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

**sid.inpe.br/mtc-m21b/2014/05.15.18.05-TDI**

# APLICAÇÃO DE METODOLOGIAS DE TESTE BASEADO EM MODELOS NA VERIFICAÇÃO E VALIDAÇÃO DE MECANISMOS DE FDIR DE SISTEMAS DE CONTROLE DE ATITUDE E ÓRBITA

Andre Corsetti

Dissertação de Mestrado do Curso de Pós-Graduação em Engenharia e Tecnologia Espaciais/Engenharia e Gerenciamento de Sistemas Espaciais, orientada pelas Dras. Ana Maria Ambrosio, e Maria de Fátima Mattiello-Francisco, aprovada em 27 de junho de 2014.

URL do documento original:
<http://urlib.net/8JMKD3MGP5W34M/3GAMHC5>

INPE
São José dos Campos
2014

**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/2014/05.15.18.05-TDI**

# APLICAÇÃO DE METODOLOGIAS DE TESTE BASEADO EM MODELOS NA VERIFICAÇÃO E VALIDAÇÃO DE MECANISMOS DE FDIR DE SISTEMAS DE CONTROLE DE ATITUDE E ÓRBITA

Andre Corsetti

Dissertação de Mestrado do Curso de Pós-Graduação em Engenharia e Tecnologia Espaciais/Engenharia e Gerenciamento de Sistemas Espaciais, orientada pelas Dras. Ana Maria Ambrosio, e Maria de Fátima Mattiello-Francisco, aprovada em 27 de junho de 2014.

URL do documento original:
<http://urlib.net/8JMKD3MGP5W34M/3GAMHC5>

INPE
São José dos Campos
2014

Dados Internacionais de Catalogação na Publicação (CIP)

Corsetti, Andre.

C818a Aplicação de metodologias de teste baseado em modelos na verificação e validação de mecanismos de FDIR de sistemas de controle de atitude e órbita / Andre Corsetti. – São José dos Campos : INPE, 2014.

xxii + 149 p. ; (sid.inpe.br/mtc-m21b/2014/05.15.18.05-TDI)

Dissertação (Mestrado em Engenharia e Tecnologia Espaciais/Engenharia e Gerenciamento de Sistemas Espaciais) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2014.

Orientadoras : Dras. Ana Maria Ambrosio, e Maria de Fátima Mattiello-Francisco.

1. CoFI. 2. InRob. 3. Teste baseado em modelos. 4. AOCS. 5. FDIR. I.Título.

CDU 629.7.062.2

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***

Dr. Marcio Silva Alves Branco

***Presidente / INPE / São José dos Campos - SP***

Dra. Ana Maria Ambrosio

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

Dra. Maria de Fátima Mattiello Francisco

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

Dr. José Maria Parente de Oliveira

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

***Este trabalho foi aprovado por:***

***( ) maioria simples***

***(x) unanimidade***

Aluno (a): ***Andre Corsetti***

***São José dos Campos, 27 de Junho de 2014***

*Dedico está dissertação à minha família.*

## AGRADECIMENTOS

Agradeço a Deus e a minha família por me trazerem até aqui.

Agradeço às minhas orientadoras Dra. Ana Maria Ambrosio e Dra. Maria de Fátima Mattiello-Francisco por me darem a oportunidade de estudar sob suas orientações, me puxar a melhorar, me motivar e me levar ao término deste mestrado.

Agradeço à empresa COMPSIS por me apoiar no mestrado, tanto permitindo meu trabalho no mesmo quanto disponibilizando material de trabalho para realização de experimentos.

Agradeço aos meus amigos por me darem momentos de alegria e diversão na vida.

Agradeço aos professores do INPE e colegas de trabalho na COMPSIS por me fazerem um profissional da área espacial.

Agradeço a todos que contribuíram e ajudaram na realização deste mestrado.

E um agradecimento especial à Roberta Galetti por fazer parte da minha vida.

## RESUMO

Sistemas de controle de atitude e órbita são sistemas complexos e críticos para realização da missão de satélites, contendo a parte central de sua lógica implementada em software. Cabe ao software lidar com questões de falhas de todos os equipamentos envolvidos no sistema de controle, o qual consiste de sensores, atuadores e computador de bordo, e do próprio software. A capacidade de se recuperar de falhas influencia fortemente a vida operacional do satélite com possibilidades desde morte pré matura até extensão da sua vida útil. Os mecanismos de detecção, isolamento e recuperação de falhas (FDIR) são responsáveis por detectar e tratar as possíveis falhas, e a verificação e validação destes mecanismos se posta como um grande desafio visto a complexidade do software, a dimensão do sistema, e dos esforços da atividade, além da qualidade exigida do sistema de software crítico. Metodologias de teste baseado em modelos visam padronizar os processos de teste, normatizando-os tanto em termos de qualidade dos testes quanto em termos de esforços requeridos nas atividades de teste. Duas metodologias de teste baseado em modelos, denominadas CoFI e InRob, desenvolvidas