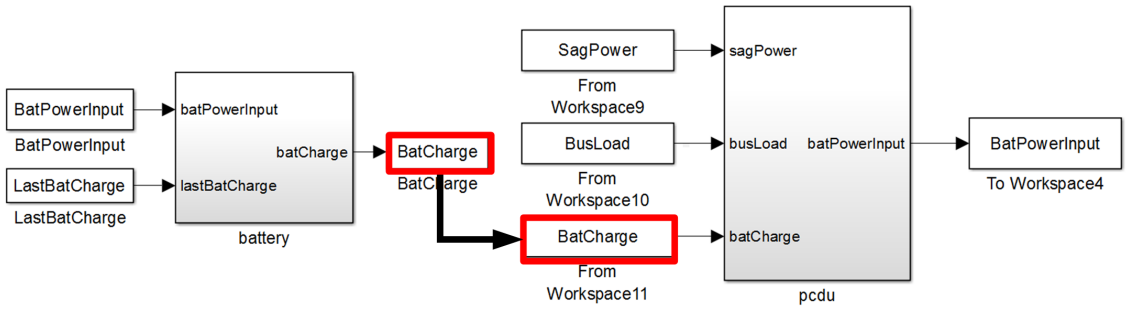
Figura 5.10 - Modelos Executáveis e suas Interfaces.



A figura apresenta o conjunto dos modelos executáveis que são interligados pela Matriz de Sequência.

Fonte: Produção do autor.

Figura 5.11 - Blocos "*To Workspace*" e "*From Workspace*".

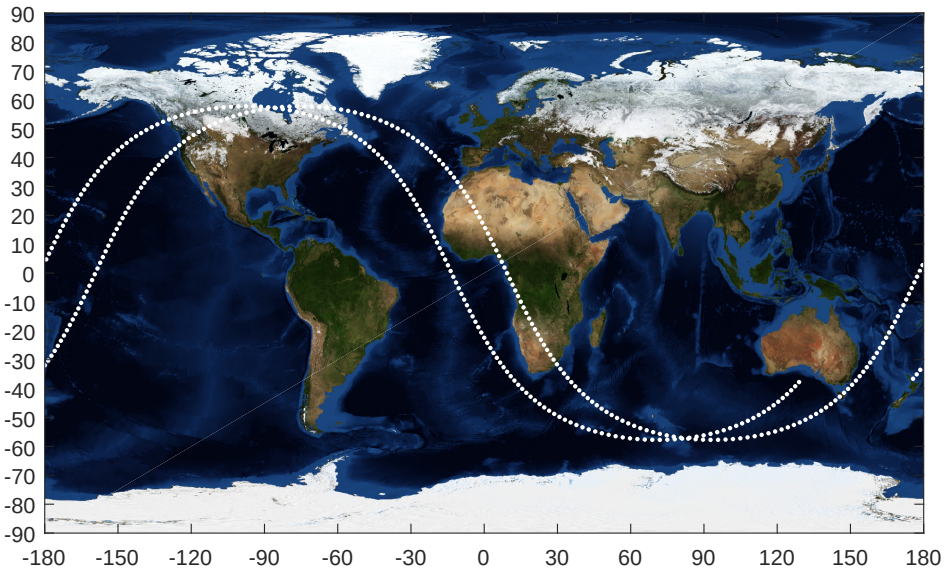


A figura apresenta a relação entre as interfaces da bateria e da PCDU.

Fonte: Produção do autor.

O modelo propagador de órbita, neste exemplo fornece para o modelo do EPS: (i) posição do satélite e (ii) a informação se o satélite está ou não em eclipse.

Figura 5.12 - Propagador de Órbita.



Os pontos brancos indicam a posição do satélite em um determinado instante de tempo.

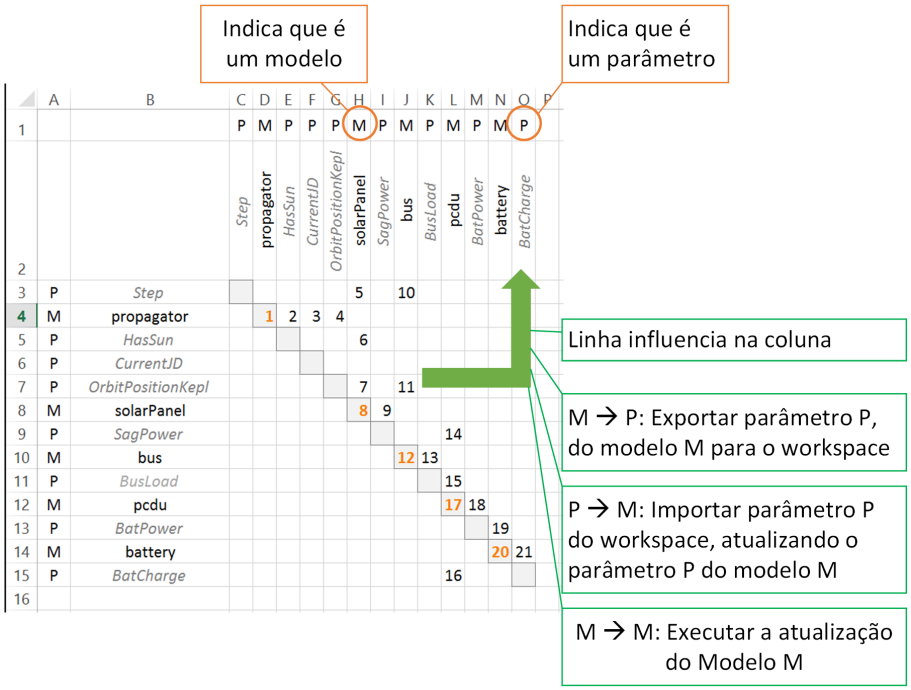
Fonte: Produção do autor.

5.3 Matriz de Sequência

Esta seção apresenta o modo como a Matriz de Sequência é processada para conectar os modelos e permitir a troca de informações entre eles.

A Figura 5.13 ilustra uma Matriz de Sequência relacionando os modelos e os parâmetros que compõe o subsistema EPS do Tancredo 1.

Figura 5.13 - Matriz de Sequência do Estudo de Caso.



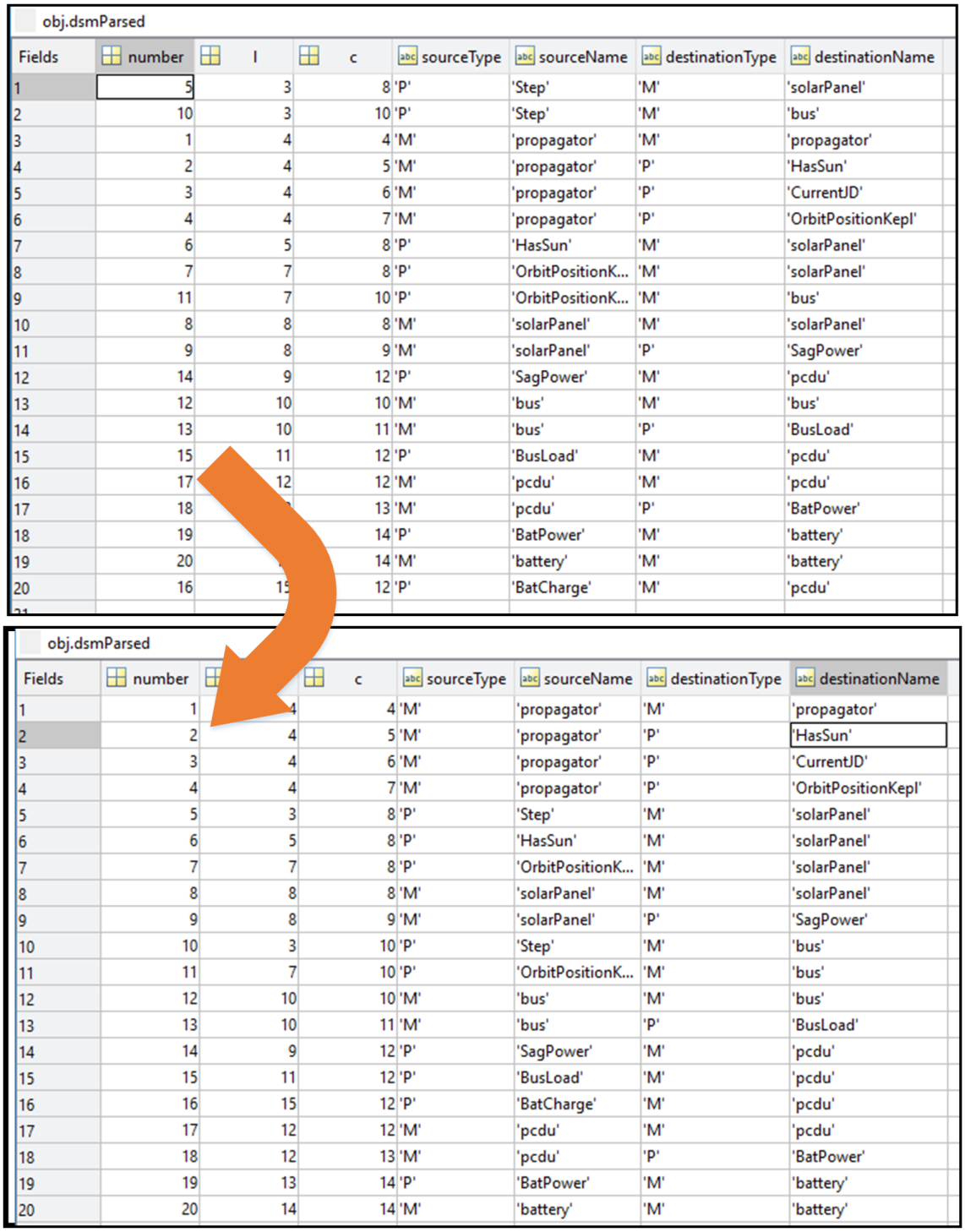
A figura apresenta a Matriz de Sequência, que inter-relaciona os modelos e os parâmetros do estudo de caso.

Fonte: Produção do autor.

A conversão da Matriz de Sequência em um objeto é realizada através dos passos descritos a seguir e ilustrados nas Figuras 5.14 até 5.18.

- (1) distribuir de forma crescente um número que indica a ordem de execução„ conforme ilustra a Figura 5.14;
- (2) atribuir as sentenças que indicam o que deve ser feito (ou ações). As possíveis setenças, ou ações, são: (i) ler um parâmetro (P→M), (ii) executar um modelo referente ao equipamento do subsistema (M→M) e (iii) declarar uma variável (M→P), conforme Figuras 5.15 e 5.16;
- (3) estabelecer as sentenças de inicialização, conforme Figura 5.17. As setenças necessárias são: "*import*", instanciar e "*setup*";
- (4) preparar a simulação levando em conta os casos de teste;
- (5) executar a simulação. Cada passo da execução chama o método "runModel" de cada modelo ou Função de Teste, como pode ser observado na Figura 5.18.

Figura 5.14 - Passo (1): Indicar a ordem das ações na da Matriz de Sequência.



obj.dsmParsed

Fields	number	l	c	sourceType	sourceName	destinationType	destinationName
1	5	3	8	'P'	'Step'	'M'	'solarPanel'
2	10	3	10	'P'	'Step'	'M'	'bus'
3	1	4	4	'M'	'propagator'	'M'	'propagator'
4	2	4	5	'M'	'propagator'	'P'	'HasSun'
5	3	4	6	'M'	'propagator'	'P'	'CurrentJD'
6	4	4	7	'M'	'propagator'	'P'	'OrbitPositionKepl'
7	6	5	8	'P'	'HasSun'	'M'	'solarPanel'
8	7	7	8	'P'	'OrbitPositionK...	'M'	'solarPanel'
9	11	7	10	'P'	'OrbitPositionK...	'M'	'bus'
10	8	8	8	'M'	'solarPanel'	'M'	'solarPanel'
11	9	8	9	'M'	'solarPanel'	'P'	'SagPower'
12	14	9	12	'P'	'SagPower'	'M'	'pcdu'
13	12	10	10	'M'	'bus'	'M'	'bus'
14	13	10	11	'M'	'bus'	'P'	'BusLoad'
15	15	11	12	'P'	'BusLoad'	'M'	'pcdu'
16	17	12	12	'M'	'pcdu'	'M'	'pcdu'
17	18	[illegible]	13	'M'	'pcdu'	'P'	'BatPower'
18	19	[illegible]	14	'P'	'BatPower'	'M'	'battery'
19	20	[illegible]	14	'M'	'battery'	'M'	'battery'
20	16	15	12	'P'	'BatCharge'	'M'	'pcdu'

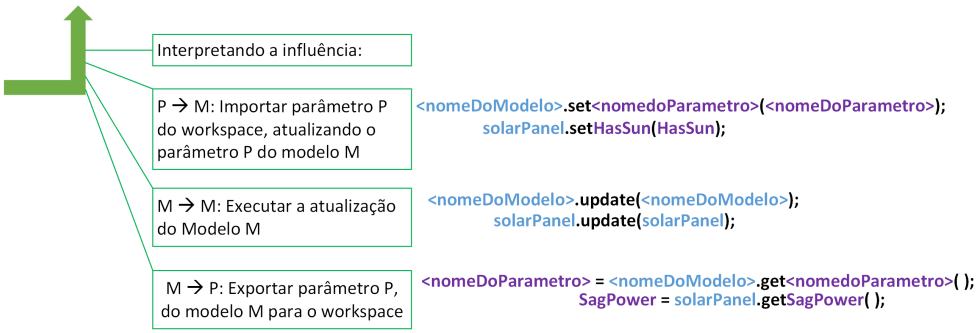
obj.dsmParsed

Fields	number	[illegible]	c	sourceType	sourceName	destinationType	destinationName
1	1	[illegible]	4	'M'	'propagator'	'M'	'propagator'
2	2	4	5	'M'	'propagator'	'P'	'HasSun'
3	3	4	6	'M'	'propagator'	'P'	'CurrentJD'
4	4	4	7	'M'	'propagator'	'P'	'OrbitPositionKepl'
5	5	3	8	'P'	'Step'	'M'	'solarPanel'
6	6	5	8	'P'	'HasSun'	'M'	'solarPanel'
7	7	7	8	'P'	'OrbitPositionK...	'M'	'solarPanel'
8	8	8	8	'M'	'solarPanel'	'M'	'solarPanel'
9	9	8	9	'M'	'solarPanel'	'P'	'SagPower'
10	10	3	10	'P'	'Step'	'M'	'bus'
11	11	7	10	'P'	'OrbitPositionK...	'M'	'bus'
12	12	10	10	'M'	'bus'	'M'	'bus'
13	13	10	11	'M'	'bus'	'P'	'BusLoad'
14	14	9	12	'P'	'SagPower'	'M'	'pcdu'
15	15	11	12	'P'	'BusLoad'	'M'	'pcdu'
16	16	15	12	'P'	'BatCharge'	'M'	'pcdu'
17	17	12	12	'M'	'pcdu'	'M'	'pcdu'
18	18	12	13	'M'	'pcdu'	'P'	'BatPower'
19	19	13	14	'P'	'BatPower'	'M'	'battery'
20	20	14	14	'M'	'battery'	'M'	'battery'

A figura apresenta o resultado da ordenação da Matriz de Sequência.

Fonte: Produção do autor.

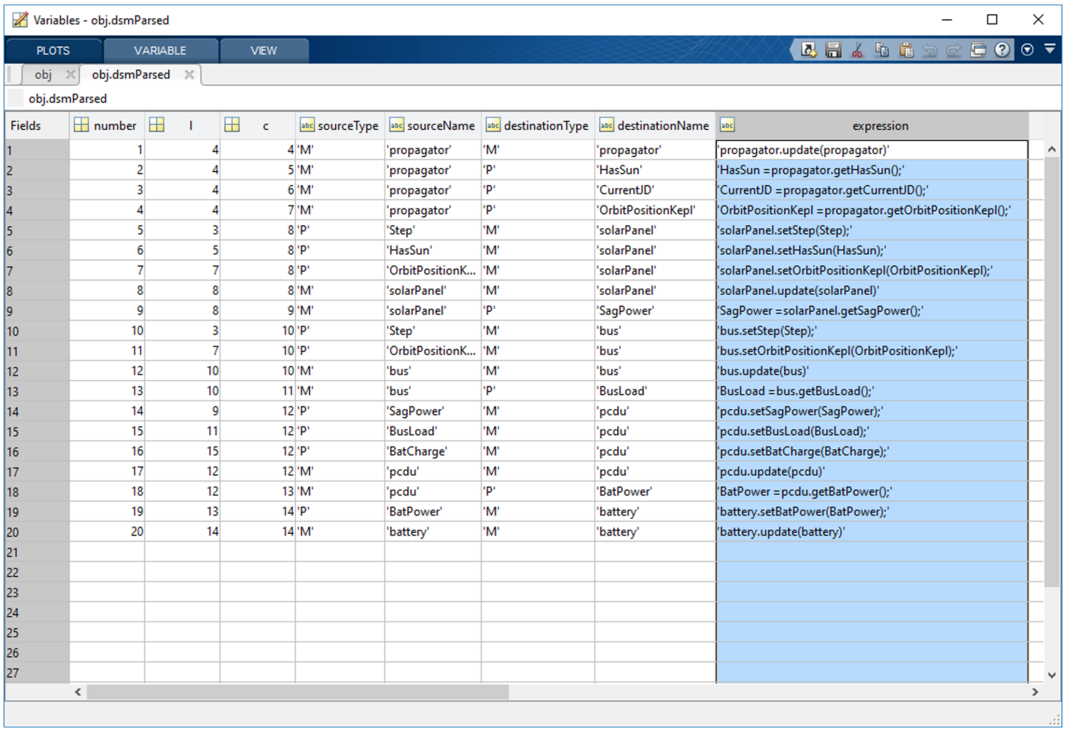
Figura 5.15 - Passo (2): Atribuir as Ações.



A figura apresenta as ações de leitura, atualização e declaração.

Fonte: Produção do autor.

Figura 5.16 - Ilustrando Sentenças de Execução.


Variables - obj.dsmParsed

PLOTS VARIABLE VIEW

obj obj.dsmParsed

obj.dsmParsed

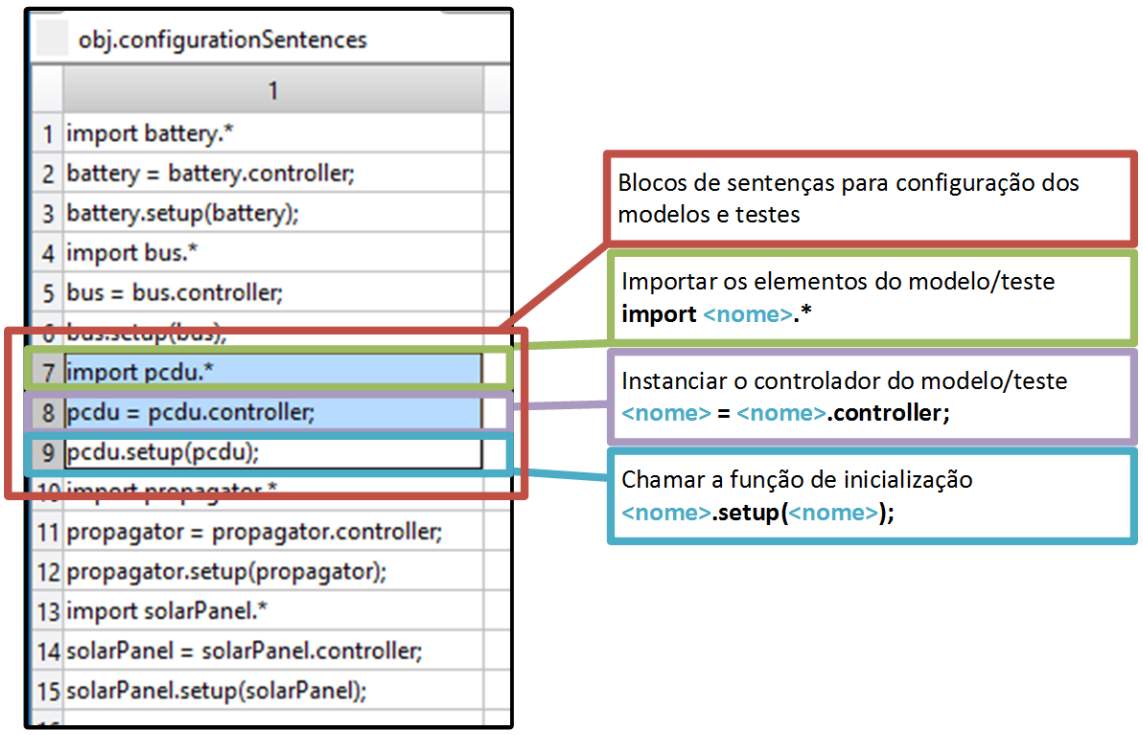
Fields	number	l	c	sourceType	sourceName	destinationType	destinationName	expression
1	1	4	4	'M'	'propagator'	'M'	'propagator'	'propagator.update(propagator)'
2	2	4	5	'M'	'propagator'	'P'	'HasSun'	'HasSun = propagator.getHasSun();'
3	3	4	6	'M'	'propagator'	'P'	'CurrentJD'	'CurrentJD = propagator.getCurrentJD();'
4	4	4	7	'M'	'propagator'	'P'	'OrbitPositionKepl'	'OrbitPositionKepl = propagator.getOrbitPositionKepl();'
5	5	3	8	'P'	'Step'	'M'	'solarPanel'	'solarPanel.setStep(Step);'
6	6	5	8	'P'	'HasSun'	'M'	'solarPanel'	'solarPanel.setHasSun(HasSun);'
7	7	7	8	'P'	'OrbitPositionK...	'M'	'solarPanel'	'solarPanel.setOrbitPositionKepl(OrbitPositionKepl);'
8	8	8	8	'M'	'solarPanel'	'M'	'solarPanel'	'solarPanel.update(solarPanel)'
9	9	8	9	'M'	'solarPanel'	'P'	'SagPower'	'SagPower = solarPanel.getSagPower();'
10	10	3	10	'P'	'Step'	'M'	'bus'	'bus.setStep(Step);'
11	11	7	10	'P'	'OrbitPositionK...	'M'	'bus'	'bus.setOrbitPositionKepl(OrbitPositionKepl);'
12	12	10	10	'M'	'bus'	'M'	'bus'	'bus.update(bus)'
13	13	10	11	'M'	'bus'	'P'	'BusLoad'	'BusLoad = bus.getBusLoad();'
14	14	9	12	'P'	'SagPower'	'M'	'pcdu'	'pcdu.setSagPower(SagPower);'
15	15	11	12	'P'	'BusLoad'	'M'	'pcdu'	'pcdu.setBusLoad(BusLoad);'
16	16	15	12	'P'	'BatCharge'	'M'	'pcdu'	'pcdu.setBatCharge(BatCharge);'
17	17	12	12	'M'	'pcdu'	'M'	'pcdu'	'pcdu.update(pcdu)'
18	18	12	13	'M'	'pcdu'	'P'	'BatPower'	'BatPower = pcdu.getBatPower();'
19	19	13	14	'P'	'BatPower'	'M'	'battery'	'battery.setBatPower(BatPower);'
20	20	14	14	'M'	'battery'	'M'	'battery'	'battery.update(battery)'
21								
22								
23								
24								
25								
26								
27								

A figura ilustra ações de leitura, atualização e declaração.

Fonte: Produção do autor.

As sentenças de inicialização são chamadas antes de iniciar o *loop* de execução de todos os componentes a serem utilizados pelo *framework* (modelos e funções de teste). Para cada componente são necessárias três sentenças: (i) fazer um `import` da pasta do modelo, (ii) instanciar um novo modelo e (iii) chamar o método "*setup*" desta nova instância, conforme pode ser observado na Figura 5.17.

Figura 5.17 - Passo (3): Estabelecer as Sentenças de Inicialização.



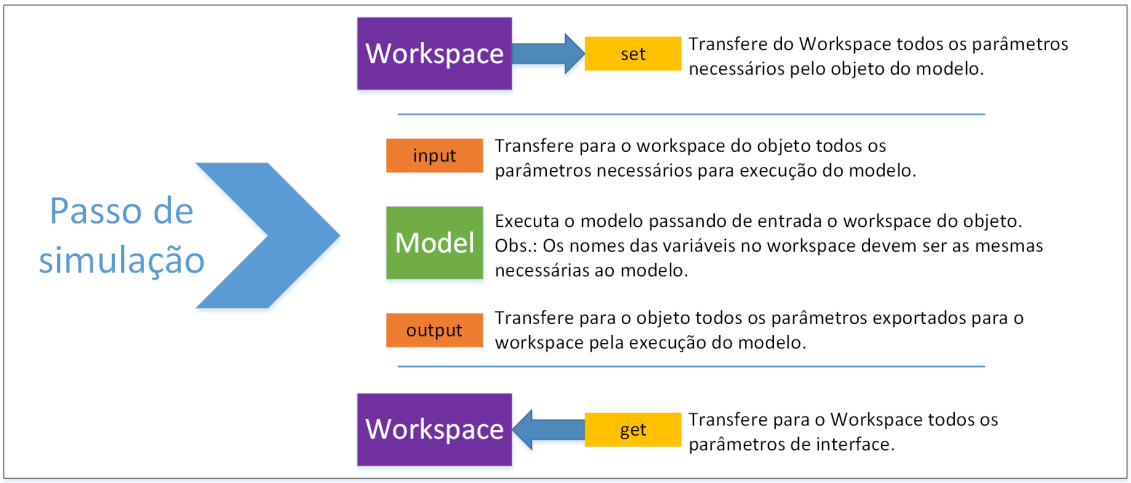
A figura exemplifica a sequência de inicialização dos componentes, ressaltando as três sentenças necessárias: `import`, instanciar, "*setup*".

Fonte: Produção do autor.

A quarta etapa consta de preparar a simulação, tendo em vista os casos de teste ilustrados pela Figura 4.13.

Na quinta etapa executa-se um passo da simulação, dessa forma, o passo da execução chama a "runModel.m" de cada modelo ou Função de Teste, como pode ser observado na Figura 5.18.

Figura 5.18 - Passo (5): Executar um Passo de Simulação.



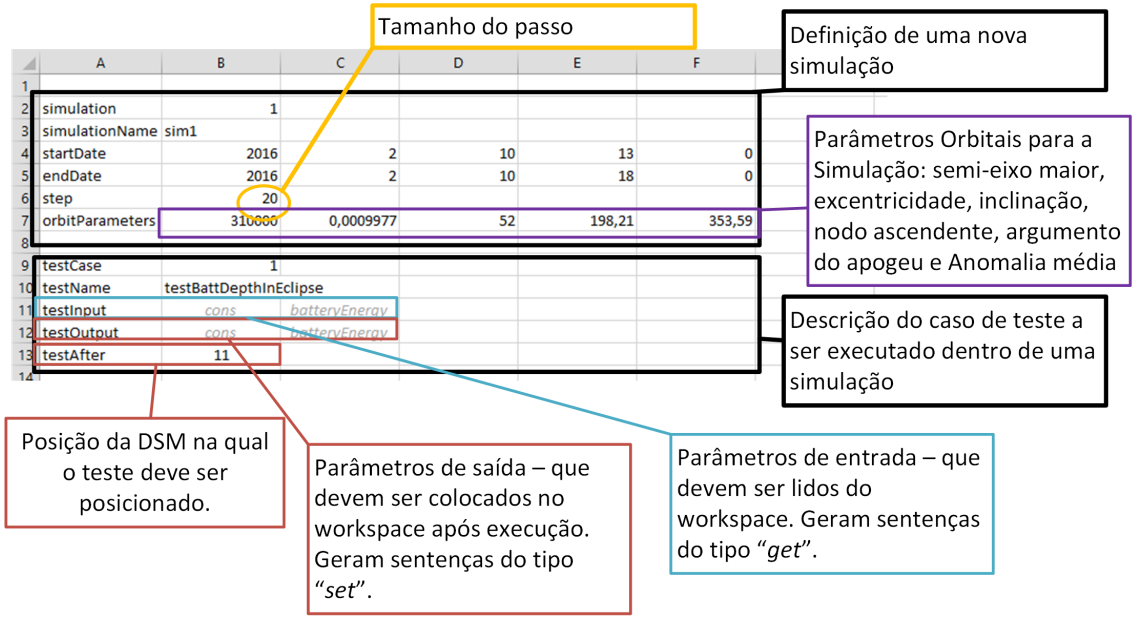
A figura ilustra a execução de um passo de simulação.
Fonte: Produção do autor.

5.4 Caso de Teste

Os casos de teste especificados inicialmente de forma manial no Plano de V&V são transformados em Funções de Teste, e para que sejam executados automaticamente, outras informações são incluídas na Matriz de Sequência, já lida pelo *framework*, para sequência de execução durante uma rodada de simulação.

O processo de inclusão de um caso de teste para ser excetuado durante uma simulação é descrito na Seção 4.3, do Capítulo 4. A Figura 5.19 ilustra a interpretação de um caso de teste e da sua execução numa rodada de simulação.

Figura 5.19 - Campos que Descrevem um do Caso de Teste e uma Simulação.



A figura ilustra os campos que descrevem um caso de teste e campos que descrevem as simulações.

Fonte: Produção do autor.

Da mesma forma que as ações foram definidas e inseridas na Matriz de Sequência para a execução dos modelos dos componentes do subsistema EPS, são definidas sentenças específicas que indicam as ações para execução de um caso de teste automaticamente, conforme Figura 5.20, sendo elas:

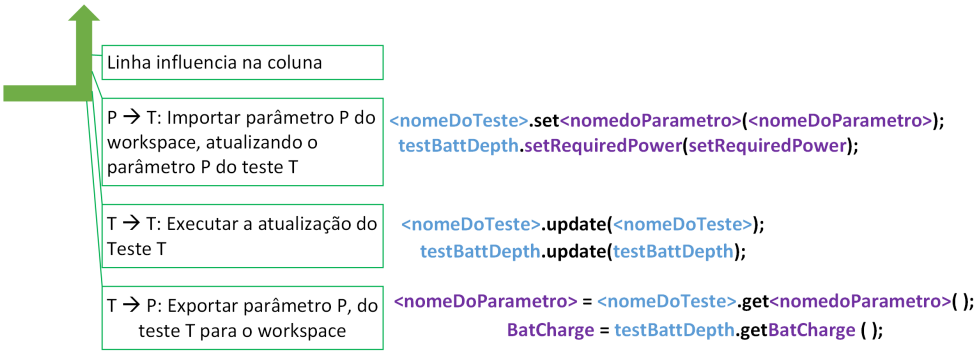
a) P→T: o valor de uma variável é lido de forma que o teste possa executar sua lógica interna;

b) T→T: a lógica interna do teste é executada;

c) T→P: após a execução da lógica interna do teste, o resultado é declarado no *workspace*, de modo a ser utilizado por outro modelo do subsistema EPS.

A inclusão dos casos de teste modifica a Matriz de Sequência e, consequentemente, altera a ordem de execução dos modelos, conforme Figura 4.14, gerando um objeto (ilustrado na Figura 5.21), no qual a ordem dos modelos já foi alterada.

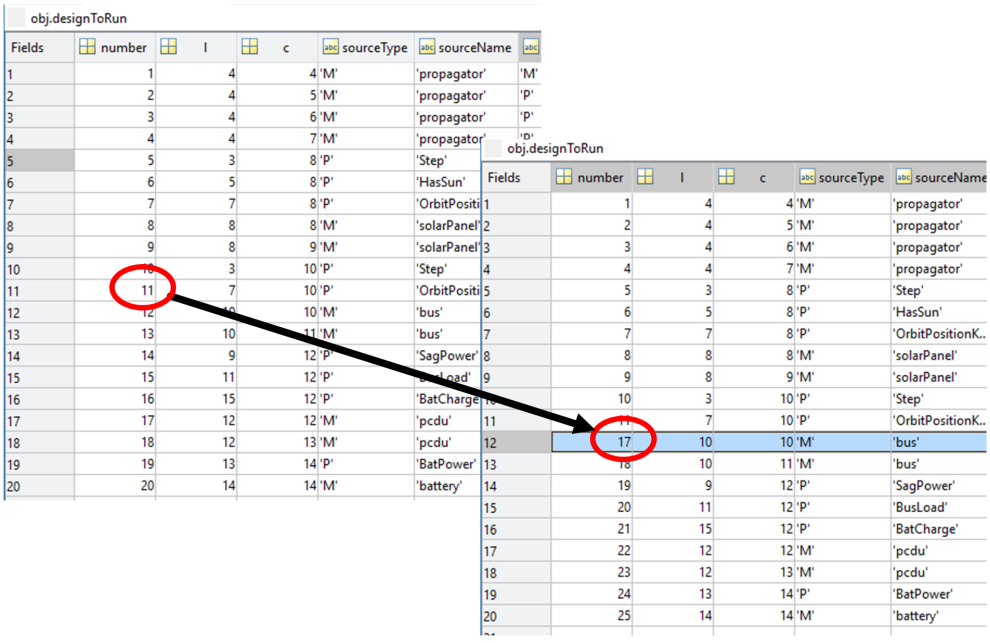
Figura 5.20 - Criando as Ações para Execução dos Casos de Teste.



A figura apresenta a criação das ações de uma Função de Teste para executar o caso de teste.

Fonte: Produção do autor.

Figura 5.21 - Objeto da Matriz de Sequência Modificado.

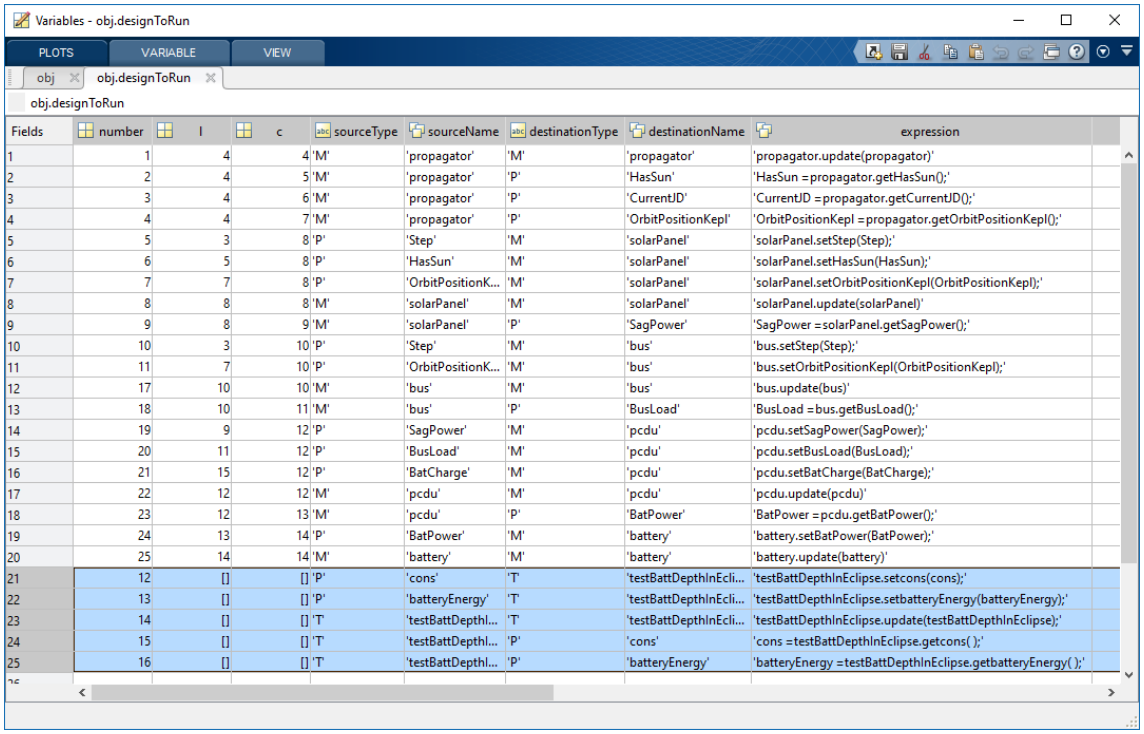


A figura apresenta o objeto da Matriz de Sequência modificado após a inclusão do caso de teste na Matriz de Sequência.

Fonte: Produção do autor.

Na Figura 5.22 é possível visualizar que os casos de teste são inseridos no final do objeto, fazendo-se necessário coloca-los novamente em ordem, como pode ser observado na Figura 5.23. Com esta solução o acoplamento dos casos de teste, para execução dos testes, ocorre automaticamente.

Figura 5.22 - Testes Inseridos na Matriz de Sequência.

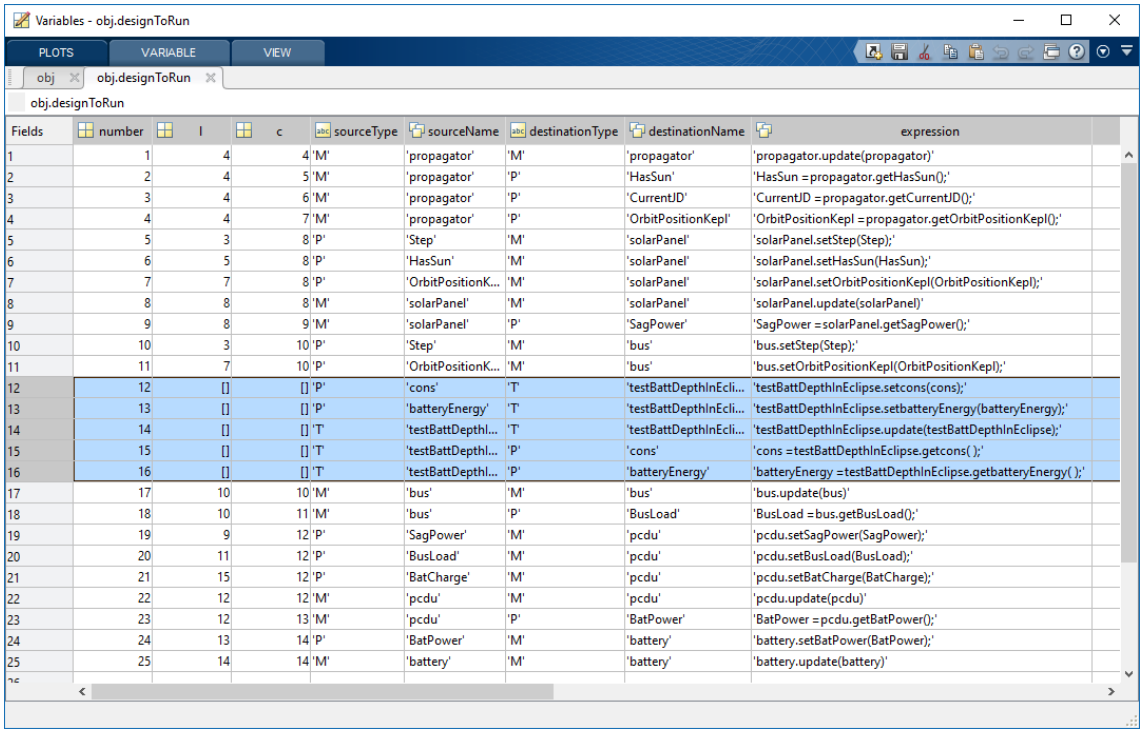


Fields	number	l	c	sourceType	sourceName	destinationType	destinationName	expression
1	1	4	4	'M'	'propagator'	'M'	'propagator'	'propagator.update(propagator)'
2	2	4	5	'M'	'propagator'	'P'	'HasSun'	'HasSun = propagator.getHasSun();'
3	3	4	6	'M'	'propagator'	'P'	'CurrentJD'	'CurrentJD = propagator.getCurrentJD();'
4	4	4	7	'M'	'propagator'	'P'	'OrbitPositionKepl'	'OrbitPositionKepl = propagator.getOrbitPositionKepl();'
5	5	3	8	'P'	'Step'	'M'	'solarPanel'	'solarPanel.setStep(Step);'
6	6	5	8	'P'	'HasSun'	'M'	'solarPanel'	'solarPanel.setHasSun(HasSun);'
7	7	7	8	'P'	'OrbitPositionK...	'M'	'solarPanel'	'solarPanel.setOrbitPositionKepl(OrbitPositionKepl);'
8	8	8	8	'M'	'solarPanel'	'M'	'solarPanel'	'solarPanel.update(solarPanel)'
9	9	8	9	'M'	'solarPanel'	'P'	'SagPower'	'SagPower = solarPanel.getSagPower();'
10	10	3	10	'P'	'Step'	'M'	'bus'	'bus.setStep(Step);'
11	11	7	10	'P'	'OrbitPositionK...	'M'	'bus'	'bus.setOrbitPositionKepl(OrbitPositionKepl);'
12	17	10	10	'M'	'bus'	'M'	'bus'	'bus.update(bus)'
13	18	10	11	'M'	'bus'	'P'	'BusLoad'	'BusLoad = bus.getBusLoad();'
14	19	9	12	'P'	'SagPower'	'M'	'pcdu'	'pcdu.setSagPower(SagPower);'
15	20	11	12	'P'	'BusLoad'	'M'	'pcdu'	'pcdu.setBusLoad(BusLoad);'
16	21	15	12	'P'	'BatCharge'	'M'	'pcdu'	'pcdu.setBatCharge(BatCharge);'
17	22	12	12	'M'	'pcdu'	'M'	'pcdu'	'pcdu.update(pcdu)'
18	23	12	13	'M'	'pcdu'	'P'	'BatPower'	'BatPower = pcdu.getBatPower();'
19	24	13	14	'P'	'BatPower'	'M'	'battery'	'battery.setBatPower(BatPower);'
20	25	14	14	'M'	'battery'	'M'	'battery'	'battery.update(battery)'
21	12	[]	[]	'P'	'cons'	'T'	'testBattDepthInEcli...	'testBattDepthInEclipse.setcons(cons);'
22	13	[]	[]	'P'	'batteryEnergy'	'T'	'testBattDepthInEcli...	'testBattDepthInEclipse.setbatteryEnergy(batteryEnergy);'
23	14	[]	[]	'T'	'testBattDepthI...	'T'	'testBattDepthInEcli...	'testBattDepthInEclipse.update(testBattDepthInEclipse);'
24	15	[]	[]	'T'	'testBattDepthI...	'P'	'cons'	'cons = testBattDepthInEclipse.getcons( );'
25	16	[]	[]	'T'	'testBattDepthI...	'P'	'batteryEnergy'	'batteryEnergy = testBattDepthInEclipse.getbatteryEnergy( );'

A figura apresenta testes inseridos no final da Matriz de Sequência.

Fonte: Produção do autor.

Figura 5.23 - Matriz de Sequência Reordenada após a Inclusão dos Testes.



Fields	number	l	c	sourceType	sourceName	destinationType	destinationName	expression
1	1	4	4	'M'	'propagator'	'M'	'propagator'	'propagator.update(propagator)'
2	2	4	5	'M'	'propagator'	'P'	'HasSun'	'HasSun = propagator.getHasSun();'
3	3	4	6	'M'	'propagator'	'P'	'CurrentJD'	'CurrentJD = propagator.getCurrentJD();'
4	4	4	7	'M'	'propagator'	'P'	'OrbitPositionKepl'	'OrbitPositionKepl = propagator.getOrbitPositionKepl();'
5	5	3	8	'P'	'Step'	'M'	'solarPanel'	'solarPanel.setStep(Step);'
6	6	5	8	'P'	'HasSun'	'M'	'solarPanel'	'solarPanel.setHasSun(HasSun);'
7	7	7	8	'P'	'OrbitPositionK...	'M'	'solarPanel'	'solarPanel.setOrbitPositionKepl(OrbitPositionKepl);'
8	8	8	8	'M'	'solarPanel'	'M'	'solarPanel'	'solarPanel.update(solarPanel)'
9	9	8	9	'M'	'solarPanel'	'P'	'SagPower'	'SagPower = solarPanel.getSagPower();'
10	10	3	10	'P'	'Step'	'M'	'bus'	'bus.setStep(Step);'
11	11	7	10	'P'	'OrbitPositionK...	'M'	'bus'	'bus.setOrbitPositionKepl(OrbitPositionKepl);'
12	12	[]	[]	'P'	'cons'	'T'	'testBattDepthInEcli...	'testBattDepthInEclipse.setcons(cons);'
13	13	[]	[]	'P'	'batteryEnergy'	'T'	'testBattDepthInEcli...	'testBattDepthInEclipse.setbatteryEnergy(batteryEnergy);'
14	14	[]	[]	'T'	'testBattDepthI...	'T'	'testBattDepthInEcli...	'testBattDepthInEclipse.update(testBattDepthInEclipse);'
15	15	[]	[]	'T'	'testBattDepthI...	'P'	'cons'	'cons = testBattDepthInEclipse.getcons( );'
16	16	[]	[]	'T'	'testBattDepthI...	'P'	'batteryEnergy'	'batteryEnergy = testBattDepthInEclipse.getbatteryEnergy( );'
17	17	10	10	'M'	'bus'	'M'	'bus'	'bus.update(bus)'
18	18	10	11	'M'	'bus'	'P'	'BusLoad'	'BusLoad = bus.getBusLoad();'
19	19	9	12	'P'	'SagPower'	'M'	'pcdu'	'pcdu.setSagPower(SagPower);'
20	20	11	12	'P'	'BusLoad'	'M'	'pcdu'	'pcdu.setBusLoad(BusLoad);'
21	21	15	12	'P'	'BatCharge'	'M'	'pcdu'	'pcdu.setBatCharge(BatCharge);'
22	22	12	12	'M'	'pcdu'	'M'	'pcdu'	'pcdu.update(pcdu)'
23	23	12	13	'M'	'pcdu'	'P'	'BatPower'	'BatPower = pcdu.getBatPower();'
24	24	13	14	'P'	'BatPower'	'M'	'battery'	'battery.setBatPower(BatPower);'
25	25	14	14	'M'	'battery'	'M'	'battery'	'battery.update(battery)'

A figura apresenta a Matriz de Sequência reordenada e com os testes inseridos entre os modelos.

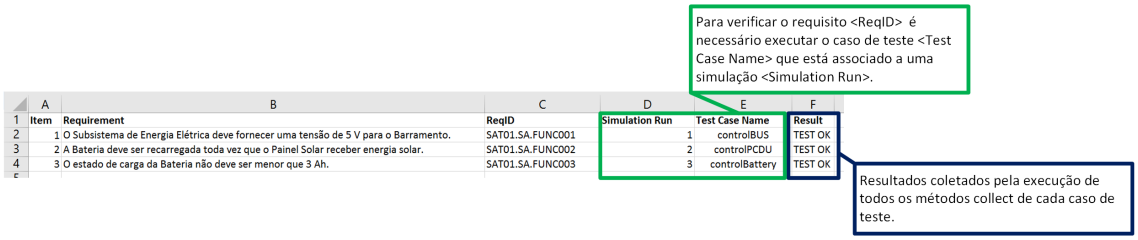
Fonte: Produção do autor.

5.5 Matriz de Teste

Um caso de teste abstrato é executado por uma Função de Tedte, numa associação um-a-um. Após a execução das funções de testes, a Matriz de Teste é preenchida com o resultado da execução do teste, indicando se o requisito foi verificado ou não. Para, a Função de Teste captura os resultados de cada teste através do método "*collect.m*".

Cada Função de teste executada preenche a coluna "*Result*" da Matriz de Teste com as informações do método "*collect.m*". Na Matriz de Teste pode aparecer apenas "OK", um pequeno texto (*string*) com uma explicação, a localização do arquivo de *log*, etc. A Matriz de Teste com os resultados pode ser vista na Figura 5.24.

Figura 5.24 - *Matriz de Teste com os Resultados da Simulação.*



	A	B	C	D	E	F
1	Item	Requirement	ReqID	Simulation Run	Test Case Name	Result
2	1	O Subsistema de Energia Elétrica deve fornecer uma tensão de 5 V para o Barramento.	SAT01.SA.FUNC001	1	controlBUS	TEST OK
3	2	A Bateria deve ser recarregada toda vez que o Painel Solar receber energia solar.	SAT01.SA.FUNC002	2	controlPCDU	TEST OK
4	3	O estado de carga da Bateria não deve ser menor que 3 Ah.	SAT01.SA.FUNC003	3	controlBattery	TEST OK

A figura apresenta a Matriz de Teste preenchida após a execução dos testes, durante uma rodada de simulação.

Fonte: Produção do autor.

A equipe de V&V deve ser responsável por criar todo procedimento específico da função que executa o teste no modelo, assim como o método de coleta que exporta a *string* ("Test OK" e "Test NOK") para a matriz, de forma que, após a verificação, a equipe possa ter o veredicto relativo às conclusões do teste.

5.6 Adaptação para as Diferentes Configurações de Teste

Como descrito na Seção 4.1.4 do Capítulo 4, o *framework* permite a realização dos testes e, consequentemente a validação dos modelos em três configurações, sendo elas: (i) somente *software*, (ii) somente software com o modelo do FEE e (iii) *hardware-in-the-loop*.

Na configuração, na qual os modelos do subsistema a ser validado são todos virtuais (ver Seção 4.1.4.1), basta utilizar o *framework* conforme descrito acima, neste capítulo, não sendo necessárias adaptações.

Quando a configuração que inclui o modelo do equipamento de interface (ver Seção 4.1.4.2) for utilizada, é necessário incluir uma instância na classe "*controller*.m", que represente o modelo do FEE. Assim, toda comunicação entre o ambiente de simulação virtual e o modelo do subsistema em teste deve ser realizada a partir do modelo do FEE, além disso, este modelo deve considerar o tempo de comunicação entre as duas partes, já prevendo o tempo que o *hardware* físico leva para processar as informações e devolvê-las ao ambiente de simulação virtual.

A utilização da configuração que inclui o *hardware* à malha de simulação (Seção 4.1.4.3, do Capítulo 4) também requer adaptações, as quais são descritas na subseção a seguir. Nesta seção descreve-se a adaptação do *framework* para a configuração que inclui o *hardware* à

malha de simulação.

5.6.1 *Hardware-in-the-loop*

O *framework* pode incluir o protótipo do *hardware* para representar um subsistema, ou seja, um artefato físico, além do equipamento de *front-end* (FEE), que realiza a comunicação entre o simulador e o *hardware*.

O ideal para construção de protótipos é a utilização de componentes COTS, uma vez que eles podem ser adquiridos com facilidade, por um custo baixo, e aderente à filosofia dos pico e nanossatélites que também são construídos utilizando este tipo de componentes (SLAVINSKIS et al., 2015).

Durante o desenvolvimento desta dissertação não foi possível utilizar o modelo físico real da PCDU do Tancredo 1, então para fins de demonstração foi construído um protótipo com *Arduino*, para representar este equipamento. Os modos de operação da PCDU foram codificados e carregados para o *Arduino*.

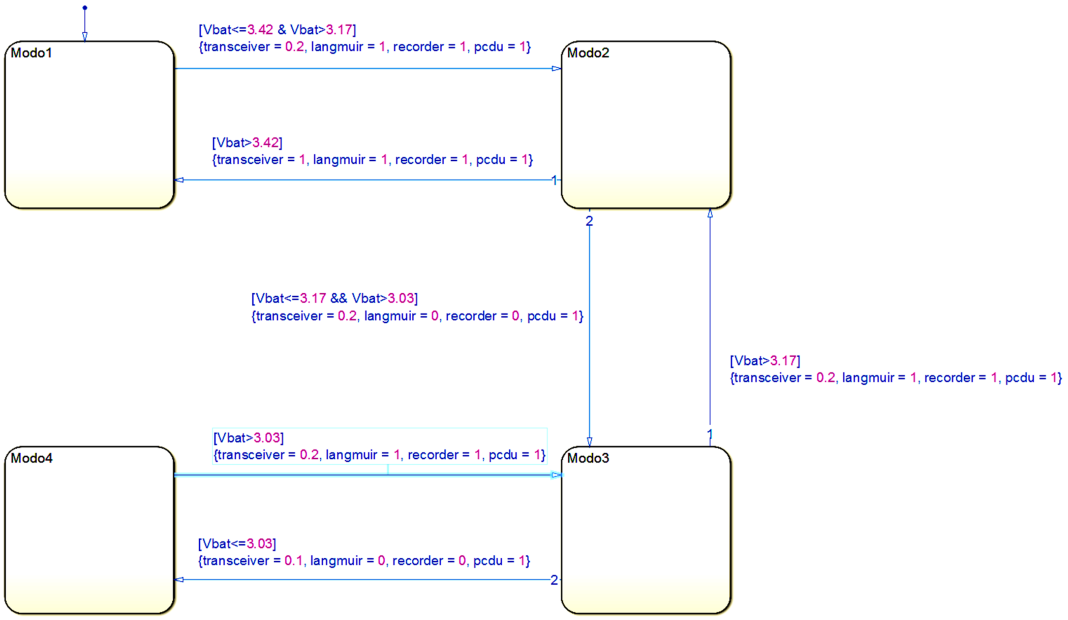
O código do comportamento da PCDU no modelo físico foi gerado automaticamente a partir do modelo em Máquinas de Estados Finitos (MEF) usando o *Simulink*. A Figura 5.25 ilustra os modos de operação da PCDU do Tancredo 1. De acordo com o nível de tensão da bateria, quanto maior a tensão mais equipamentos podem permanecer ligados. O Modo1 indica que a bateria possui carga suficiente para manter os equipamentos ligados e o Modo4 indica que o satélite está em modo crítico, desligando os equipamentos de maior consumo de energia.

O *Arduino Uno*, ilustrado na Figura 5.26 foi usado para representar a PCDU do EPS, conectando os outros componentes (bateria e painéis solares, por exemplo) em seus terminais.

O equipamento responsável pela interface entre o simulador virtual e o *hardware*, o FEE, foi um *Arduino Mega*, como pode ser observado na Figura 5.27. O FEE recebe informações da simulação virtual, e então estimula o *hardware* do EPS, e também é responsável por monitorar as grandezas físicas do *hardware*, coletá-las e enviá-las à parte virtual da simulação. Por possuir mais pinos (analógicos e digitais) que o *Arduino Uno*, o *Arduino Mega* permite o monitoramento de mais componentes.

A adaptação para a Configuração com *hardware-in-the-loop* através do uso do *Arduino*, permitiu demonstrar o uso de um FEE para estabelecer a comunicação entre o ambiente de simulação virtual e o modelo físico do subsistema.

Figura 5.25 - Modos de Operação da PCDU do Tancredo 1.



A figura ilustra a MEF dos modos de operação da PCDU do Tancredo 1.
Fonte: Produção do autor.

5.6.2 Caso de Teste para Modelo Físico

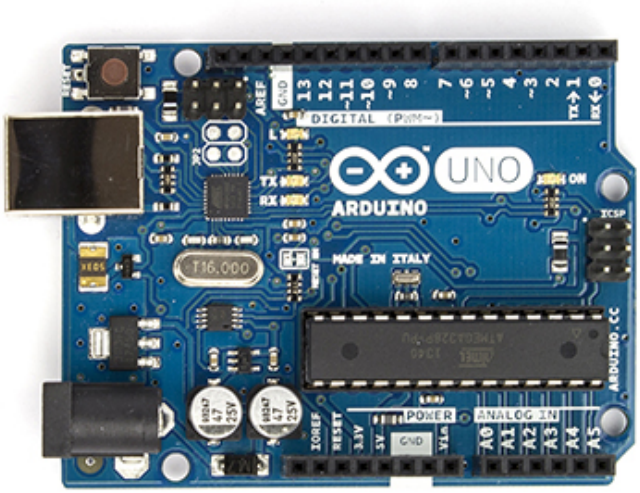
A diferença entre um caso de teste realizado no ambiente de simulação virtual e o caso de teste realizado no *hardware* associado à malha de simulação é que neste segundo, deve-se considerar o equipamento de interface (FEE), de modo a permitir a troca de informação entre o ambiente virtual e o ambiente físico.

Para verificar a troca dos modos de operação o modelo da PCDU foi excitado com diferentes níveis de tensão, dessa forma percebeu-se que o consumo diminuía toda vez que o nível de tensão abaixava, obedecendo o modelo, conforme pode ser observado na Figura 5.28.

Quando a configuração para execução dos testes envolve *hardware-in-the-loop*, algumas modificações na classe "*controller*", e nos métodos "*setup*" e "*update*" são também necessárias necessárias.

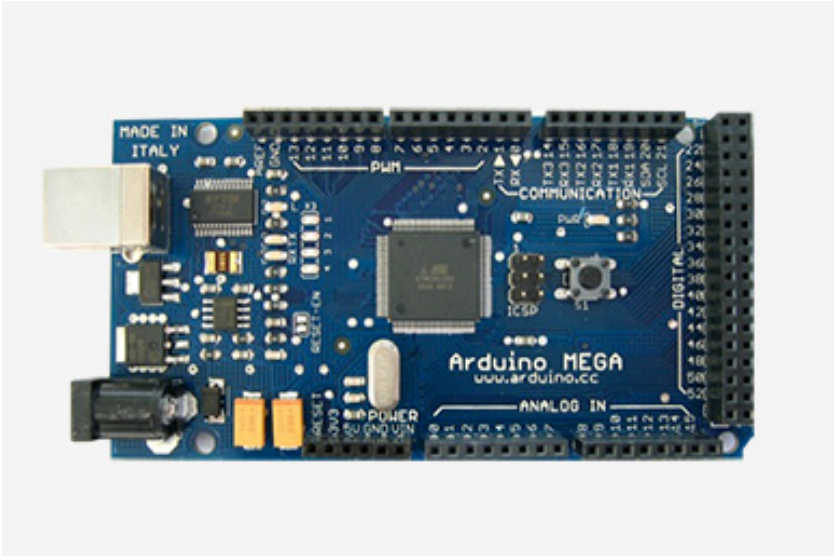
Na classe "*controller*" é necessário fazer a chamada da instância do equipamento de *interface*, neste caso, o *Arduino Mega*, como pode ser observado na Figura 5.29. A função `normalize` é necessária para padronizar os valores, uma vez que, as informações analógicas do *Arduino* vão de 0 a 255 (0 a 5 V).

Figura 5.26 - *Arduino Uno*.



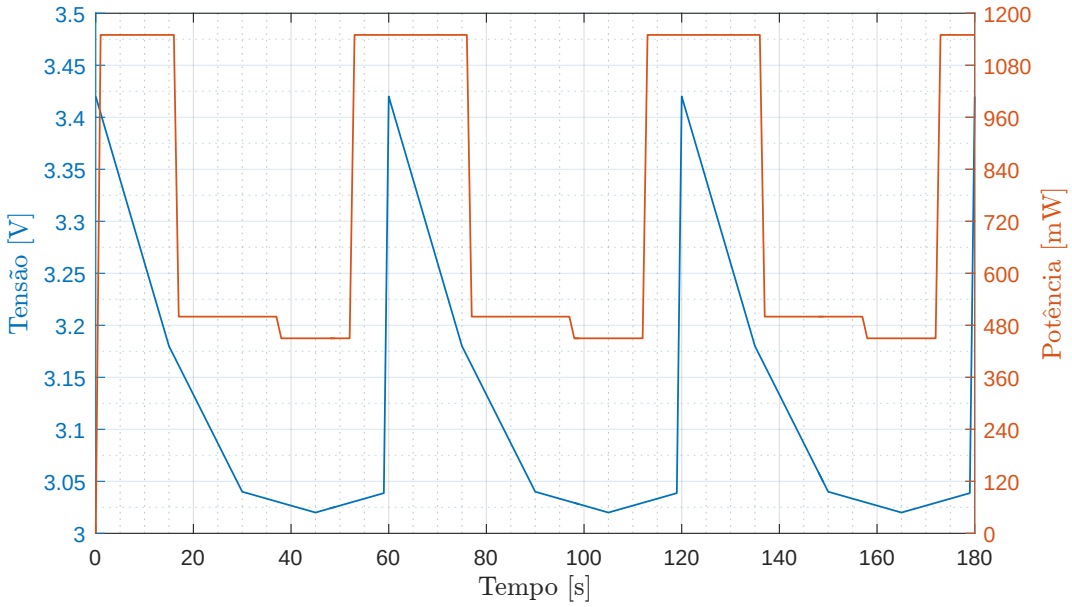
Vista superior da placa *Arduino Uno*.
Fonte: ARDUINO (2015b)

Figura 5.27 - *Arduino Mega*.



Vista superior da placa *Arduino Mega*.
Fonte: ARDUINO (2015a)

Figura 5.28 - Troca entre os Modos de Operação.



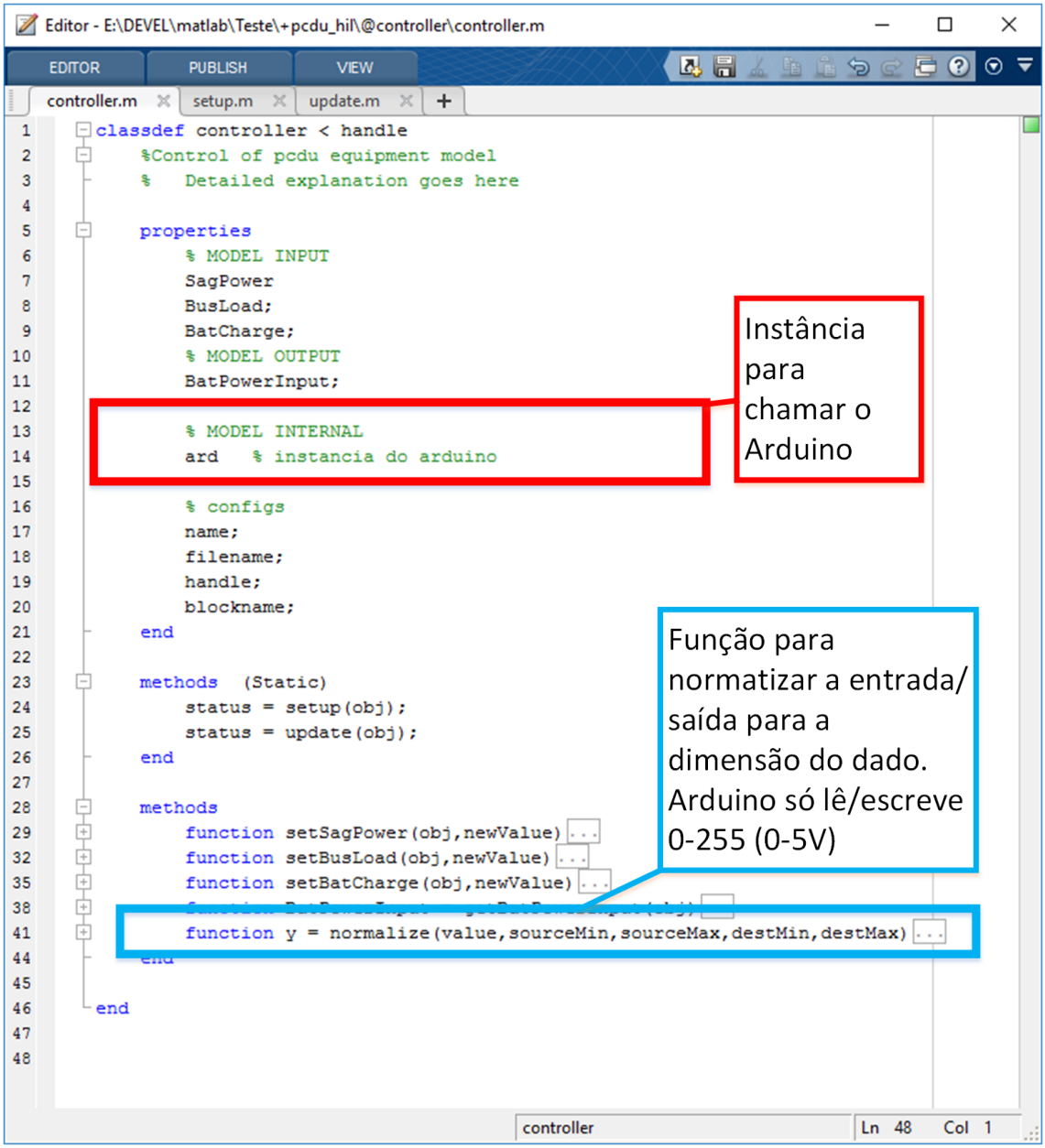
A figura ilustra a resposta do consumo de potência do Tancredo 1 frente aos níveis de tensão.

Fonte: Produção do autor.

No método "*setup*" o equipamento de interface é instanciado conforme ilustra a Figura 5.30.

No método "*update*" são colocados os parâmetros que devem ser enviados e lidos do equipamento de interface, como podem ser observados na Figura 5.31. Assim, um parâmetro é enviado para o equipamento de interface, de modo que o equipamento em teste receba o comando. Então, o equipamento em teste, neste caso a PCDU, executa sua lógica interna, devolve os valores ao equipamento de interface, que por sua vez, devolve ao ambiente de simulação virtual, como ilustra a Figura 5.32

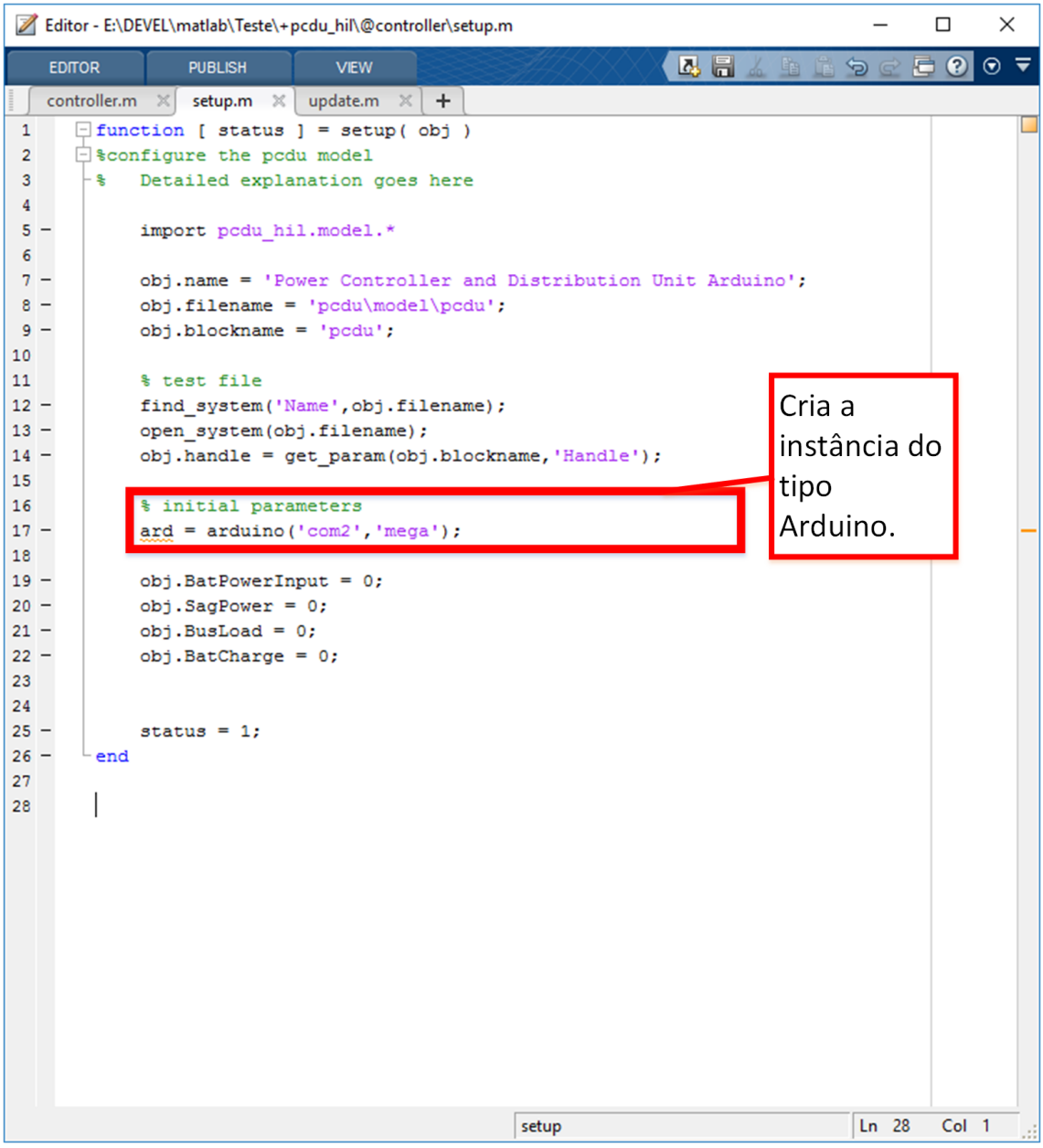
Figura 5.29 - Classe "*controller*" com HIL.



A figura apresenta a classe "*controller*" quando os testes são aplicados ao *hardware*.

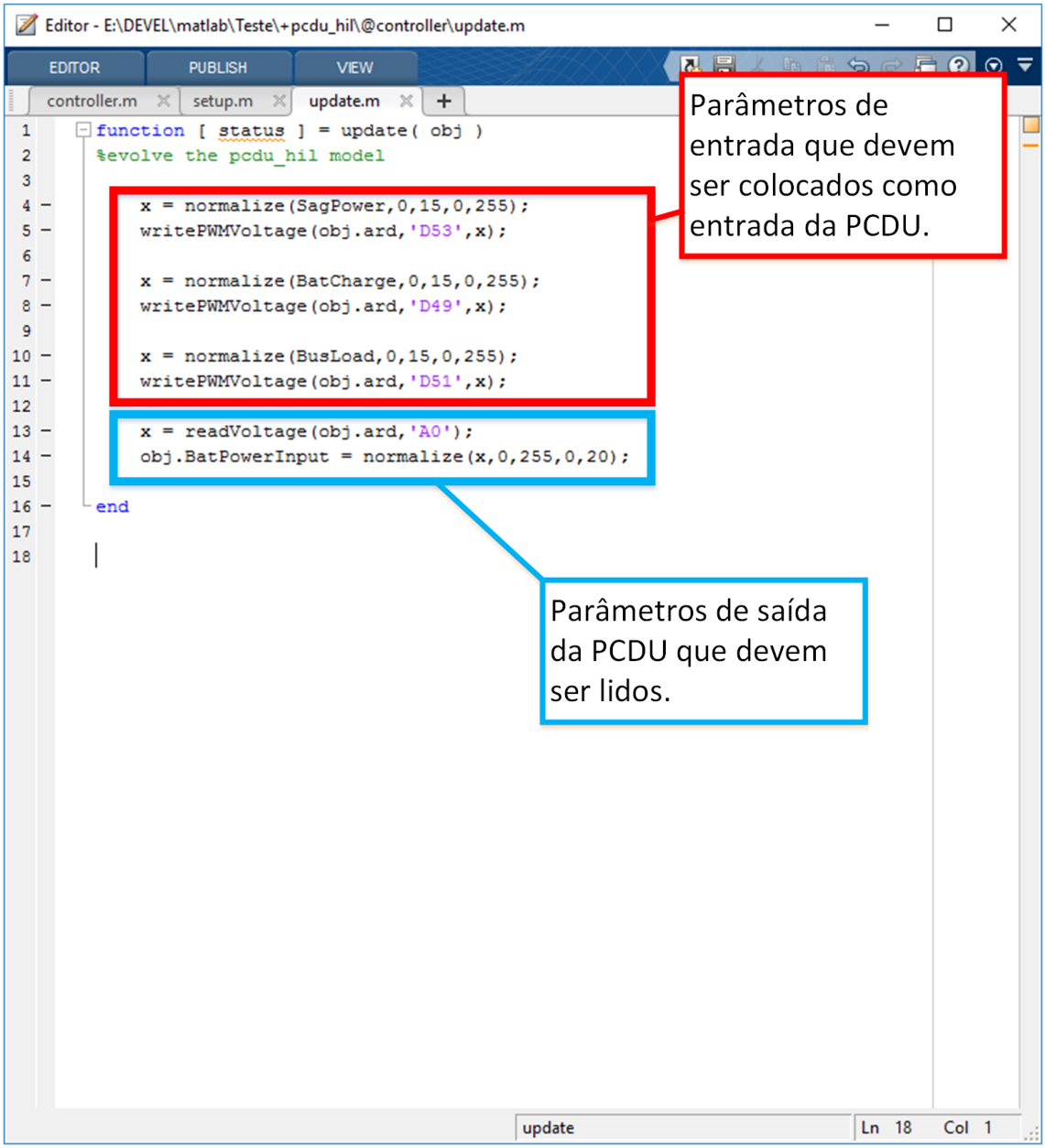
Fonte: Produção do autor.

Figura 5.30 - Método "*setup*" com HIL.



A figura apresenta o método "*setup*" quando os testes são aplicados ao *hardware*.
Fonte: Produção do autor.

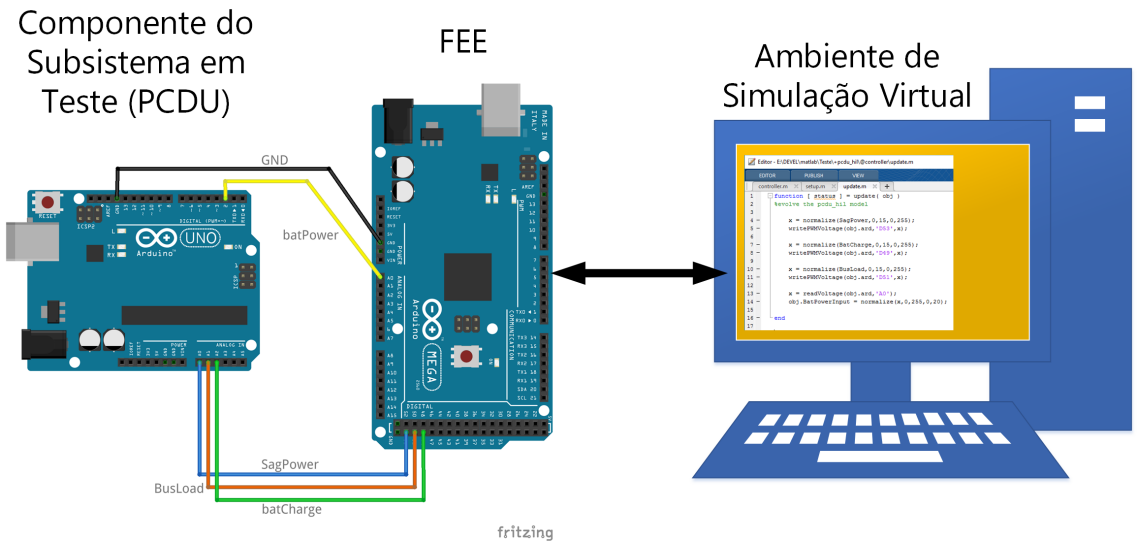
Figura 5.31 - Método "*update*" com HIL.



A figura apresenta o método "*update*" quando os testes são aplicados ao *hardware*.

Fonte: Produção do autor.

Figura 5.32 - Visão geral da Configuração *Hardware-in-the-loop*.



A figura apresenta a visão geral da configuração que inclui o *hardware* à malha de simulação.

Fonte: Produção do autor.

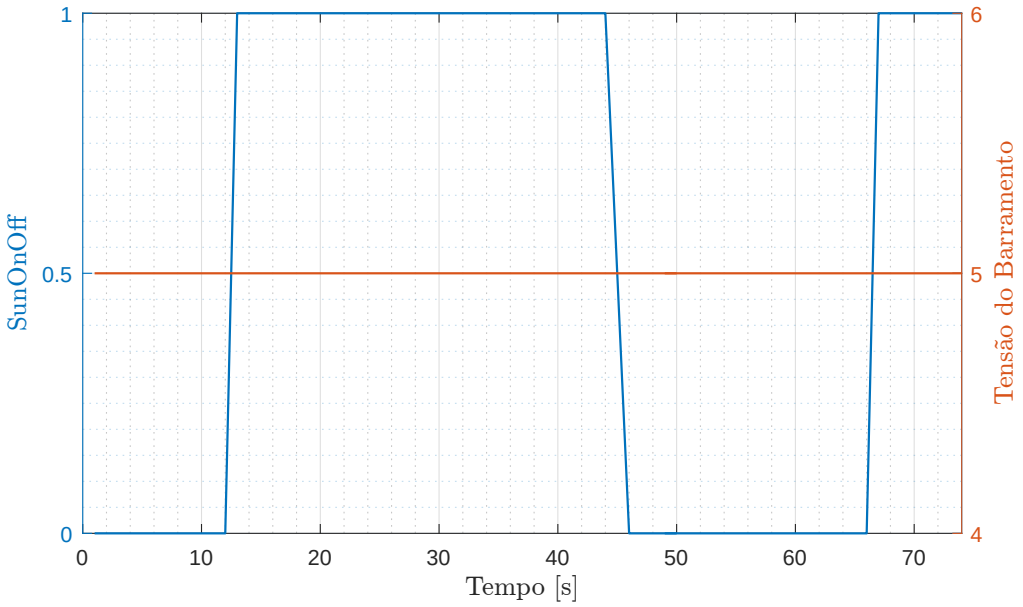
5.7 Resultados da Execução Automática

Esta seção tem por objetivo apresentar os resultados da simulação e dos casos de teste para a verificação dos três requisitos apresentados na Tabela 5.1, consequentemente, a validação do modelo utilizado. Vale ressaltar que os requisitos, para este caso, foram extraídos a partir da documentação do EPS do Tancredo 1.

Os resultados foram obtidos seguindo as características do *framework*, descritas no Capítulo 4 e nas seções anteriores do Capítulo 5, utilizando a configuração de teste onde todos os modelos são virtuais (ver Seção 4.1.4.1).

O primeiro requisito a ser verificado está enunciado da seguinte maneira: “O Subsistema de Energia Elétrica deve fornecer uma tensão de 5 V para o Barramento”. Neste caso, deve-se observar a curva de tensão, sendo que esta informação é coletada nas interfaces do modelo do barramento. A Figura 5.33 apresenta os resultados da execução dos testes, demonstrando que durante todo o período o EPS forneceu a tensão de 5 V para o barramento. E, a Figura 5.34 apresenta a Matriz de Teste preenchida com a informação de que o requisito foi verificado.

Figura 5.33 - Resultados da Verificação do Requisito 1.



As curvas apresentam a incidência de Sol e a tensão no barramento.
Fonte: Produção do autor.

Figura 5.34 - Matriz de Verificação do Requisito 1.

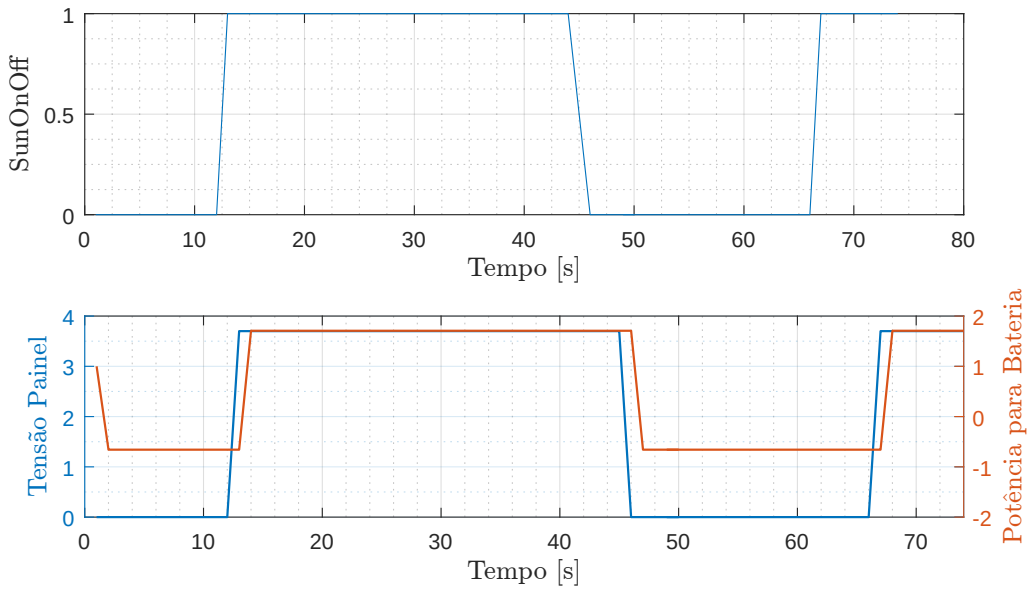
A	B	C	D	E	F
Item	**Requirement**	**ReqID**	**Simulation Run**	**Test Case Name**	**Result**
1	O Subsistema de Energia Elétrica deve fornecer uma tensão de 5 V para o Barramento.	SAT01.SA.FUNC001	1	controlBUS	TEST OK

A informação “TEST OK” indica que o requisito foi verificado.
Fonte: Produção do autor.

O enunciado do segundo requisito diz que: “A Bateria deve ser recarregada toda vez que o Painel Solar receber energia solar”, assim, deve-se observar se o painel solar está recebendo a energia solar, no período em que o satélite está iluminado pelo Sol, e se a bateria realmente está sendo carregada. Essas informações podem ser observadas nos terminais de entrada da PCDU, uma vez que, os sensores, tanto dos painéis solares, quanto

dos terminais da bateria, enviam informação à PCDU. Na parte superior da Figura 5.35, observa-se a curva que indica a incidência de Sol, dessa forma, quando o satélite estiver iluminado, os painéis devem transformar a energia solar em energia elétrica e as baterias devem ser recarregadas. As curvas das grandezas físicas do painel solar e da bateria podem ser observadas na parte inferior da Figura 5.35.

Figura 5.35 - Resultados da Verificação do Requisito 2.



Na parte superior observa-se a curva de incidência de Sol e na parte inferior as grandes físicas relacionadas ao painel solar e à bateria.

Fonte: Produção do autor.

Como pode ser observado na Figura 5.35 o requisito foi verificado, uma vez que a bateria foi recarregada durante o período de incidência solar. A Figura 5.36 indica a Matriz de Teste preenchida com a informação de que o requisito foi verificado.

Figura 5.36 - Matriz de Verificação do Requisito 2.

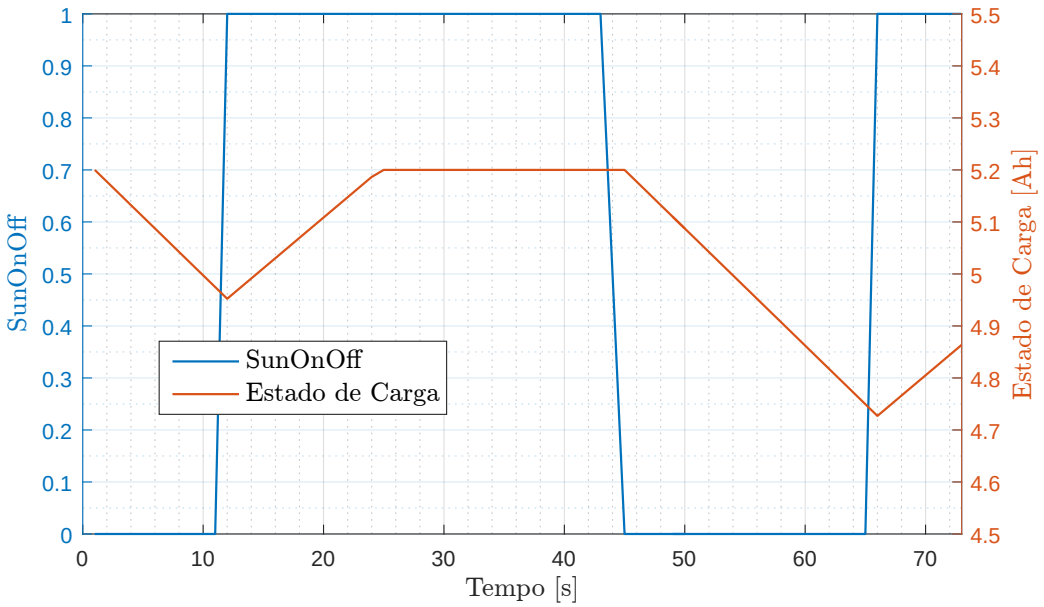
A	B	C	D	E	F
Item	**Requirement**	**ReqID**	**Simulation Run**	**Test Case Name**	**Result**
2	A Bateria deve ser recarregada toda vez que o Painel Solar receber energia solar.	SAT01.SA.FUNC002	2	controlPCDU	TEST OK

A informação “TEST OK” indica que o requisito foi verificado.

Fonte: Produção do autor.

O último requisito listado na Tabela 5.1 estabelece que “O estado de carga da Bateria não deve ser menor 3 Ah”. Neste caso, deve-se observar o comportamento da bateria durante o período de incidência do Sol e, sobretudo, durante o eclipse, período no qual a descarga da bateria é maior. Na Figura 5.37 é possível observar a curva de incidência do Sol e a curva com o estado de carga da bateria, sendo que está última, indica que o estado de carga da bateria durante o eclipse não é menor que 3 Ah, dessa forma, o requisito pode ser verificado, como ilustra a Figura 5.38.

Figura 5.37 - Resultados da Verificação do Requisito 3.



As curvas apresentam a incidência de Sol e o estado de carga da bateria.

Fonte: Produção do autor.

Figura 5.38 - Matriz de Verificação do Requisito 3.

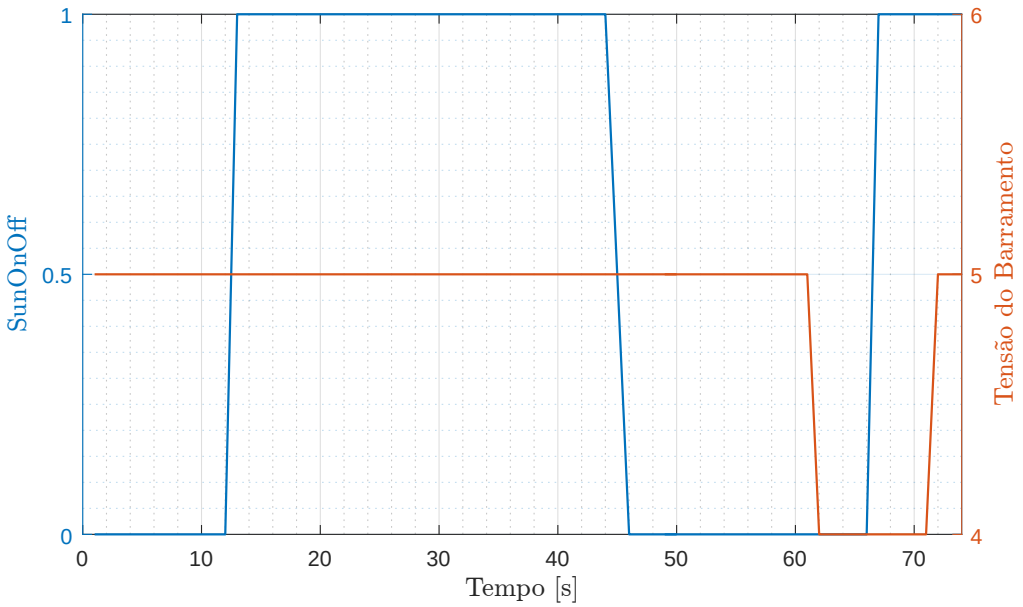
A	B	C	D	E	F
Item	**Requirement**	**ReqID**	**Simulation Run**	**Test Case Name**	**Result**
3	O estado de carga da Bateria não deve ser menor que 3 Ah.	SAT01.SA.FUNC003	3	controlBattery	TEST OK

A informação “TEST OK” indica que o requisito foi verificado.

Fonte: Produção do autor.

Para exemplificar um requisito que não está em conformidade, foi criado uma situação na qual o modelo apresenta um erro: o EPS não fornece a tensão de 5 V para o barramento durante todo o período, conforme pode ser observado na Figura 5.39. Dessa forma, o primeiro requisito listado na Tabela 5.1 apresenta a informação “TEST NOK”, indicando que o modelo não satisfaz requisito, como pode ser observado na Figura 5.40.

Figura 5.39 - Resultados da Verificação do Requisito 1, com Erro no Modelo.



As curvas apresentam a incidência de Sol e a tensão no barramento.

Fonte: Produção do autor.

Figura 5.40 - Matriz de Verificação do Requisito 1, com Erro no Modelo.

1	Item	Requirement	ReqID	Simulation Run	Test Case Name	Result
2	1	O Subsistema de Energia Elétrica deve fornecer uma tensão de 5 V para o Barramento.	SAT01.SA.FUNC001	1	controlBus	TEST NOK

A informação “TEST NOK” indica que o requisito foi verificado, contudo não está em conformidade com o modelo.

Fonte: Produção do autor.

Vale ressaltar que os requisitos também poderiam ter sido verificados nas outras configurações de teste propostas, frente as adaptações necessárias, o que também acarretaria a validação dos modelos do subsistema. Ainda que não tenha sido possível realizar os testes no modelo final do subsistema de energia do Tancredo 1, pode-se dizer que é possível aplicar os testes, devido à flexibilidade do FEE integrado a outros equipamentos (RODRIGUES; AMBROSIO, 2015).

5.8 Limitações e Lições Aprendidas

No decorrer deste trabalho percebeu-se que o desenvolvimento de pico e nanossatélites ainda não é muito bem definido, isto é, seu desenvolvimento não se compromete com os processos da Engenharia de Sistemas, dessa forma, as atividades são realizadas de modo a atender o cronograma do projeto, que normalmente é curto, deixando de lado a aspectos relacionados à documentação de requisitos e de testes.

Assim, na aplicação no *framework* foram encontradas dificuldades em relação aos requisitos do EPS do Tancredo 1, uma vez que não havia um documento com as especificações do satélite que fosse possível capturar todas as informações. Dessa forma, para estabelecer os requisitos, fez-se necessário o conhecer o funcionamento do satélite. Todavia, a análise *bottom-up* pode prejudicar a verificação funcional, pois o levantamento das informações pode acabar desconsiderando algum dado relevante relacionado ao projeto.

Recomenda-se então, que todo o desenvolvimento esteja em afinidade com os processos definidos (NASA, 2007) ou ainda se utilize processos adaptados para pico e nanossatélites (COSTA, 2015), e que estabeleça-se todos os requisitos necessários para a concepção de um sistema que atingir sua missão com êxito. Com os requisitos funcionais bem definidos, a equipe de verificação e validação que utilizará o *framework* poderá cobrir mais casos de teste e revelar informações importantes já nas fases iniciais do desenvolvimento do subsistema, como por exemplo, antecipação de falhas.

Durante o desenvolvimento do trabalho não foram observados todos os testes que são realizados no Subsistema de Energia Elétrica de pico e nanossatélites, assim, não é possível afirmar se o *framework* cobriria todos os casos de testes necessários para validar os modelos que descrevem o subsistema e verificar todos os requisitos. Na terceira configuração de teste, com o *hardware-in-the-loop*, por exemplo, pode haver um caso de teste onde não seja possível representar alguma grande física com o FEE. Observa-se que foi possível verificar os requisitos elencados na Tabela 5.1.

6 CONCLUSÃO

Esta dissertação apresentou um *framework* para validação de um subsistema virtual de energia elétrica para pico e nanossatélites baseadas em simulação, bem como a verificação funcional de seus requisitos funcionais. O *framework* é baseado em simulação. Onde tradicionalmente são executadas simulações e posteriormente análises dos *logs* gerados, aqui propomos que casos de testes para verificação funcional sejam executados em conjunto com modelos do subsistema em um ambiente de simulação. Automaticamente após a execução das simulações, os resultados provenientes dos testes são coletados para informar o *status* dos pontos avaliados durante a rodada de simulação. O *framework* explorou a combinação de simulação e execução automática de casos de teste, estendendo o poder da simulação como ferramenta de verificação e validação.

O *framework* apresentado possibilita o acoplamento de modelos físicos em *hardwares*, associando-os à simulação. Dessa forma, é possível testar tanto os modelos lógicos virtuais quanto os modelos híbridos, que pode incluir os modelos virtuais e partes do modelo físico - protótipos.

Como estudo de caso, foi utilizado o modelo do subsistema de energia elétrica do picossatélite Tancredo 1. Contudo, as características do *framework*, tais como, dado o encapsulamento dos modelos específicos, o cerne que faz a inicialização e evolução dos modelos, este *framework* poderá ser aplicado para verificar diferentes subsistemas de energia elétrica de pico ou nanossatélites, bem como, outros subsistemas.

6.1 Contribuições

A principal contribuição desta dissertação foi definir um *framework* processo de verificação de subsistema de satélites miniaturizados, sobretudo aos pico e nanossatélites, que vêm desempenhando um papel crescente na área espacial, principalmente, com a formação de mão-de-obra e validação de tecnologia no espaço.

O *framework* estabeleceu etapas que devem ser seguidas, de modo a descrever a interface que encapsula os modelos e os arquivos de entrada que podem melhorar o processo de verificação do subsistema de energia elétrica, os quais correspondem a 17% das falhas em satélites miniaturizados. As atividades, no contexto do *framework*, são: criação da matriz de testes, criação do plano de verificação, modelagem do subsistema, definição da configuração de execução dos testes; criação da matriz de sequência, criação da função de teste, execução dos testes, análise dos resultados e veredicto de teste.

No que se refere à modelagem do subsistema (ou à criação dos modelos), sugere-se que os componentes do subsistema sejam modelados separadamente, seguindo o conjunto de regras definidas, permitindo assim, o reuso de um ou mais componentes em fases mais

avançadas do desenvolvimento, ou ainda, em outros projetos, com poucas modificações no modelo.

As três configurações de execução dos testes (somente *software*, *software* com modelo de interfaces, *hardware-in-the-loop*) são fundamentais para o amadurecimento em relação à solução candidata a ser utilizada no subsistema. Em se tratando de novas soluções, desenvolvidas pela própria equipe do projeto, recomenda-se o uso de todas as configurações, para se executar uma verificação mais ampla. Todavia, se a solução COTS for adquirida de terceiros, cabe à equipe definir qual a melhor estratégia para realizar a verificação funcional.

O mecanismo definido para interpretação da Matriz de Sequência pode ser aplicado para qualquer subsistema em teste, desde que sejam respeitadas as regras estabelecidas para este artefato.

A execução automática de testes pode endereçar diversos benefícios, como a repetibilidade do teste e melhoria na produtividade. Além disso, o reuso dos casos de teste justifica o tempo gasto para implementá-los.

Os resultados demonstraram a facilidade no uso da ferramenta implementada no contexto do *framework*, para a validação dos modelos e verificação dos requisitos. Além disso, foram executados alguns testes, que permitiram apresentar o uso prático do *framework*, dessa forma, possibilitando a validação dos modelos de alguns componentes do subsistema de energia elétrica do Tancredo 1, sendo que este modelos foram obtidos a partir de seus diagramas elétricos.

Nesta dissertação foi utilizado o *Arduino* como ferramenta de interface entre o ambiente de simulação e o *hardware* do subsistema em teste do pico ou nanossatélite, todavia o *Arduino* possui algumas limitações, por exemplo, fornecimento de tensão e pontos disponíveis para observação de grandezas físicas, dessa forma, faz-se necessário a substituição do FEE, em se tratando de projetos de satélites maiores.

O programa de computador que implementa o mecanismo da Matriz de Sequência e execução das Funções de Testes foi implementado com o conceito de metamodelos, assim, não seriam necessárias grandes modificações e poderão ser facilmente reutilizados.

6.2 Trabalhos Futuros

Os programas implementados para demonstração das capacidades do *framework*, foram suficientes para exemplificar sua aplicação e funcionalidade. Entretanto é possível estabelecer melhorias de modo a simplificar ainda mais seu uso, como por exemplo, a criação de um arquivo executável com uma interface gráfica. Nesta solução o código só poderia ser

modificado pelo responsável, e seu uso pela Equipe de Verificação e Validação seria ainda mais fácil. Na atual versão da ferramenta, o código está acessível, permitindo que qualquer pessoa realize modificações. Além disso, por não haver uma interface gráfica, é necessário executar os códigos dentro do ambiente MATLAB para executar a validação dos modelos.

De modo a prover melhorias no *framework* e estabelecer mais estudos, no que diz respeito à verificação e validação de pico e nanossatélites, destacam-se alguns trabalhos futuros que podem ser realizados:

- Aplicar o *framework* para o EPS de um pico ou nanossatélite mais complexo, que contenham um conjunto de requisitos maior;
- Aplicar o *framework* para outros subsistemas de pico ou nanossatélites, como por exemplo, o subsistema de comunicação que representa 27% de falhas nesta classe de satélites;
- Adaptar o *framework* para subsistemas de satélites convencionais;
- Estender o *framework* para outras fases do ciclo de desenvolvimento de um satélite;
- Acoplar ferramentas apropriadas para geração automática dos casos de teste, de forma que uma parcela dos casos de teste possa ser gerada automaticamente a partir do modelo comportamental (MEF) do subsistem em teste;
- Permitir a ligação de componentes dos subsistemas provenientes de outros *software*, tais como, *Excel*, *LabView*, programas em C++, que possam ser interligados pelo *framework*, criando um ambiente de *co-simulation*;
- Realizar a transformação automática de uma linguagem de descrição de modelos, como OPM (*Object Process Methodology*), para a Matriz de Sequência, assim como os outros elementos de entrada, de forma a facilitar a abstração de uso para o engenheiro especialista no subsistema e o engenheiro de teste.

REFERÊNCIAS BIBLIOGRÁFICAS

ALMEIDA, E. R. C. **Geração automática de casos de testes executáveis a partir de casos de teste abstratos para aplicações web**. 112 p. Dissertação (Mestrado) — Universidade Estadual de Campinas (UNICAMP), Campinas, 2012-05-21 2004. Disponível em: <http://www.bibliotecadigital.unicamp.br/document/?code=000919651>. Acesso em: 02 dez. 2015. 22

ALMINDE, L.; BISGAARD, M.; GUDMUNDSSON, F.; KEJSER, C.; KOUSTRUP, T.; LODBERG, C.; VISCOR, T. **Power Supply for the AAU CubeSat**. Fredrik Bajers Vej 5, 9100 Aalborg, Dinamarca: [s.n.], fev. 2001. Report. Disponível em: <http://www.space.aau.dk/cubesat/dokumenter/psu.pdf>. Acesso em: 21 nov. 2015. 30

ALMINDE, L.; BISGAARD, M.; MELVILLE, N.; SCHAEFER, J. The sseti-express mission: from idea to launch in one and a half year. In: RECENT ADVANCES IN SPACE TECHNOLOGIES, 2005. **Proceedings...** [S.l.]: IEEE, 2005. p. 100–105. 30

ALVES, P. G.; RODRIGUES, I. P.; JORGE, J. M.; OLIVEIRA, K. F. de. Identificação de um laminador de encruamento em malha fechada através de métodos de subespaços. In: COLÓQUIO TÉCNICO CINETÓFICO, 7, 22-23 out. 2013. **Anais...** Volta Redonda: Editora FOA, 2013. p. 385–386. ISBN 978-85-60144-63-1. Disponível em: <http://web.unifoa.edu.br/editorafoa/wp-content/uploads/2014/06/vii-coloquio-unifoa-20131.pdf>. Acesso em: 20 nov. 2015. 60

AMBROSIO, A. M.; MARTINS, E.; VIJAYKUMAR, N. L.; CARVALHO, S. V. d. Systematic generation of test and fault cases for space application validation. In: DATA SYSTEM IN AEROESPACE CONFERENCE, 9. (DASIA)., 30 may - 2 june, Edinburg, Scotland. **Proceedings...** INPE, 2005. p. 12. Disponível em: <http://urlib.net/sid.inpe.br/iris@1916/2005/11.21.18.43>. Acesso em: 08 dez. 2015. 22

ARDUINO. **Arduino - ArduinoBoardMega2560**. 2015. Disponível em: <https://www.arduino.cc/en/Main/ArduinoBoardMega2560>. Acesso em: 25 nov. 2015. 85

_____. **Arduino - ArduinoBoardUno**. 2015. Disponível em: <https://www.arduino.cc/en/Main/ArduinoBoardUno>. Acesso em: 25 nov. 2015. 85

AZEVEDO, D. N. R. **Benchmarking de resiliência para infraestruturas de simuladores de satélites baseadas em HLA**. 293 p. Tese (Doutorado) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2014-11-28 2014.

Disponível em: <http://urlib.net/sid.inpe.br/mtc-m21b/2014/10.28.22.44>. Acesso em: 12 fev. 2016. 7

BAUER, P. Computer simulation of satellite electric power systems. **IEEE Transactions on Aerospace and Electronic Systems**, AES-5, n. 6, p. 934–942, Nov 1969. ISSN 0018-9251. 33

BIZARRIA, C.; HOFFMANN, C. T.; LOUREIRO, G.; BIZARRIA, J. W. et al. Simulator energy produced by photovoltaic panels used in the validation of power balance in cubesats. **Global Journal of Researches In Engineering**, v. 14, n. 3, 2014. ISSN 2249-4596. 31

BÜRGER, E. E. **Proposta de método para AIT de pico e nanossatélites**. 306 p. Dissertação (Mestrado) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2014-02-24 2014. Disponível em: <http://urlib.net/sid.inpe.br/mtc-m21b/2014/02.13.16.21>. Acesso em: 12 nov. 2015. 1, 3, 31

BÜRGER, E. E.; LACAVA, P.; LOUREIRO, G.; HOFFMANN, C. T.; PEREIRA, M. O. Lessons learned by the first brazilian CubeSat platform. In: INTERNATIONAL ASTRONAUTICAL CONGRESS, 66. (IAC), 12-16 Oct., Jerusalem, Israel. **Proceedings...** Jerusalem, 2015. Acesso em: 11 nov. 2015. 10

BURT, R. **Distributed electrical power system in Cubesat Applications**. 84 p. Dissertação (Mestrado) — Utah State University, Lohan, 2011. Disponível em: <http://www.ece.usu.edu/grad/reports_theses_disseratations/2011/Burt_Robert/thesis.pdf>. Acesso em: 12 nov. 2015. 23, 26, 28

BURT, R. Distributed EPS in a CubeSat application. In: ANNUAL AIAA/USU CONFERENCE ON SMALL SATELLITES, 25, Logan, Utah, USA. **Proceedings...** Logan, 2011. Disponível em: <http://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=1136&context=smallsat>. Acesso em: 11 nov. 2015. 27, 28

CARVALHO, M. J. M. d.; LIMA, J. S. d. S.; JOTHA, L. d. S.; AQUINO, P. S. d. CONASAT - constelação de nano satélites para coleta de dados ambientais. In: SIMPÓSIO BRASILEIRO DE SENSORIAMENTO REMOTO, 16. (SBSR)., 13-18 abr. 2013, Foz do Iguaçu. **Anais...** São José dos Campos: Instituto Nacional de Pesquisas Espaciais (INPE), 2013. p. 9108–9115. ISBN 978-85-17-00066-9 (Internet) and 978-85-17-00065-2 (DVD). Disponível em: <http://urlib.net/dpi.inpe.br/marte2/2013/05.29.00.50.13>. Acesso em: 11 nov. 2015. 10

CERQUEIRA, C. S.; RODRIGUES, I. P.; MOREIRA, C. J. A.; CARRARA, V.; AMBROSIO, A. M.; KIRNER, C. Utilização de realidade virtual, aumentada e cruzada

em simuladores de satélite no inpe. **Tendências e Técnicas em Realidade Virtual e Aumentada**, v. 5, p. 139–159, 2015. ISSN 2177-6776. Disponível em: <http://urlib.net/sid.inpe.br/plutao/2015/06.01.12.30>. Acesso em: 16 nov. 2015. 60, 141

CHO, M.; MASUI, H.; HATAMURA, T.; DATE, K.; HORII, S.; OBATA, S. Overview of nano-satellite environmental tests standardization project: test campaign and standard draft. In: ANNUAL AIAA/USU CONFERENCE ON SMALL SATELLITES, 26, Logan, Utah, USA. **Proceedings...** Logan, 2012. Disponível em: <http://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=1068&context=smallsat>. Acesso em: 20 nov. 2015. 33

COLOMBO, G.; GRASSELLI, U.; LUCA, A. D.; SPIZZICHINO, A.; FALZINI, S. Satellite power system simulation. **Acta Astronautica**, v. 40, n. 1, p. 41 – 50, 1997. ISSN 0094-5765. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0094576597000222>. Acesso em: 22 nov. 2015. 33, 34

COPELAND, L. **A practitioner's guide to software test design**. Norwood: Artech House, 2004. 300 p. ISBN 978-1-580-53791-9. 22

CORPINO, S.; STESINA, F. Verification of a cubesat via hardware-in-the-loop simulation. **Aerospace and Electronic Systems, IEEE Transactions on**, v. 50, n. 4, p. 2807–2818, October 2014. ISSN 0018-9251. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6978879>. Acesso em: 22 nov. 2015. 3, 32, 33, 35

CORSETTI, A. **Aplicação de metodologias de teste baseado em modelos na verificação e validação de mecanismos de FDIR de sistemas de controle de atitude e órbita**. 171 p. Dissertação (Mestrado) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2014-06-27 2014. Disponível em: <http://urlib.net/sid.inpe.br/mtc-m21b/2014/05.15.18.05>. Acesso em: 22 nov. 2015. 22

COSTA, L. L. **Processo de referência para o desenvolvimento da arquitetura de sistemas de pico e nanosatélites**. 331 p. Dissertação (Mestrado) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2015-04-23 2015. Disponível em: <http://urlib.net/sid.inpe.br/mtc-m21b/2015/04.08.17.42>. Acesso em: 14 nov. 2015. 29, 95

COSTA, L. Z. d.; PIOVESAN, T.; MANICA, T. R.; DURÃO, O. S. C.; SCHUCH, N. J. NANOSATC-BR1 launch process and technical management. In: IAA LATIN AMERICA CUBESAT WORKSHOP, 1, 8-11 dez. 2014, Brasília, Brasil. **Proceedings...**

Brasília, 2014. Disponível em:
<http://urlib.net/sid.inpe.br/mtc-m21b/2014/12.22.17.25>. Acesso em: 11 nov. 2015. 10

COSTA, M. G. **Estratégia de automação em testes: requisitos, arquitetura e acompanhamento de sua implantação**. 102 p. Dissertação (Mestrado) — Universidade Estadual de Campinas (UNICAMP), Campinas, 2004-02-23 2004. Disponível em: <http://www.bibliotecadigital.unicamp.br/document/?code=vtls000389004&fd=y>. Acesso em: 02 dez. 2015. 22

CUTLER, J. W.; SPRINGMANN, J. C.; SPANGELO, S. Initial flight assessment of the radio aurora explorer. In: ANNUAL AIAA/USU CONFERENCE ON SMALL SATELLITES, 25, 2011, Logan, Utah, USA. **Proceedings...** Logan, 2011. Disponível em: <http://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=1137&context=smallsat>. Acesso em: 20 nov. 2015. 30

DELAMARO, M. E.; MALDONADO, J. C.; JINO, M. **Introdução ao teste de software**. 1. ed. Rio de Janeiro: Campus, 2007. 394 p. ISBN 978-85-352-2634-8. 22

DENG, L.; YANG, Z. The application of dsm in concurrent design for space science mission. In: SYSTEMS ENGINEERING AND CONCURRENT ENGINEERING FOR SPACE APPLICATIONS (SECESA), 8-10 oct. 2014, Stuttgart, Germany. **Proceedings...** [S.l.], 2014. 21

DEPARTAMENT OF DEFENSE (DOD). **Modeling and Simulation (M&S) Glossary**. Modeling and Simulation Coordination Office 1901 N. Beauregard St., Suite 500, Alexandria, VA 22311, oct. 2011. Glossary. Disponível em: <http://goo.gl/dSZeeW>. Acesso em: 5 nov. 2015. 16

DSM. **DSMweb.org: Design Structure Matrix (DSM)**. 2009. Disponível em: <http://www.dsmweb.org/>. Acesso em: 04 dez. 2015. 21

_____. _____. 2009. Disponível em: <http://www.dsmweb.org/en/understand-dsm/tutorials-overview/descripton-design-structre.html>. Acesso em: 08 dez. 2015. 21

EICKHOFF, J. **Simulating spacecraft systems**. 1. ed. Heidelberg, Germany: Springer Aerospace Technology, 2009. 352 p. ISBN 978-3-642-01275-4. 1, 18, 20

EUROPEAN COOPERATION FOR SPACE STANDARDIZATION (ECSS). **Space engineering**: Verification. ESTEC, P.O. Box 299, 2200 AG Noordwijk - The Netherlands, mar. 2009. ECSS-E-ST-10-02C. Disponível em: <http://www.ecss.nl/>. Acesso em: 3 nov. 2015. 11, 12, 13, 14, 16, 32, 135

_____. **Space engineering**: System engineering general requirements. ESTEC, P.O. Box 299, 2200 AG Noordwijk - The Netherlands, mar. 2009. ECSS-E-ST-10C. Disponível em: <http://www.ecss.nl/>. Acesso em: 3 nov. 2015. 12

_____. **Space project management**: Project planning and implementation. ESTEC, P.O. Box 299, 2200 AG Noordwijk - The Netherlands, mar. 2009. ECSS-M-ST-10C. Disponível em: <http://www.ecss.nl/>. Acesso em: 3 nov. 2015. 1, 19

_____. **Space engineering**: System modelling and simulation. ESTEC, P.O. Box 299, 2200 AG Noordwijk - The Netherlands, apr. 2010. ECSS-E-TM-10-21A. Disponível em: <http://www.ecss.nl/>. Acesso em: 28 out. 2015. 1, 16, 17, 18, 19, 20, 21, 43

_____. **Space engineering**: Verification guidelines. ESTEC, P.O. Box 299, 2200 AG Noordwijk - The Netherlands, dec. 2010. ECSS-E-HB-10-02A. Disponível em: <http://www.ecss.nl/>. Acesso em: 3 nov. 2015. 14, 16, 17, 20, 32

_____. **ECSS system**: Glossary of terms. ESTEC, P.O. Box 299, 2200 AG Noordwijk - The Netherlands, oct. 2012. ECSS-S-ST-00-01C. Disponível em: <http://www.ecss.nl/>. Acesso em: 28 out. 2015. 7

_____. **Space engineering**: Testing. ESTEC, P.O. Box 299, 2200 AG Noordwijk - The Netherlands, jun. 2012. ECSS-E-ST-10-03C. Disponível em: <http://www.ecss.nl/>. Acesso em: 3 nov. 2015. 12, 32

FARID, H.; EL-KOOSY, M.; EL-SHATER, T.; EL-KOSHAIRY, A.; MAHMOUD, A. Simulation of a LEO satellite electrical power supply subsystem in-orbit operation. In: EUROPEAN PHOTOVOLTAIC SOLAR ENERGY CONFERENCE AND EXHIBITION, 23, 1-5 sep. 2008, Valencia, Spain. **Proceedings...** Valencia, 2008. Disponível em: <http://www.eupvsec-proceedings.com/proceedings?paper=2472>. Acesso em: 20 nov. 2015. 35

FILHO, J. P. de A. **Simulação de um sistema de suprimento de energia de um satélite**. 140 p. Dissertação (Mestrado) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 1983-12-22 1984. Acesso em: 13 nov. 2015. 28, 33

FORTESCUE, P.; SWINERD, G.; STARK, J. **Spacecraft systems engineering**. 4. ed. Chichester, UK: John Wiley and Sons, 2004. 691 p. ISBN 978-0-470-75012-4. 7, 8, 9, 24, 25, 26, 28, 29, 116

FRANKE, L. L. C.; FARIAS, T. T.; BALESTRIN, M. R.; SILVEIRA, I. C.; JAENISCH, G. P.; PIAZ, M. A. D.; DURÃO, O. S. C.; SCHUCH, N. J. Simulação térmica do primeiro Cubesat brasileiro: NANOSATC-BR1. In: CONGRESSO NACIONAL DE ESTUDANTES DE ENGENHARIA MECÂNICA., 2014, Rio de Janeiro, Brasil. **Anais...** Rio de Janeiro, 2014. Disponível em:

<http://urlib.net/sid.inpe.br/plutao/2014/12.01.11.48.23>. Acesso em: 11 nov. 2015. 10

FREIRE, C. F. S. **Estudo de topologias de subsistemas de suprimento de energia de satélites e desenvolvimento de um procedimento de projeto da topologia híbrida**. 238 p. Dissertação (Mestrado) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2009-05-25 2009. Disponível em: <http://urlib.net/rep/8JMKD3MGP8W/34RUHR2?ibiurl.language=pt-BR>. Acesso em: 13 nov. 2015. 26, 27

GENTINA, J.; YAMAGUTI, W.; VAROTTO, S. E. C.; WALDMANN, J.; FERNANDES, D. A proposal for ITASAT satellite configuration and its preliminary mission analysis. In: BRAZILIAN SYMPOSIUM ON AEROSPACE ENGINEERING AND APPLICATIONS, 2009; CTA-DLR WORKSHOP ON DATA ANALYSIS AND FLIGHT CONTROL, 3., 14-16 Set. 2009, São José dos Campos, São Paulo, Brasil. **Proceedings...** São José dos Campos, 2009. v. 3. Disponível em: <http://www.cta-dlr2009.ita.br/Proceedings/PDF/59599.pdf>. Acesso em: 11 nov. 2015. 10

HAYES, L. G. **The automated testing handbook**. 2. ed. São Paulo, Brasil: Novatec, 2012. 151 p. ISBN 978-85-7522-290-4. 22

HEIDT, H.; PUIG-SUARI, J.; MOORE, A.; NAKASUKA, S.; TWIGGS, R. CubeSat: A new generation of picosatellite for education and industry low-cost space experimentation. In: ANNUAL/USU CONFERENCE ON SMALL SATELLITES, 14, 2000, Logan, Utah, USA. **Proceedings...** Logan, 2000. Disponível em: <http://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=2069&context=smallsat>. Acesso em: 11 nov. 2015. 9

HOFFMANN, C. T. **Proposta de arquitetura para executar testes em módulos elétricos de CubeSat**. 112 p. Dissertação (Mestrado) — Instituto Técnológico de Aeronáutica (ITA), São José dos Campos, 2015. Disponível em: <http://www2.units.it/atmocube/theses/alimentazione_2_Pendalo.pdf>. Acesso em: 14 nov. 2015. 29, 31

HWANG, J. T. Efficient and scalable computational design of a small satellite. In: ANNUAL AIAA/USU CONFERENCE ON SMALL SATELLITES, 28, Logan, Utah, USA. **Proceedings...** Logan, 2014. Disponível em: <http://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=3099&context=smallsat>. Acesso em: 08 dez. 2015. 21

INTERORBITAL SYSTEMS. **TubeSat 1**. P.O. Box 662 Mojave, Ca 93502-0662, USA: [s.n.], 2015. Disponível em:

<http://www.interorbital.com/interorbital_06222015_002.htm>. Acesso em: 11 nov. 2015. 11

JIANG, Z.; DOUGAL, R. A.; LIU, S. Application of {VTB} in design and testing of satellite electrical power systems. **Journal of Power Sources**, v. 122, n. 1, p. 95 – 108, 2003. ISSN 0378-7753. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0378775303003604>. 34

KAYA, M.; BAYRAKCEKEN, M. Complete electrical model and simulation of a medium size satellite. In: RECENT ADVANCES IN SPACE TECHNOLOGIES, 2011. **Proceedings...** [S.l.], 2011. p. 522–525. 35

LAMY, A. **Set of tools for space flight dynamics mission analysis**. 2012. Apresentação de Slides. Disponível em: <https://www.scilab.org/content/download/420/3050/file/CNES_ALamy_ScilabTEC2012.pdf>. Acesso em: 22 nov. 2015. 36

LOUREIRO, G. **A systems engineering and concurrent engineering framework for the integrated development of complex products**. 606 p. Tese (Doutorado) — Loughborough University, Leicestershire, UK, 1999. Disponível em: <https://dspace.lboro.ac.uk/dspace-jspui/handle/2134/10687>. Acesso em: 25 nov. 2015. 39

MAGALHÃES, R. O. d. **Modelagem, simulação e validação experimental de um sistema de rastreio de potência máxima para geradores solares de satélites artificiais**. 92 p. Dissertação (Mestrado) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2005-08-26 2005. Disponível em: <http://urlib.net/sid.inpe.br/MTC-m13@80/2005/10.25.18.56>. Acesso em: 13 nov. 2015. 24, 25, 27

MARTIN, T. **Use of Scilab for space mission analysys and flight dynamics activities**. 2009. Apresentação de Slides. Disponível em: <https://www.scilab.org/content/download/388/2760/file/TMartin_ScilabTec.pdf>. Acesso em: 22 nov. 2015. 36

MARTINS, E.; GUIMARÃES, D. C. .; AMBROSIO, A. M. Ster: a strategy for testing reactive systems. In: INTERNATIONAL CONFERENCE ON DEPENDABLE SYSTEM AND NETWORK; IEEE CONFERENCE PROCEEDINGS., 28 June - 01 July 2004, Florence, Italy. **Proceedings...** INPE, 2004. p. 2. Disponível em: <http://urlib.net/sid.inpe.br/marciana/2004/12.03.10.21>. Acesso em: 11 dez. 2015. 22

MARTINS, M. S. **Aplicação da DSM no processo de análise de segurança no desenvolvimento de aeronaves comerciais**. 143 p. Dissertação (Mestrado) —

Universidade Estadual de Campinas (UNICAMP), Campinas, 2010-06-14 2010. Disponível em: <http://www.bibliotecadigital.unicamp.br/document/?code=000773169&fd=y>. Acesso em: 03 dez. 2015. 21

MATHWORKS. **MATLAB - The Language of Technical Computing**. 2015. Disponível em: <http://www.mathworks.com/products/matlab/>. Acesso em: 22 nov. 2015. 36

_____. **Simulink - Simulation and Model-Based Design**. 2015. Disponível em: <http://www.mathworks.com/products/simulink/>. Acesso em: 22 nov. 2015. 36

MATLAB. **Search hardware support**. 2015. Disponível em: <http://www.mathworks.com/hardware-support/index.html?suggestion=true&c%5B%5D=hardware_catalog_en&q=raspberry>. Acesso em: 22 nov. 2015. 60

_____. **Why Use Object-Oriented Design - MATLAB & Simulink**. 2015. Disponível em: <http://www.mathworks.com/help/matlab/matlab_oop/why-use-object-oriented-design.html>. Acesso em: 22 nov. 2015. 59

MEHRPARVAR, A.; PIGNATELLI, D.; CARNAHAN, J.; MUNAKATA, R.; LAN, W.; TOORIAN, A.; HUTPUTANASIN, A.; LEE, S. **CubeSat Design Specification Rev. 13**: The cubesat program. San Luis Obispo, CA 93407, United States of America: [s.n.], apr. 2014. Rev. 13. Disponível em: <http://www.cubesat.org/index.php/documents/developers>. Acesso em: 3 nov. 2015. 1, 9

MELONE, C. W. **Preliminary design, simulation, and test of the electrical power subsystem of the tinyscope nanosatellite**. 203 p. Dissertação (Mestrado) — Naval Postgraduate School (NPS), Monterey, California, 2009-12 2009. Disponível em: <http://www.dtic.mil/dtic/tr/fulltext/u2/a514417.pdf>. Acesso em: 14 nov. 2015. 69, 70, 131, 132, 133

MODELICA. **Modelica and the Modelica Association - Modelica Association**. 2015. Disponível em: <https://www.modelica.org/>. Acesso em: 22 nov. 2015. 36

MOURA, C. O. de; TIKAMI, A.; SANTOS, W. A. dos. UbatubaSat - A roadmap from public brazilian schools towards knowledge. In: 30TH ISTS - INTERNATIONAL SYMPOSIUM ON SPACE TECHNOLOGY AND SCIENCE, 4-10 jul. 2015, Kobe, Japan. **Proceedings...** Kobe, 2015. Disponível em: <https://www.researchgate.net/publication/280043259_UbatubaSat__A_Roadmap_from_Public_Brazilian_Schools_Towards_Knowledge>. Acesso em: 20 out. 2015. 115, 116, 118

NAIK, K.; TRIPATHY, P. **Software testing and quality assurance: theory and practice**. Danvers: John Wiley & Sons, 2011. 648 p. ISBN 978-1-118-21163-2. 22

NANOSATC-BR. **NanoSatC-BR**. 2014. Disponível em: <http://www.inpe.br/crs/nanosat/noticias_2014.php#noticia5>. Acesso em: 15 nov. 2015. 30

NARAYANI, L. **The art and the science of test case writing: A guide to be a better software tester!** [S.l.]: CreateSpace Independent Publishing Platform, 2012. 102 p. ISBN 978-1-475-22430-6. 22

NASON, I.; PUIG-SUARI, J.; TWIGGS, R. Development of a family of picosatellite deployers based on the cubesat standard. In: AEROSPACE CONFERENCE, 2002, Big Sky, Montana, USA. **Proceedings...** Big Sky: IEEE, 2002. v. 1, p. 1–457–1–464 vol.1. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1036865>. Acesso em: 22 nov. 2015. 9

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION (NASA). **NASA Systems Engineering Handbook**. NASA Headquarters, Washington, D.C. 20546, dec. 2007. NASA/SP-2007-6105 Rev1. Disponível em: <http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20080008301.pdf>. Acesso em: 3 nov. 2015. 11, 17, 39, 95

NATIONAL INSTRUMENTS. **Multisim: Poderoso software de projeto e ensino de circuitos - National Instruments**. 2015. Disponível em: <http://www.ni.com/multisim/pt/>. Acesso em: 22 nov. 2015. 36

OLIVA, L. L. **Comparação da modelagem e simulação de um subsistema propulsivo da PMM orientada por fluxos físicos e de informação**. 285 p. Dissertação (Mestrado) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2012-11-23 2012. Disponível em: <http://urlib.net/sid.inpe.br/mtc-m19/2012/10.03.00.27>. Acesso em: 12 fev. 2016. 36

OrCAD. **Overview Page - OrCAD PSpice Designer - OrCAD**. 2015. Disponível em: <http://www.orcad.com/products/orcad-pspice-designer/overview>. Acesso em: 22 nov. 2015. 36

PAJUSALU, M.; ILBIS, T. I. E.; VESKE, M.; KALDE, J.; LILLMAA, H.; RANTSUS, R.; PELAKAUSKAS, M.; LEITU, A.; VOORMANSIK, K.; ALLIK, V.; LÄTT, S.; ENVALL, J.; NOORMA, M. Design and pre-flight testing of the electrical power system for the ESTCube-1 nanosatellite. In: **Proceedings of the Estonian Academy of Sciences**. [s.n.], 2014. p. 232–241. Disponível em: <http://www.eap.ee/public/

proceedings_pdf/2014/issue_2S/Proc-2014-2S-222-231.pdf>. Acesso em: 15 nov. 2015. 28, 29, 32

PATEL, M. R. **Spacecraft power systems**. [S.l.]: CRC Press, 2004. 736 p. ISBN 978-0-8493-2786-5. 2, 26, 30

PIAZ, M. A. L. D.; FRANKE, L. L. C.; BALESTRINI, M. R.; SILVEIRA, I. C.; JAENISCH, G. P.; FARIAS, T. T.; DURÃO, O. S. C.; SCHUCH, N. J. Risk analysis comparison between the mission nanosatc-br1 and nanosatc-br2. **Journal of Mechanics Engineering and Automation**, v. 5, 2015. ISSN 2159-5275. Disponível em: <http://urlib.net/sid.inpe.br/mtc-m21b/2015/07.08.17.22>. Acesso em: 11 nov. 2015. 10

PINHEIRO, A. C.; SIMÃO, A.; AMBROSIO, A. M. Fsm-based test case generation methods applied to test the communication software on board the itasat university satellite: A case study. **Journal of Aerospace Technology and Management**, v. 6, n. 4, p. 447–461, 2014. ISSN 1984-9648. Disponível em: <http://urlib.net/sid.inpe.br/mtc-m21b/2015/01.13.18.16>. Acesso em: 22 nov. 2015. 22

PIOVESAN, T.; COSTA, L. Z. d.; MÂNICA, T. R.; DURÃO, O. S. C.; SCHUCH, N. J. The NANOSATC-BR1 electrical power system. In: IAA LATIN AMERICA CUBESAT WORKSHOP, 1, 8-11 dez. 2014, Brasília, Brasil. **Proceedings...** Brasília, 2014. Disponível em: <http://urlib.net/sid.inpe.br/mtc-m21b/2014/12.22.17.38>. Acesso em: 14 nov. 2015. 29

PULECCHI, T.; CASELLA, F.; LOVERA, M. A modelica library for space flight dynamics. In: 5TH INTERNATIONAL MODELICA CONFERENCE, 4-5 sep. 2006, Vienna, Austria. **Proceedings...** 2006. Disponível em: <https://www.modelica.org/events/modelica2006/Proceedings/sessions/Session1d3.pdf>. Acesso em: 22 nov. 2015. 36

QIAO, L.; RYAN, M. Applying the design structure matrix (DSM) to SAR satellite formation flying design. In: SYSTEMS ENGINEERING / TEST AND EVALUATION CONFERENCE (SETE), 27-29 apr. 2015, Canberra. **Proceedings...** 2015. Disponível em: <https://www.researchgate.net/publication/274660086_Applying_the_Design_Structure_Matrix_(DSM)_to_SAR_Satellite_Formation_Flying_Design>. Acesso em: 03 dez. 2015. 21

RIOS, E.; BASTOS, A.; CRISTALLI, R.; MOREIRA, T. **Base de conhecimento em teste de software**. 3. ed. São Paulo: Martin Editora, 2012. ISBN 978-8-580-63053-4. 22

RODRIGUES, I. P.; AMBROSIO, A. M. Simulação com hardware-in-the-loop integrada por Arduino a um simulador de satélite. In: WORKSHOP EM ENGENHARIA E

TECNOLOGIA ESPACIAIS, 2014. (WETE)., 12-14 ago. 2014, São José dos Campos. São José dos Campos: Instituto Nacional de Pesquisas Espaciais (INPE), 2014. Disponível em: <http://urlib.net/sid.inpe.br/mtc-m21b/2014/11.19.18.09>. Acesso em: 22 nov. 2015. 60, 140

_____. Verificação de requisitos através do uso de um simulador funcional. In: WORKSHOP EM ENGENHARIA E TECNOLOGIA ESPACIAIS, 2015. (WETE)., 18-20 ago. 2015, São José dos Campos. São José dos Campos: Instituto Nacional de Pesquisas Espaciais (INPE), 2015. 60, 95, 142

RODRIGUES, I. P.; JORGE, J. M.; OLIVEIRA, K. F. de. **Identificação e controle de um laminador de encruamento em malha fechada através de mÉtodos de subespaços**. Volta Redonda: [s.n.], 2013-11-30 2013. 147 p. Trabalho de Conclusão de Curso. Disponível em: <http://www.crea-rj.org.br/premiocrearjniemeyer/files/2014/10/UNIFOA-Grad-Eng-Eletrica.pdf>. Acesso em: 22 nov. 2015. 60

SANTOS, W. A. D.; MOURA, C. O. de; YAMAGUTI, W.; SILVA, M. M. Q. da. Space education and public outreach for aerospace engineering in a brazilian perspective. In: 28TH ISTS - INTERNATIONAL SYMPOSIUM ON SPACE TECHNOLOGY AND SCIENCE, 5-12 jun. 2011, Okinawa, Japan. **Proceedings...** Okinawa, 2011. Disponível em: <http://archive.ists.or.jp/search/query/1/>. Acesso em: 20 out. 2015. 11, 59, 115

SANTOS, W. A. D.; TIKAMI, A.; DOMINGOS, S.; MOURA, C. O. de; BARBOSA, J. I. M.; MURALIKRISHNA, P. A langmuir probe payload adaptation for CubeSats-TubeSats. In: IAA LATIN AMERICA CUBESAT WORKSHOP, 1, 8-11 dez. 2014, Brasília, Brasil. **Proceedings...** Brasília, 2014. Disponível em: <https://www.researchgate.net/publication/272487010_A_Langmuir_Probe_Payload_Adaptation_for_CubeSats-TubeSats>. Acesso em: 20 out. 2015. 116, 117, 119, 120, 122, 123, 124, 125

SCHOLZ, A.; JUANG, J.-N. Toward open source CubeSat design. **Acta Astronautica**, v. 115, p. 384 – 392, 2015. ISSN 0094-5765. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0094576515002507>. 1, 9

SCILAB. **Scilab**. 2015. Disponível em: <http://www.scilab.org/>. Acesso em: 22 nov. 2015. 36

SLAVINSKIS, A.; PAJUSALU, M.; KUUSTE, H.; ILBIS, E.; EENMAE, T.; SUNTER, I.; LAIZANS, K.; EHRPAIS, H.; LIIAS, P.; KULU, E.; VIRU, J.; KALDE, J.; KVELL, U.; KUTT, J.; ZALITE, K.; KAHN, K.; LATT, S.; ENVALL, J.; TOIVANEN, P.; POLKKO, J.; JANHUNEN, P.; ROSTA, R.; KALVAS, T.; VENDT, R.; ALLIK, V.; NOORMA, M. Estcube-1 in-orbit experience and lessons learned. **Aerospace and**

Electronic Systems Magazine, IEEE, v. 30, n. 8, p. 12–22, Aug 2015. ISSN 0885-8985. 32, 83

SWARTWOUT, M. Attack of the CubeSats: a statiscal look. In: ANNUAL AIAA/USU CONFERENCE ON SMALL SATELLITES, 25, 2011, Logan, Utah, USA. **Proceedings...** Logan, 2011. Disponível em: <http://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=2069&context=smallsat>. Acesso em: 11 nov. 2015. 9

_____. The first one hundred Cubesats: a statistical look. **Journal of Small Satellite**, v. 2, n. 2, p. 213–233, 2013. ISSN 2327-4123. Disponível em: <http://web.csulb.edu/~hill/ee400d/Project%20Folder/CubeSat/The%20First%20One%20Hundred%20Cubesats.pdf>. 2

_____. **CubeSat database**. 1 North Grand Boulevard, Saint Louis, MO 63103, USA, 2015. Disponível em: <https://sites.google.com/a/slu.edu/swartwout/home/cubesat-database>. Acesso em: 11 nov. 2015. 2, 9

THIRION, P. **Design and implementation of on-board electrical power supply of student nanosatellite OUFTI-1 of University of Liège**. 116 p. Dissertação (Mestrado) — University of Liège, Place du 20 Août 7, 4000 Liège, Bélgica, 2009. Disponível em: <http://www.leodium.ulg.ac.be/cmsms/uploads/08-09_Thirion.pdf>. Acesso em: 20 nov. 2015. 28, 29, 31

TIKAMI, A. Re-Engenharia de sistemas em plataforma TubeSat para picossatélites. In: Workshop em Engenharia e Tecnologia Espaciais, 2014. (WETE)., 12-14 ago. 2014, São José dos Campos, São Paulo. **Anais...** São José dos Campos: Instituto Nacional de Pesquisas Espaciais (INPE), 2014. Disponível em: <http://urlib.net/sid.inpe.br/mtc-m21b/2014/11.19.19.34>. Acesso em: 20 out. 2015. 117, 119

TIKAMI, A.; BRITO, A. F. de; FILHO, A. C. J.; MOURA, C. O. de; SANTOS, W. A. dos. Space systems re-engineering and a CubeSat interconnection solution for a TubeSat-based platform. In: LATIN AMERICA CUBESAT IAA WORKSHOP, 1, 8-11 dez. 2014, Brasília, Brasil. **Proceedings...** Brasília, 2014. Acesso em: 20 out. 2015. 29, 116, 117, 119, 120, 121, 122, 125, 127, 128, 129, 130

TIKAMI, A.; SANTOS, W. A. dos. Re-Engineering a picosatellite for a langmuir probe payload. In: WORKSHOP EM ENGENHARIA E TECNOLOGIA ESPACIAIS, 2014. (WETE)., 18-20 ago. 2015, São José dos Campos, São Paulo. **Proceedings...** São José dos Campos: Instituto Nacional de Pesquisas Espaciais (INPE), 2015. Acesso em: 20 out. 2015. 117, 126, 127, 130

TORRES, L. C. G. **Análise do comportamento elétrico dos geradores solares da série de satélites CBERS e a confrontação dos resultados com os dados de projeto**. 201 p. Dissertação (Mestrado) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2014-07-16 2014. Disponível em: <http://urlib.net/sid.inpe.br/mtc-m21b/2014/06.18.14.10>. Acesso em: 13 nov. 2015. 27

ULRICH, S.; VEILLEUX, J. F.; CORBIN, F. L. Power system design of ESMO. **Acta Astronautica**, v. 64, n. 2–3, p. 244–255, 2009. ISSN 0094-5765. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0094576508003032>. 23, 28, 29, 30

WERTZ, J.; LARSON, W. **Space mission analysis and design**. 3. ed. [S.l.]: Springer Netherlands, 1999. (Space Technology Library). ISBN 9780792359012. 7, 8, 23, 24, 25, 26, 28, 29

WIKIPÉDIA. **MATLAB**. 2015. Disponível em: <https://pt.wikipedia.org/wiki/MATLAB>. Acesso em: 22 nov. 2015. 36

WOELLERT, K.; EHRENFREUND, P.; RICCO, A. J.; HERTZFELD, H. Cubesats: Cost-effective science and technology platforms for emerging and developing nations. **Advances in Space Research**, v. 47, n. 4, p. 663 – 684, 2011. ISSN 0273-1177. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0273117710006836>. 10

ZAHRAN, M. In orbit performance of LEO satellite electrical power subsystem - SW package for modelling and simulation based on MATLAB.7 GUI. In: 2006 IASME/WSEAS INTERNATIONAL CONFERENCE ON ENERGY & ENVIRONMENTAL SYSTEMS, 8-10 may. 2006, Chalkida, Greece. **Proceedings...** Chalkida, 2006. p. 379–384. Disponível em: <https://www.researchgate.net/publication/237755595_In_Orbit_Performance_of_LEO_Satellite_Electrical_Power_Subsystem_-_SW_Package_for_Modelling_and_Simulation_Based_on_MatLab.7_GUI>. Acesso em: 20 nov. 2015. 34, 35

ZAMBRANO, H.; COSTA, L.; HOFFMANN, C.; BÜRGER, E. E. **Nanossatélite AESP 14: Arquitetura e Descriçã dos Subsistemas**. Av. dos Astronautas, 1.758, Jd. da Granja, São José dos Campos - SP, 12227-010: [s.n.], ago. 2013. Documento Interno - LIT21-LIT00-ES-005. 29

APÊNDICE A - ESTUDO SOBRE O SATÉLITE TANCREDO 1

Neste apêndice, apresenta-se algumas informações sobre o projeto do picossatélite Tancredo 1.

A.1 Um breve histórico

O Projeto UbatubaSat teve início em 2010 na Escola Municipal Presidente Tancredo de Almeida Neves, que está situada na cidade de Ubatuba, no estado de São Paulo, Brasil (MOURA et al., 2015).

De acordo com SANTOS et al. (2011) a primeira grande motivação para o início do projeto foi uma nota publicada pela Revista Superinteressante[1], que trazia informações sobre a possibilidade de se comprar um kit para montar um satélite, do tipo *TubeSat*, por um valor razoavelmente baixo, aproximadamente *R*$14 mil. A nota publicada pela revista despertou o interesse do professor Cândido Osvaldo de Moura, que por sua vez, achou interessante a ideia de montar um *TubeSat* com seus alunos do sexto ano. Então, em contato direto com a *Interorbital Systems Company*[2], foi possível confirmar as informações publicadas pela revista e, além disso, a empresa enviou as especificações técnicas, orçamentos e salientou a importância de uma assessoria técnica para a montagem do satélite, uma vez que, àquela seria a equipe mais jovem a colocar um satélite em órbita. Dessa forma, no início de 2011, um acordo formal foi assinado com o INPE, para que o instituto pudesse dar suporte técnico para o projeto (MOURA et al., 2015).

Após alguns meses de aprendizado e treinamento, nas áreas de eletrônica e soldagem, os estudantes iniciaram a construção do protótipo do primeiro satélite do projeto UbatubaSat, o picossatélite Tancredo 1. A primeira versão construída foi, basicamente, baseada no projeto original fornecido pela *Interorbital* e possuía, como carga útil, um gravador digital desenvolvido pelo INPE. (MOURA et al., 2015)

Devido aos atrasos no desenvolvimento do veículo lançador da *Interobital*, o *Neptune*, pelo qual o Tancredo 1 seria lançado, a equipe de coordenação do projeto decidiu buscar outras alternativas que pudessem colocar o satélite em órbita. Tomando conhecimento do projeto, em Fevereiro de 2014, a Agência Espacial Brasileira (AEB), incluiu o Tancredo 1 na iniciativa de apoio ao programa de satélites universitários, então, contratou o lançamento para 4 satélites (pico e nanossatélites), sendo que um deles seria lançado com o foguete russo DNEPR e o outros três, incluindo o Tancredo 1, com a Agência Japonesa de Exploração Espacial (JAXA - Japan Aerospace Exploration Agency), através da empresa

[1]Revista Superinteressante: http://super.abril.com.br/tecnologia/satelite-voce-ainda-vai-ter-um. Disponível em: 20 out. 2015.

[2]Site da *Interobital Systems Company*: http://www.interorbital.com/interorbital_06222015_002.htm. Disponível em: 20 out. 2015.

JAMSS (Japan Manned Space Systems Corporation), que gerencia os lançamentos realizados a partir módulo japonês *Kibo*, na Estação Espacial Internacional (ISS - Internacional Space Station). (MOURA et al., 2015)

Com a mudança do veículo lançador, foram adicionados alguns requisitos de segurança que não haviam no projeto original e, além disso, foi adicionada uma nova carga útil ao satélite, em parceria com a Divisão de Aeronomia (DAE) do INPE. A adição dessa carga, uma sonda de *Langmuir*, gerou mudanças significativas ao projeto inicial do *TubeSat* fornecido pela *Interorbital* (MOURA et al., 2015), criando uma segunda versão do Tancredo 1 (TIKAMI et al., 2014).

A sonda de *Langmuir* é responsável por fornecer medidas de densidade do plasma na ionosfera ao longo da órbita do satélite. De acordo com SANTOS et al. (2014), essas medições podem ajudar a: (i) investigar o mecanismo de geração de bolhas de plasma, (ii) medir a densidade numérica de elétrons e a distribuição espectral das irregularidades de plasma.

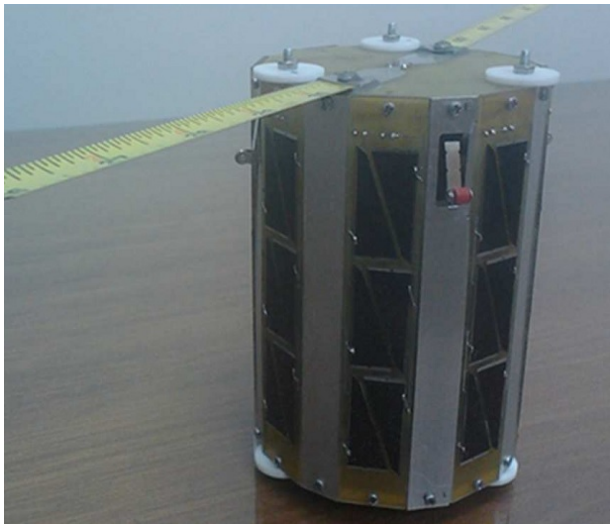
A seção a seguir, apresenta a descrição do satélite Tancredo 1.

A.2 Descrição do satélite

Segundo SANTOS et al. (2014), o picossatélite brasileiro Tancredo 1, possui massa de aproximadamente 750 g e, dessa forma, está incluído na categoria de picossatélites (FORTESCUE et al., 2004). A estrutura do satélite, como pode ser observada na Figura A.1, possui forma hexadecagonal cilíndrica, com 8,9 cm de diâmetro máximo e 12,7 cm de altura. Na Tabela A.1, é possível observar as informações orbitais do satélite[3].

[3]Para fins de simulação, pode-se utilizar os parâmetros orbitais da ISS, uma vez que, o Tancredo 1 será lançado de lá.

Figura A.1 - Foto do picossatélite Tancredo 1.



Tancredo 1 em destaque.

Fonte: Foto retirada da apresentação de Tikami e Santos (2015).

Tabela A.1 - Parâmetros Orbitais do Tancredo 1.

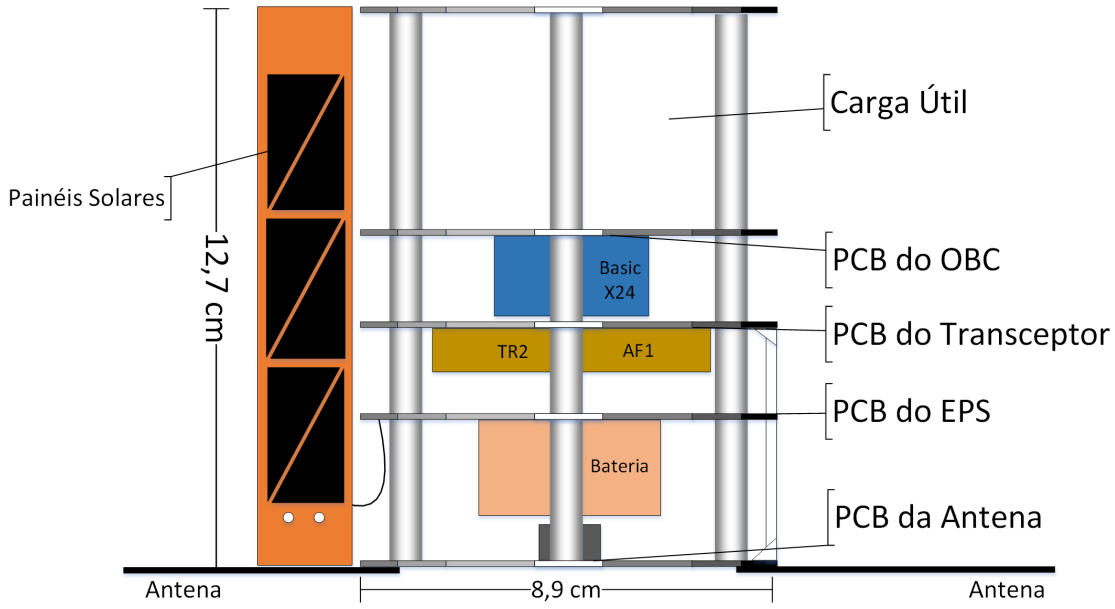
Parâmetro Orbital	**Valor**
Altitude	310 km
Excentricidade	$0,0009977$
Inclinação	$51,62°$
Argumento do Perigeu	$198,21°$
Ascensão Reta do Nodo Ascendente	$353,59°$
Anomalia Verdadeira	$299,66°$
Período orbital	90 min

Produção do Autor.

Como visto em Tikami (2014), o Tancredo 1, é composto pelos seguintes subsistemas: (i) comunicação, (ii) suprimento de energia e (iii) computador de bordo (OBC - *On Board Computer*). Além de comportar duas cargas úteis: a sonda de *Langmuir* e o gravador digital (TIKAMI et al., 2014; SANTOS et al., 2014). Para compor os subsistemas, Figura A.2, o kit para a montagem do picossatélite, fornecido pela *Interorbital*, é composto pelos seguintes componentes e distribuídos em forma de placas de circuitos impressos (PCB - *Printed Circuit Boards*) (SANTOS et al., 2014):

- (i) os arquivos no formato *Gerber*[4] para fabricação das PCBs;
- (ii) um transceptor (TR) e um amplificador (AF) de rádio frequência (RF) para comunicação com o solo;
- (iii) baterias e células solares;
- (iv) lista com os componentes eletrônicos das PCBs;
- (v) um microcontrolador (computador de bordo - OBC) e seu respectivo kit de desenvolvimento;
- (vi) *software*;
- (vii) antenas.

Figura A.2 - Layout dos equipamentos do Tancredo 1.



Layout mecânico do Tancredo 1.
Fonte: Adaptada de Moura et al. (2015).

Conforme citado anteriormente, a partir de um acordo com a DAE do INPE, adicionou-se ao Tancredo 1 uma sonda de *Langmuir* como carga útil, além disso, devido a troca do

[4]Formato do arquivo que contém o desenho do circuito.

veículo lançador foram impostos novos requisitos de segurança, dessa forma, para atender estas alterações no projeto, houve a necessidade de fazer diversas adaptações ao picossatélite. (SANTOS et al., 2014; TIKAMI et al., 2014)

Para realizar a reengenharia do projeto algumas considerações foram observadas para que adaptação pudesse ocorrer sem problemas, entre elas (TIKAMI, 2014; SANTOS et al., 2014):

- (i) Reconfiguração do *layout* das PCBs para atender os requisitos de segurança e qualidade;
- (ii) Montagem, integração e teste (AIT - *Assembly, Integration and Testing*) das novas PCBs de potência, comunicação, computador de bordo e carga útil;
- (iii) Verificação de aspectos relativos ao *budget*[5] de massa, potência, RF, bem como o *layout* das células solares;
- (iv) Estudo de inclusão de telecomando e telemetria;
- (v) Adaptação do protocolo de dados de bordo e da carga útil (gravador de voz) para uso de telemetria e telecomando;
- (vi) Aspectos de interface entre solo e espaço;
- (vii) Probabilidade de realizar experimentos a partir de uma carga de coleta de dados, utilizando a plataforma de picossatélite, alinhado aos interesses do INPE;
- (viii) Aspectos de interface com os lançador e acoplamento à ISS.

Então, o projeto pôde incorporar as duas cargas úteis, o gravador de voz da escola de Ubatuba, bem como a sonda de *Langmuir*, do INPE. A Figura A.3 apresenta o diagrama de blocos que representa o projeto, em sua versão final. (TIKAMI et al., 2014; SANTOS et al., 2014)

[5]Termo usado para representar o dimensionamento (balanço) correto dos subsistemas do satélite.

Figura A.3 - Diagrama de blocos do Tancredo 1.

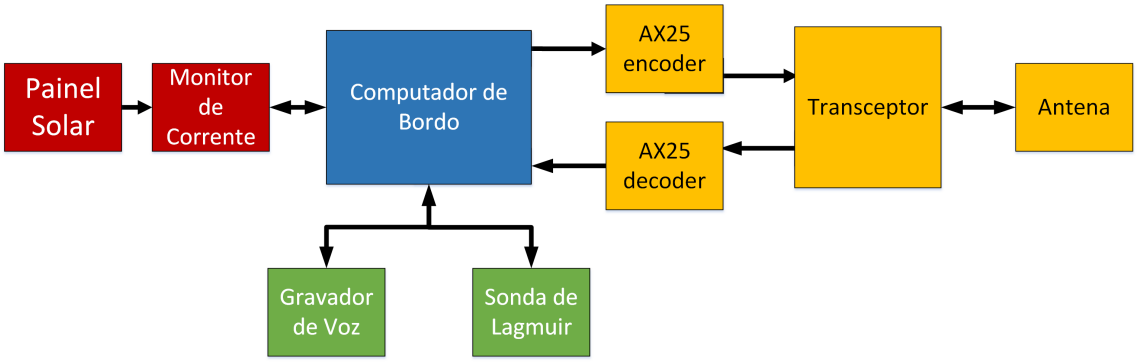


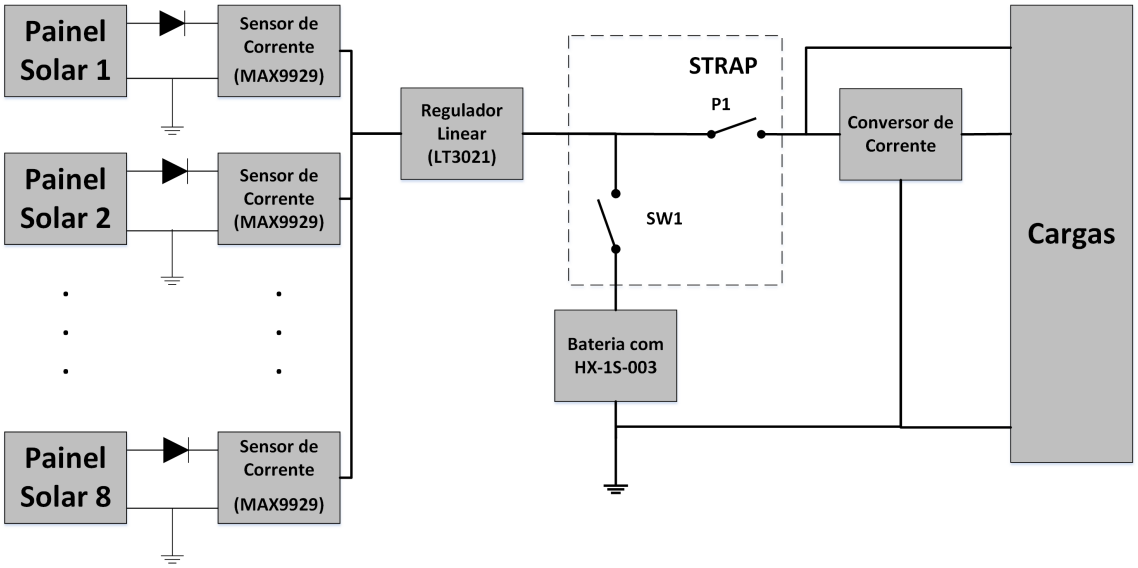
Diagrama de blocos do Tancredo 1, que inclui suas duas cargas úteis, em verde. O subsistema de energia está representado pelos blocos em vermelho. Em azul, o computador de bordo. Por fim, os blocos em amarelo, representam o subsistema de energia.

Fonte: Adaptada de Tikami et al. (2014), SANTOS et al. (2014).

O subsistema de comunicação é responsável por realizar a comunicação entre o segmento espacial (Tancredo 1) e o segmento solo (estação terrena). Para tanto, o subsistema de comunicação do Tancredo 1, é composto por um codificador e decodificador AX.25 em conjunto com um transceptor (transmissor e receptor), e também duas antenas. As antenas são feitas de aço de fita métrica, possibilitando assim, que a antenas sejam dobradas, e estão localizadas na parte externa do satélite. A frequência de operação é de aproximadamente 436 MHz com uma faixa de potência de RF selecionável, entre 100 mW e 500 mW, usando um amplificador. O protocolo do pacote de dados está modulado em AFSK com 1200 *baud rate*. Além disso, o Tancredo 1 é compatível com a banda de rádio amador e sua estação terrena é composta, basicamente por uma antena *Yagi* (espinha de peixe), projetada para operar em UHF (Ultra High Frequency) e um computador para controle do sistema. (TIKAMI et al., 2014; SANTOS et al., 2014)

O subsistema de energia, fornecido pela *Interorbital*, bem como o *deployment switch*, podem ser observados na Figura A.4. O subsistema de energia é composto por: (i) 8 painéis solares, sendo que estes painéis devem fornecer energia para o satélite, e também, e para o carregamento de duas baterias de íon-lítio, cuja carga elétrica do conjunto de baterias é de 5200 mAh; (ii) uma PCB que tem por finalidade proteger o subsistema contra sobrecarga, sobre descarga e curto-circuito. Devido ao corpo hexadecagonal do *TubeSat*, espera-se que 4 painéis solares sejam simultaneamente iluminados, fornecendo energia para o satélite nos períodos em que for iluminado pelo sol. (TIKAMI et al., 2014)

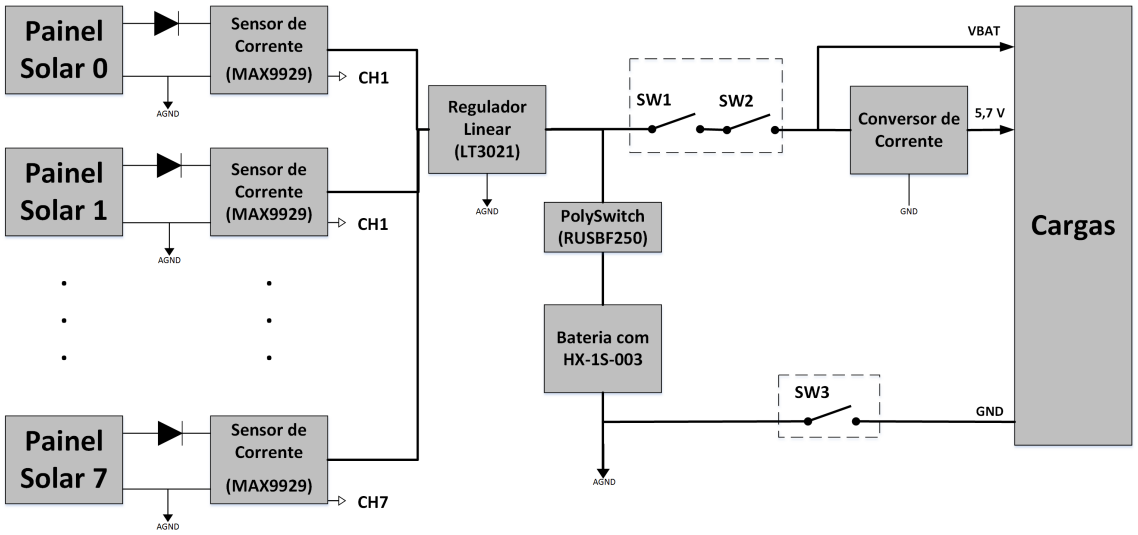
Figura A.4 - Subsistema de Energia original do Tancredo 1.



Subsistema de Energia original fornecido pela *Interorbital*.
Fonte: Adaptada de Tikami et al. (2014).

Com a mudança do veículo lançador fez-se necessário fornecer evidências de segurança da bateria e, em consequência disto, um novo esquema de proteção foi adaptado ao projeto original. Observa-se na Figura A.5, o subsistema de energia modificado, juntamente com os *deployment switches*, de acordo com as exigências da JAXA e NASA para os sistemas a bordo da ISS. (TIKAMI et al., 2014)

Figura A.5 - Versão final do Subsistema de Energia do Tancredo 1.



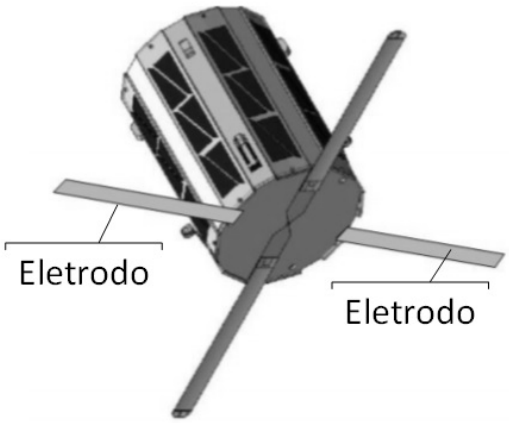
Subsistema de Energia do Tancredo 1 modificado para atender as exigências de segurança, após a troca do veículo lançador.

Fonte: Adaptada de Tikami et al. (2014).

Segundo Tikami et al. (2014), o principal componente do computador de bordo é um microcontrolador ATMEL BasicX-24 de 8 bits.

O gravador de voz, é constituído de um chip dedicado para áudio com capacidade de gravação e reprodução de 12 kHz de frequência de amostragem com áudios até 40 segundos. A sonda de *Langmuir* é um experimento científico cujo o objetivo é determinar a densidade de elétrons de plasma na ionosfera. Para a sonda, fez-se necessário a instalação de dois eletrodos, próximos as antenas, como pode ser observado na Figura A.6. (TIKAMI et al., 2014; SANTOS et al., 2014)

Figura A.6 - Diagrama de blocos do Tancredo 1.



Eletrodos inseridos no Tancredo 1 devido a inclusão da sonda de *Lagmuir*, em destaque.

Fonte: Adaptada de SANTOS et al. (2014).

Na próxima seção, apresenta-se a descrição do desenvolvimento de sonda de *Langmuir* para atender os requisitos estruturais do Tancredo 1.

A.3 Adaptação da Sonda de *Langmuir*

Planeja-se com o projeto da sonda de *Lagmuir* para o Tancredo 1, motivar o desenvolvimento de experimentos científicos que possam ser incorporados a bordo de outros satélites miniaturizados. (SANTOS et al., 2014)

A equipe de desenvolvimento enfrentou diversos desafios para realizar a adaptação da sonda de *Lagmuir* para o Tancredo 1, dentre eles:

- (i) Usar componentes eletrônicos compactos e leves;
- (ii) Atender as especificações de massa, potência e interferência e compatibilidade eletromagnética;
- (iii) Fornecer resultados experimentais atraentes para a comunidade científica; e,
- (iv) Resolver os conflitos de interface com os canais limitados de I/O do computador de bordo.

A Figura A.7 apresenta a sonda de *Langmuir* adaptada para o Tancredo 1. (SANTOS et al., 2014). E é composta pelos seguintes componentes:

- (i) Um amplificador que converte a corrente proveniente do sensor externo, o qual está exposto a ionosfera, para um certo potencial elétrico;
- (ii) Um conversor DC/DC, cuja a alimentação é proveniente do barramento principal do *TubeSat*, que fornece ±5 V para os amplificadores e gerador de polarização;
- (iii) Um gerador de polarização que fornece um potencial positivo 2,5 V ao pré-amplificador do sensor metálico;
- (iv) Dois amplificadores, um para os sinais de baixo ganho e outro para sinais de ganho elevado, provenientes do pré-amplificador; e,
- (v) Um multiplexador que envia duas telemetrias, referentes às cargas úteis, para o *downlink* do satélite.

Figura A.7 - Sonda de *Langmuir* adaptada para o *TubeSat* Tancredo 1.

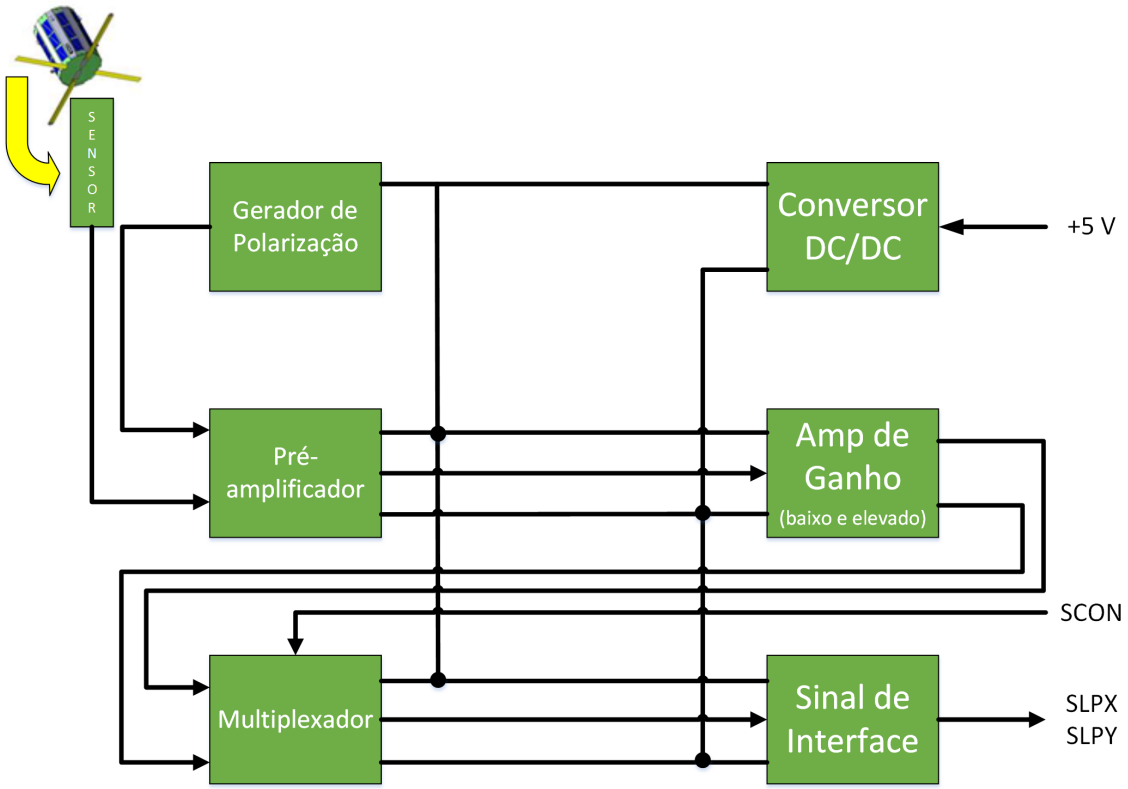


Diagrama de blocos da Sonda de *Langmuir* do Tancredo 1.
Fonte: Adaptada de SANTOS et al. (2014).

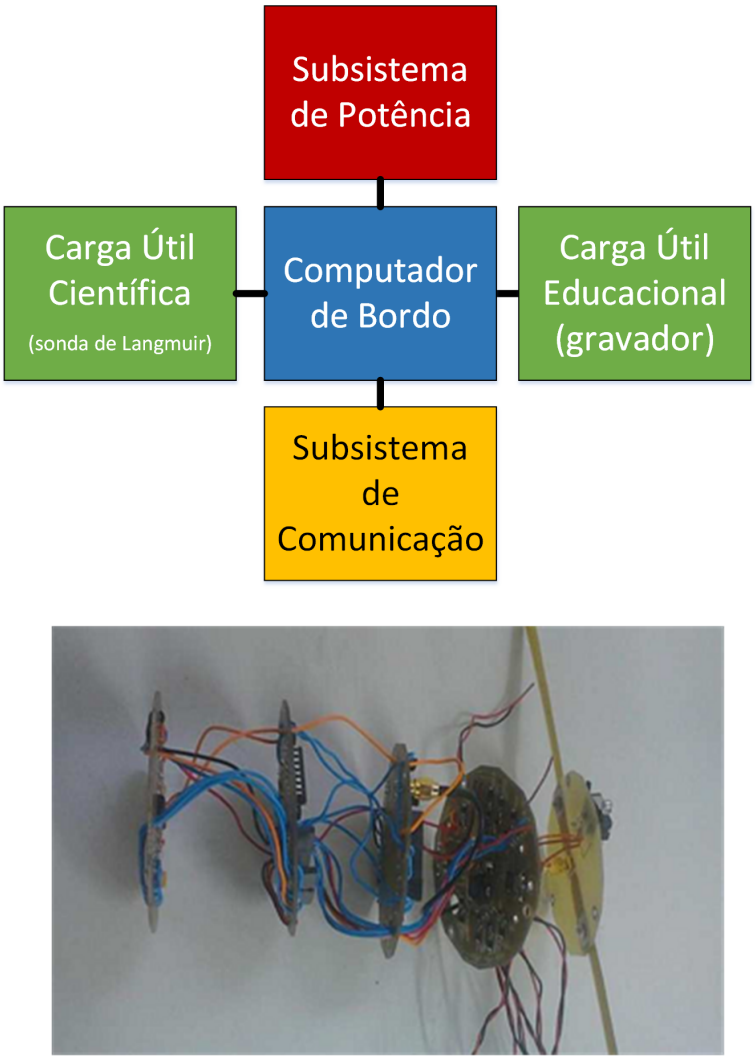
Ainda, de acordo com SANTOS et al. (2014), o sinal de controle do multiplexador, denominado SCON, é fornecido pelo computador de bordo como parte do seu *software* do módulo de carga útil. Há uma interface entre a sonda de *Langmuir* e o conversor analógico-digital (AD), que utiliza uma saída do circuito de baixa impedância, de modo a não comprometer os sinais analógicos que são disponibilizados para o sistema de aquisição do satélite.

A.4 Tancredo 1 após reengenharia

Durante o projeto do modelo de engenharia do Tancredo 1, foi percebido, segundo Tikami et al. (2014), que o uso de cabos soldados, como pode ser observado na Figura A.8, para a interconexão dos subsistemas causava certa instabilidade. Então, fez-se necessário a reconfiguração das PCBs. Esta nova interface, atendia os requisitos de segurança e qualidade e a interconexão realizada através de um barramento. A nova configuração também contemplou uma placa para transporte de sinais de potência e dados foram considerados,

como pode ser observado na Figura A.9.

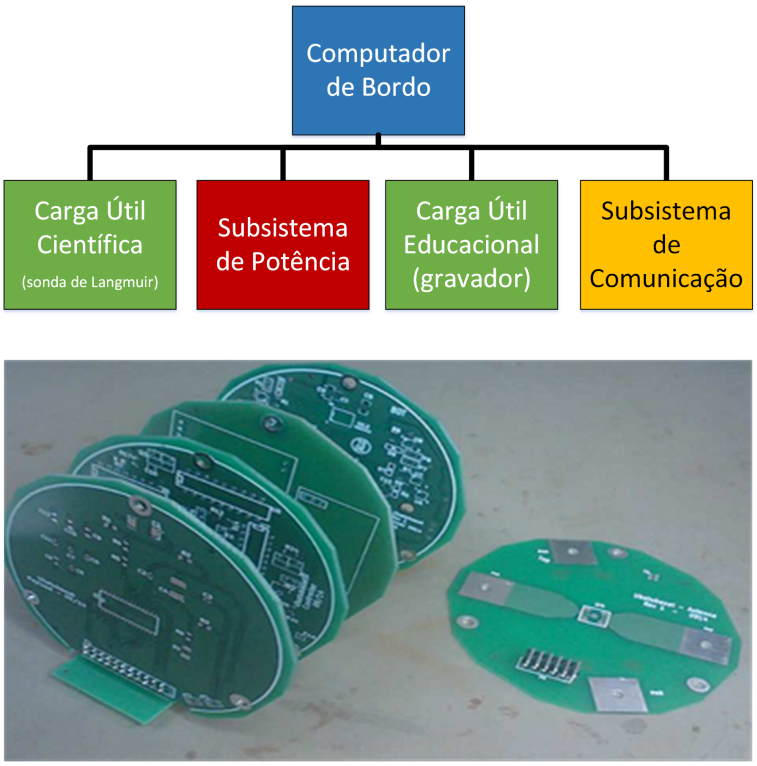
Figura A.8 - Conexão feita por cabos.



A figura ilustra a foto da interconexão centralizada do Tancredo 1. A foto destaca a grande quantidade de cabos.

Fonte: Adaptada da apresentação de Tikami e Santos (2015).

Figura A.9 - Conexão por barramento.



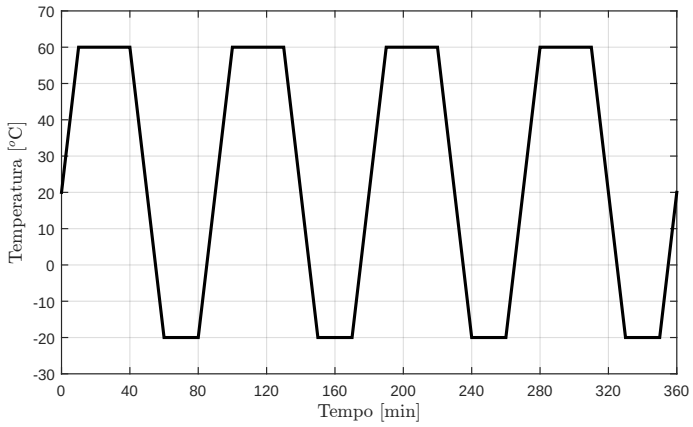
A figura ilustra a foto da interconexão distribuída do Tancredo 1.
Fonte: Adaptada da apresentação de Tikami e Santos (2015).

A.5 Testes realizados com o Tancredo 1

Após a implementação da adaptação do *hardware*, o Tancredo 1, teve de cumprir um conjunto de novas exigências, principalmente, o subsistema de potência e suas proteções. A adaptação também exigiu a realização de testes ambientais, tais como, vibração e termo-vácuo. Assim após cada teste ambientais, era feito um ciclo de carga e descarga da bateria.

O primeiro teste realizado foi o de vibração da bateria, cujos perfis de vibração abrangem diversos lançadores. Os testes de vibração foram realizados em 3 eixos. Em seguida, realizou-se outro teste ambiental, o de termo-vácuo. Este teste foi realizado no período de 6 horas, alternando as temperaturas da bateria entre -20 °C e 60 °C com 10^{-5} mbar, durante o tempo do período orbital do satélite. A Figura A.10, apresenta a variação da temperatura durante o teste executado. (TIKAMI et al., 2014)

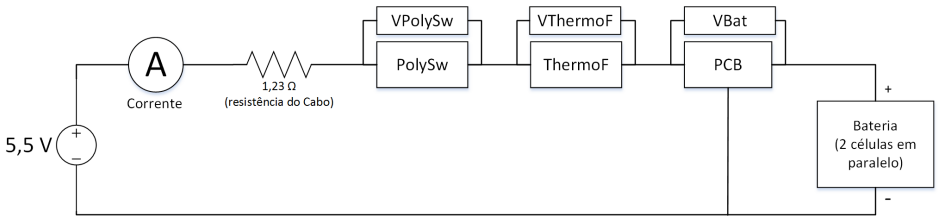
Figura A.10 - Perfil de Temperatura para o teste de termo-vácuo da bateria.



Variação de temperatura durante o teste de termo-vácuo da bateria.
Fonte: Adaptada de Tikami et al. (2014).

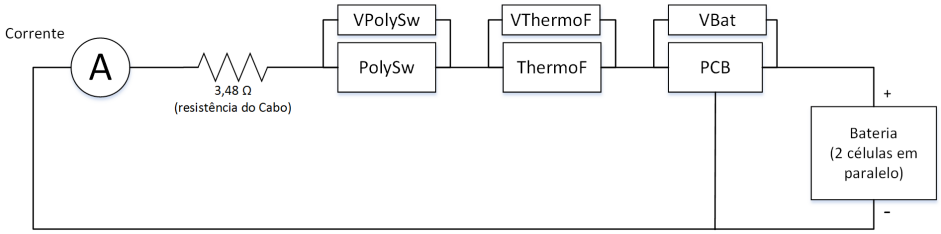
Para avaliar o perfil de carga e descarga da bateria, após os testes ambientais, adotou-se as configurações que podem ser observadas nas Figuras A.11 e A.12. (TIKAMI et al., 2014)

Figura A.11 - Teste de carga da bateria.



Configuração do teste para carregar a bateria.
Fonte: Adaptada de Tikami et al. (2014).

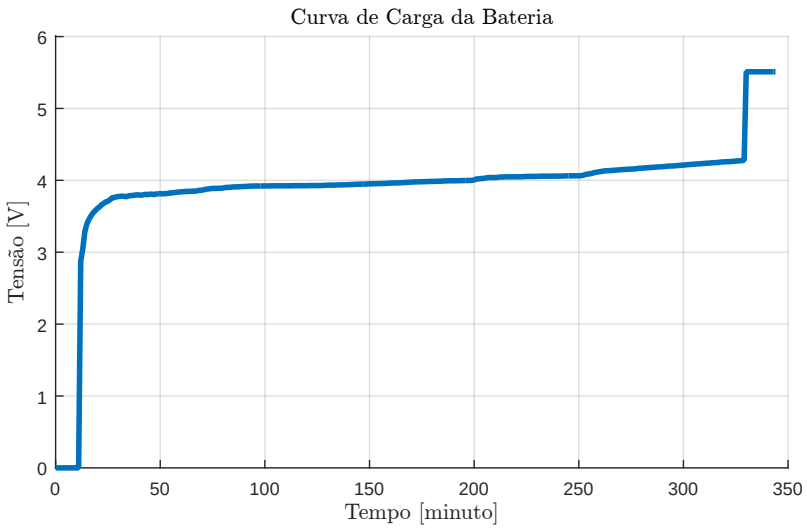
Figura A.12 - Teste de descarga da bateria.



Configuração do teste para descarregar a bateria.
Fonte: Adaptada de Tikami et al. (2014).

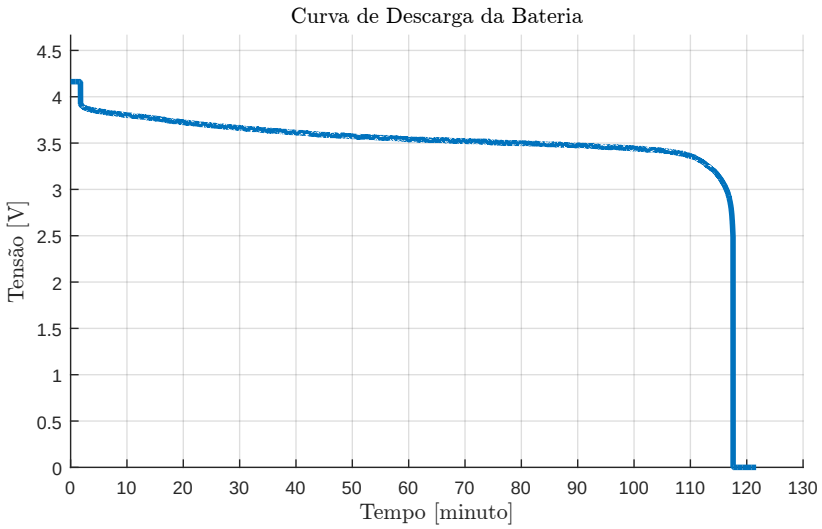
De acordo com Tikami et al. (2014), após os testes de termo-vácuo as baterias apresentaram as assinaturas de carga e descarga, que podem ser observadas, respectivamente, nas Figuras A.13 e A.14.

Figura A.13 - Curva de Carga.



Curva de carga da bateria após testes ambientais.
Fonte: Adaptada de Tikami et al. (2014).

Figura A.14 - Curva de Descarga.



Curva de descarga da bateria após testes ambientais.
Fonte: Adaptada de Tikami et al. (2014).

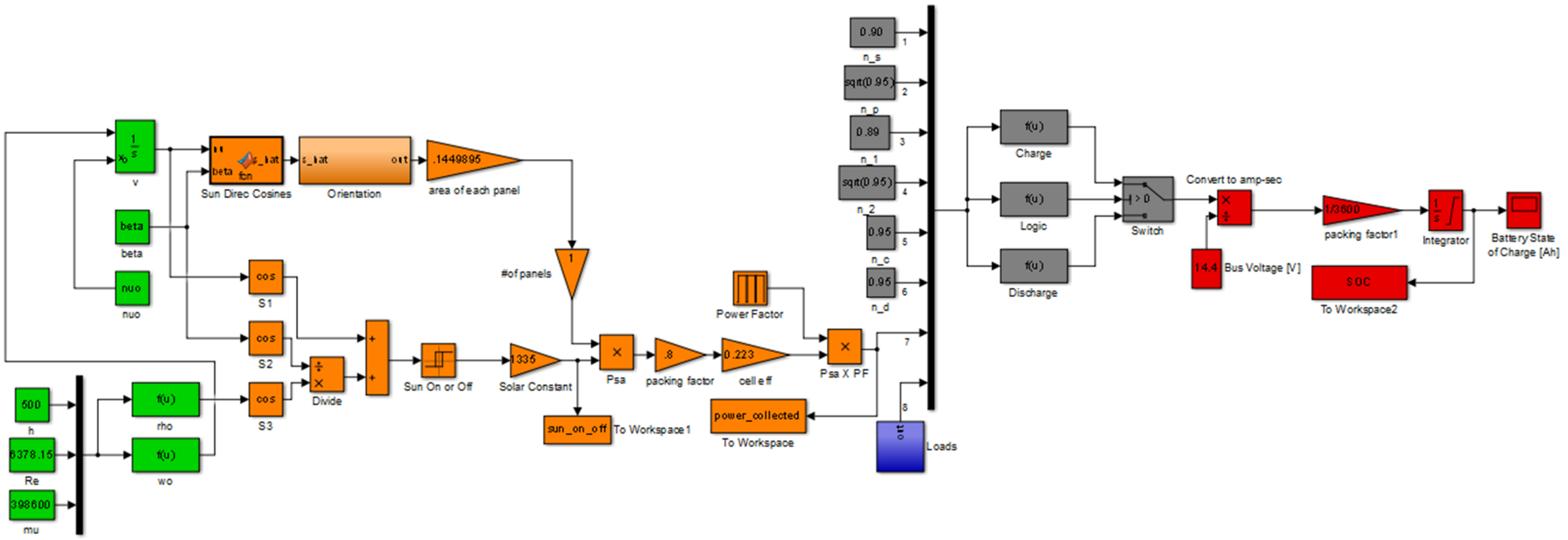
A previsão de lançamento do picossatélite Tancredo 1 é para o início de 2016. (TIKAMI; SANTOS, 2015)

APÊNDICE B - MODELO DE REFERÊNCIA DO EPS

Como visto no Capítulo 5, a Figura 5.9 foi baseada no modelo descrito por Melone (2009). A Figura B.1 apresenta o modelo completo, descrito a seguir:

- Os blocos responsáveis pela propagação de órbita estão destacados em verde.
- Os blocos em laranja dizem respeito à potência gerada pelos paineis solares, frente à incidência de Sol.
- As cargas são representadas pelo bloco em azul.
- A PCDU é representada pelos blocos em cinza.
- A bateria é representado pelos blocos em vermelho.

Figura B.1 - Visão geral do Modelo do EPS.

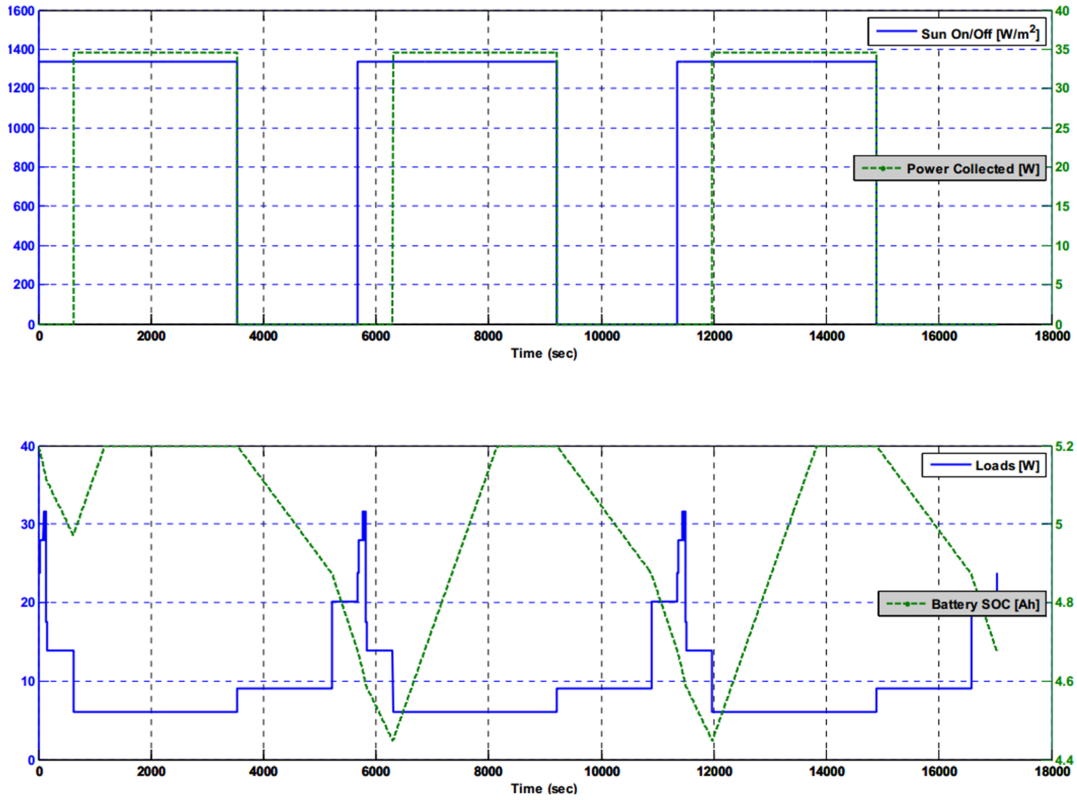


Visão geral do modelo do EPS, implementado em *Simulink*.

Fonte: Melone (2009).

Após a implementação, o modelo foi validado, a partir da inspeção visual comparando os resultados apresentados por Melone (2009), Figura B.2, e os resultados obtidos na simulação, Figura B.3 e B.4.

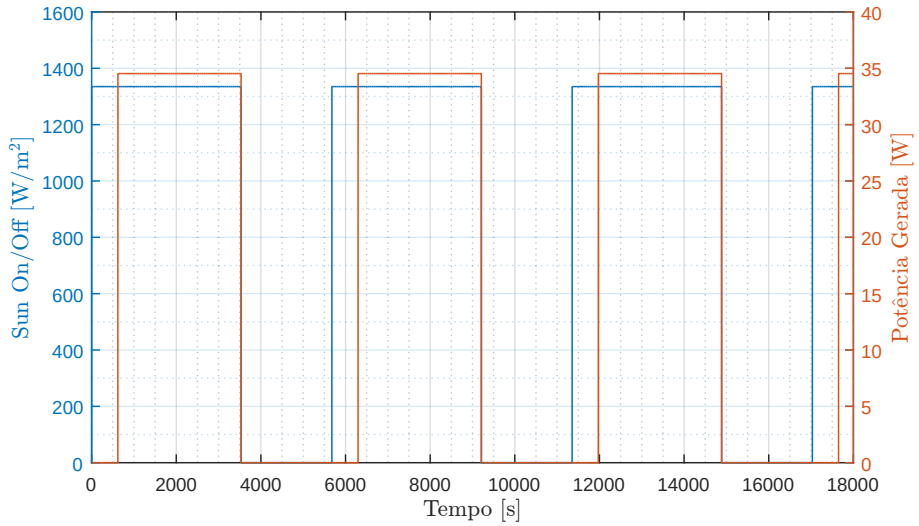
Figura B.2 - Resultados Originais.



(a) Apresenta o gráfico da incidência do sol (*Sun On/Off*) e potência gerada pelo painel solar (*Power Collected*); (b) Apresenta a potência consumida (*Loads*) e estado de carga da bateria (SOC, *State of Charge* do inglês).

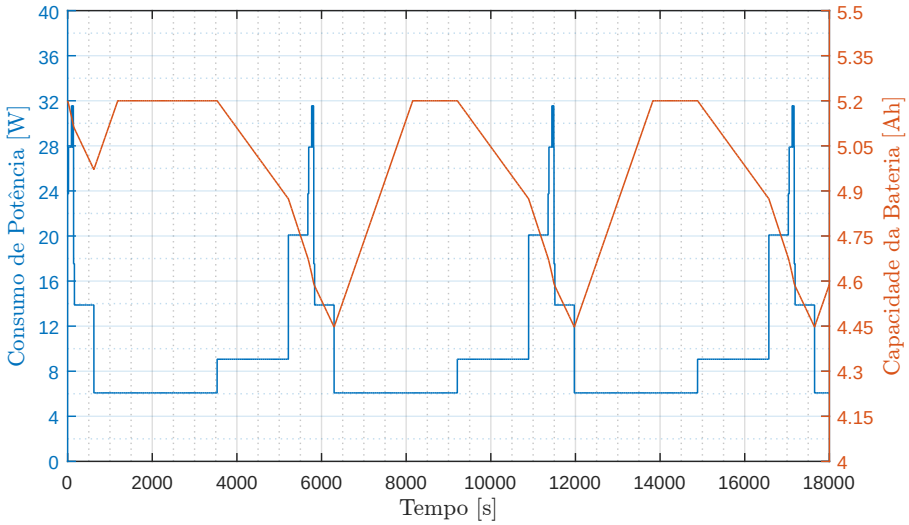
Fonte: Melone (2009).

Figura B.3 - Incidência do Sol e Potência Gerada.



A figura apresenta a incidência do sol e a potência gerada pelo painel solar.
Fonte: Produção do autor.

Figura B.4 - Potência Consumida e Estado de Carga.



A figura apresenta a potência consumida e o estado de carga da bateria.
Fonte: Produção do autor.

APÊNDICE C - EXEMPLO DO PLANO DE VERIFICAÇÃO

Este apêndice apresenta um Plano de Verificação simplificado para o subsistema de energia do Tancredo 1. Este plano segue as definições descritas pela ECSS (2009a).

Todo documento deve possuir capa, introdução e sumário e glossário para facilitar o entendimento do leitor. No entanto, para fins de simplificação, omite-se estes itens nesta seção, focando-se apenas no conteúdo do Plano de Verificação, apresentados nas seções, a seguir.

C.1 Requisitos do EPS

Devem ser verificados os requisitos apresentados na Tabela C.1

Tabela C.1 - Requisitos a Serem Verificados.

Item	Requisito
SAT01.SA.FUNC001	O Subsistema de Energia Elétrica deve fornecer uma tensão de 5 V para o Barramento.
SAT01.SA.FUNC002	A Bateria deve ser recarregada toda vez que o Painel Solar receber energia solar.
SAT01.SA.FUNC003	O estado de carga da Bateria não deve ser menor que 3 Ah.

Produção do autor.

C.2 Método de Verificação

Define como a verificação dos requisitos apresentados na Seção C.1 deve ser executada.

Os requisitos apresentados na Tabela C.1 devem ser verificados através de testes.

C.3 Nível de Verificação

A verificação dos requisitos deve ser realizada no final da Fase B, quando o projeto preliminar do subsistema é definido.

Realiza-se a verificação em nível de componentes e subsistema.

C.4 Filosofia de Modelos

Para testar o subsistema, de modo a verificar os requisitos apresentados na Tabela C.1 são utilizados os seguintes modelos:

- (i) Modelos Virtuais *software*: o modelo dos componentes do subsistema são implementados em MATALAB/*Simulink*;

- (ii) Modelos híbridos: combinam *software* e componentes de *hardware* (ver item (iii));
- (iii) Modelo prototipado: *hardware* construído com componentes COTS, que pode ser utilizados para identificar problemas na solução adotada, já nas etapas inicias de desenvolvimento.

C.5 Ferramentas de Verificação

As ferramentas utilizadas para verificar os requisitos apresentados na Tabela C.1 são:

- Simulador para Verificação Funcional: que contempla os modelos virtuais;
- *Arduino Uno*: utilizado para representar a PCDU do EPS. Neste caso, o modelo virtual do da PCDU é transferido para o *hardware*.
- *Arduino Mega*: utilizado como equipamento de interface (FEE) entre a simulação virtual e o equipamento de interface.

C.6 Casos de Teste

Para verificar os requisitos da Tabela C.1, são considerados os casos de teste das Tabelas C.2, C.3 e C.4.

Tabela C.2 - Caso de Teste 1.

Item	**Descrição**
Identificação do caso de teste	controlBus.
Descrição	Este caso de teste de verificar os níveis de tensão do EPS durante uma órbita, cujo nível esperado é de 5 V .
Configuração do Teste	Configurações: (i) somente *software* e *hardware-in-the-loop*.
Equipamento de *Front-end*	Na configuração *hardware-in-the-loop* faz-se necessário o uso de um equipamento de interface.
Requisitos a serem verificados	SAT01.SA.FUNC001.
Sequência de Teste	(i) Simular pelo menos 1 órbita completa do satélite; (ii) Verificar os níveis de tensão do barramento durante o período simulado.

Produção do autor.

Tabela C.3 - Caso de Teste 2.

Item	**Descrição**
Identificação do caso de teste	controlPCDU.
Descrição	Este caso de teste deve verificar se as baterias são recarregadas sempre que a energia gerada pelos painéis solares for maior que a energia consumida pelo satélite.
Configuração do Teste	Configurações: (i) somente *software* e *hardware-in-the-loop*.
Equipamento de *Front-end*	Na configuração *hardware-in-the-loop* faz-se necessário o uso de um equipamento de interface e de uma bateria
Requisitos a serem verificados	SAT01.SA.FUNC002.
Sequência de Teste	(i) Simular pelo menos 1 órbita completa do satélite; (ii) Verificar curva de carga e descarga da bateria.

Produção do autor.

Tabela C.4 - Caso de Teste 3.

Item	**Descrição**
Identificação do caso de teste	controlBattery
Descrição	Este caso de teste de verificar o estado de carga da bateria.
Configuração do Teste	Configurações: (i) somente *software* e *hardware-in-the-loop*.
Equipamento de *Front-end*	Na configuração *hardware-in-the-loop* faz-se necessário o uso de um equipamento de interface.
Requisitos a serem verificados	SAT01.SA.FUNC003.
Sequência de Teste	(i) Simular pelo menos 1 órbita completa; (ii) Verificar se a energia fornecida pela bateria foi o suficiente para suprir a demanda.

Produção do autor.

C.7 Matriz de Teste

A Matriz de Teste, ilustrada na Figura C.1 relaciona os requisitos e os casos de teste, e é preenchida ao final da execução dos testes para que a equipe de verificação dê o veredicto de teste.

Figura C.1 - Tabela da Matriz de Teste.

	A	B	C	D	E	F
1	**Item**	**Requirement**	**ReqID**	**Simulation Run**	**Test Case Name**	**Result**
2	1	O Subsistema de Energia Elétrica deve fornecer uma tensão de 5 V para o Barramento.	SAT01.SA.FUNC001	1	controlBUS	
3	2	A Bateria deve ser recarregada toda vez que o Painel Solar receber energia solar.	SAT01.SA.FUNC002	1	controlPCDU	
4	3	O estado de carga da Bateria não deve ser menor que 3 Ah.	SAT01.SA.FUNC003	1	controlBattery	
5						

A figura apresenta tabela da Matriz de Teste.

Fonte: Produção do autor.

APÊNDICE D - PUBLICAÇÕES

Este anexo apresenta as publicações realizadas durante o desenvolvimento desta dissertação, sendo elas:

- Simulação com *Hardware-in-the-loop* integrada por *Arduino* a um Simulador de Satélite - artigo apresentado no 5° *Workshop* de Engenharia e Tecnologias Espaciais (WETE) no INPE. A primeira página no arquivo pode ser observada na Figura D.1.
- Utilização de Realidade Virtual, Aumentada e Cruzada em Simuladores de Satélites no INPE - capítulo de livro apresentado no XVII Simpósio de Realidade Virtual e Realidade Aumentada (SVR). A primeira página no arquivo pode ser observada na Figura D.2.
- Verificação de Requisitos através do Uso de um Simulador Funcional - artigo apresentado no 6° *Workshop* de Engenharia e Tecnologias Espaciais (WETE) no INPE. A primeira página no arquivo pode ser observada na Figura D.3.
- *Towards an Automated Hybrid Test and Simulation Framework to Functional Verification of Nanosatellites' Electrical Power Supply Subsystem* - artigo aceito para apresentação no *II Latin American IAA CubeSat Workshop* (LACW) em 2016.

Figura D.1 - Simulação com *Hardware-in-the-loop* integrada por *Arduino* a um Simulador de Satélite.

5°Workshop em Engenharia e Tecnologia Espaciais

São José dos Campos/SP - 12,13 e 14 de agosto de 2014

Simulação com Hardware-in-the-loop integrada por Arduino a um Simulador de Satélite

RODRIGUES, I.P. [1], AMBROSIO, A.M [2]

[1]Instituto Nacional de Pesquisas Espaciais, São José dos Campos, SP, Brasil
Aluno de Mestrado do curso de Engenharia e Gerenciamento de Sistemas Espaciais - CSE.

[2]Instituto Nacional de Pesquisas Espaciais, São José dos Campos, SP, Brasil

italo.rodrigues@hotmail.com

Resumo. *O processo de simulação possibilita ao usuário a realização de estudos a respeito do sistema modelado, e também, explorar situações que se tenha pouco conhecimento, com a finalidade de adquirir embasamento teórico e preparação [FREITAS, 2008]. Assim, este trabalho propõe uma interface entre um simulador de satélites e um sistema real em teste, utilizando simulação com hardware na malha para verificar a viabilidade desta aplicação.*

Palavras-chave: Simulação, hardware-in-the-loop, arduino,

1. Introdução

O planejamento adequado e preparação das atividades de um determinado satélite necessitam de um cenário simulado visando aumentar sua confiabilidade [AMBROSIO, CARDOSO, ORLANDO e BIANCHI 2006]. Neste contexto, a realização de simulações irá auxiliar o teste e a validação de funções dos satélites concorrentemente ao desenvolvimento dos equipamentos reais.

Este trabalho tem por objetivo avaliar o desempenho e distribuição de funcionalidade de uma interface, utilizando tecnologias de uso comercial (Arduino) [BANZI, 2011], entre os equipamentos/subsistemas, em teste, com a infraestrutura de simulação.

2. Metodologia

Para este trabalho, será realizado: (i) o estudo da ferramenta de modelagem MATLAB e das facilidades de exportação dos modelos para embarcar no Arduino, (ii) a pesquisa e identificação dos requisitos, capacidades elétricas e de comunicação necessárias para interface, (iii) a comparação dos requisitos aos Shields para Arduino disponíveis para compra, (iv) seguido pelo desenvolvimento de uma cenário de simulação que contêm partes computacionais (software, MATLAB, por exemplo) e físicas (HIL - hardware-in-the-loop) e, (v) finalizando com o estudo da viabilidade desta solução em termos de desempenho e funcionalidades, frente as interfaces encontradas em uso.

Artigo apresentado em 2014 no WETE.

Fonte: Rodrigues e Ambrosio (2014).

Figura D.2 - Utilização de Realidade Virtual, Aumentada e Cruzada em Simuladores de Satélites no INPE.

Tendências e Técnicas em Realidade Virtual e Aumentada, v. 5, p. 139-159, maio/2015.
CERQUEIRA, Christopher Shneider et al. *Utilização de Realidade Virtual, Aumentada e Cruzada em Simuladores de Satélites no INPE.*

Utilização de Realidade Virtual, Aumentada e Cruzada em Simuladores de Satélites no INPE

Christopher Shneider Cerqueira
Italo Pinto Rodrigues
Carlos José Alves Moreira
Valdemir Carrara
Ana Maria Ambrosio
Claudio Kirner

Abstract

The National Institute for Space Research – INPE (in Portuguese "Instituto Nacional de Pesquisas Espaciais") is the major research center of the space sector at Brazil. Among the main research and development activities at INPE, is the Space Engineering and Technology ETE (in Portuguese "Engenharia e Tecnologia Espaciais"), which covers, in the institute, new space technologies and space systems development. Nowadays, ETE post-graduation has been providing studies and surveys on interaction technologies, as Augmented, Virtual and Cross Reality, to support the institute's satellite operations and developments activities in the future. This chapter describes some of the initiatives undertaken in the ETE post-graduation sector.

Resumo

O Instituto Nacional de Pesquisas Espaciais (INPE) é o principal centro de pesquisas do setor espacial no Brasil. Dentre as principais atividades de pesquisa e desenvolvimento no INPE, encontra-se a Engenharia e Tecnologia Espaciais (ETE), área de atuação voltada para o desenvolvimento de sistemas e tecnologias espaciais. Atualmente, a pós-graduação em Engenharia e Tecnologia Espaciais tem realizado pesquisas e estudos com tecnologias de interação, como Realidade Aumentada, Virtual e Cruzada para apoiarem as atividades de operações e desenvolvimento de futuros satélites do instituto. Este capítulo apresenta as iniciativas atuais realizadas na área da pós-graduação da ETE.

1. Introdução

A área de Engenharia e consequentemente, seus processos e produtos, são os primeiros setores a serem beneficiados por novos desenvolvimentos em interação, antes mesmo do consumidor final. Novas metodologias de interação indicam novos caminhos para solucionar problemas, permitem novas abordagens, novas maneiras de visualizar e interagir para solucionar um problema no ambiente de

Artigo apresentado em 2015 no SVR.
Fonte: Cerqueira et al. (2015).

Figura D.3 - Verificação de Requisitos através do Uso de um Simulador Funcional.

6º Workshop em Engenharia e Tecnologia Espaciais

São José dos Campos/SP - 18,19 e 20 de agosto de 2015

Verificação de Requisitos através do Uso de um Simulador Funcional

RODRIGUES, I.P. [1], AMBROSIO A. M. [2]

[1]Instituto Nacional de Pesquisas Espaciais, São José dos Campos, SP, Brasil
Aluno de Mestrado do curso de Engenharia e Gerenciamento de Sistemas Espaciais - CSE.

[2] Instituto Nacional de Pesquisas Espaciais, São José dos Campos, SP, Brasil

italo.rodrigues@hotmail.com

Resumo. *Este trabalho propõe o uso de um simulador para aplicação de testes no subsistema de um satélite, representado por um protótipo, com o objetivo de verificar suas funções nas fases iniciais de desenvolvimento. O simulador foi desenvolvido no MATLAB/SIMULINK e o modelo comportamental do subsistema foi embarcado no Arduino. Assim simulador deve aplicar os testes no equipamento físico para verificar os requisitos elencados.*

Palavras-chave: Simulação; Verificação; Teste; *Hardware-in-the-loop*; *Power Supply*

1. Introdução

De acordo com o Memorando Técnico de Modelagem e Simulação da [ECSS, 2010], as atividades de modelagem e simulação são efetuadas com o objetivo de apoiar a especificação, projeto, verificação e operação dos sistemas espaciais (artefato espacial, segmento espacial e/ou segmento terrestre). Os simuladores utilizados ao longo do ciclo de vida de um projeto espacial, podem ser observados na Figura 1. O uso de modelagem e simulação de forma coerente ao longo do ciclo de vida gera alguns benefícios significativos para o projeto, tais como, redução de riscos e custos. [ECSS, 2010]

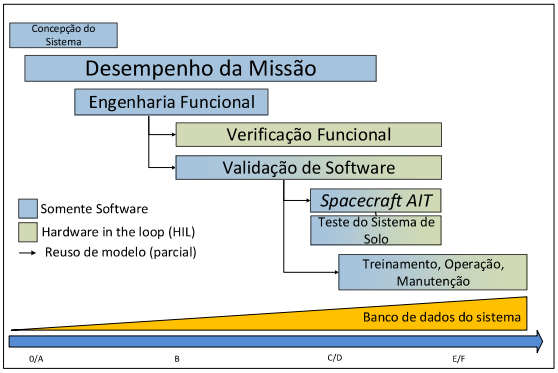


Figura 1. Simuladores ao longo do ciclo de vida.

Artigo apresentado em 2015 no WETE.

Fonte: Rodrigues e Ambrosio (2015).

Figura D.4 - *Towards an Automated Hybrid Test and Simulation Framework to Functional Verification of Nanosatellites' Electrical Power Supply Subsystem.*

IAA-BR-16-0S-0P

Towards an Automated Hybrid Test and Simulation Framework to Functional Verification of Nanosatellites' Electrical Power Supply Subsystem

Italo Pinto Rodrigues, Ana Maria Ambrosio **, Christopher Shneider Cerqueira***,*

Tests in the space systems development life cycles are necessary to early verify requirements fulfillment, ensuring that the systems developed are correct. Nowadays, the efforts to develop miniaturized satellites and so, their development suite is increasing. Meanwhile, the initiatives is adopting MBSE (Model Based System Engineering) for automating processes through life cycles by: model design, simulation and model transformation is also growing. In MBSE development approach, models are the focus of the activities. The models describe requirements, functionalities and interfaces of a system, and their subsystems, considered here as "*input models*". In the context of an electrical power subsystem (EPS), the design engineers have to (i) generate models representing solar array, battery, voltage regulators, loads, etc., to request to for implementation solutions. The engineers also should (ii) provide a verification plan, derived from requirements, to ensure the developed solutions functionality correctness. In this scenario, the following question raises: "*how to interconnect the "input models" with verification plans, developed solutions and test executions*?" This paper aims to describe the structure of an automated functional verification framework to nanosatellite's EPS, using COTS (commercial-of-the-shelf) tools, such as MATLAB/Simulink®, MS. Excel®, and Arduino. We propose the models are as granular as in the verification plans (it is not possible to test internal behaviors from a black box artifact), so, each model represent an element in a unique file and a sequencer will integrate them, as a DSM (Design Structure Matrix) in Excel. In the context of the proposed framework, the subsystem verification enables three test configurations: fully simulated, fully simulated considering physical interface model, and with hardware-in-the-loop (HIL). One of advantages of the proposed framework is the use of models from the start of the mission development, providing the reuse of these models throughout the lifecycle, minimizing costs. The paper shows also results of development of the framework using an EPS behavioral model.

* INPE, ETE/CSE, Brazil, italo.rodrigues@hotmail.com
** INPE, ETE/DSS, Brazil, ana.maria@inpe
*** INPE, ETE/CSE, Brazil, christophercerqueira@gmail.com

Artigo aceito para o LACW.

Fonte: Produção do autor.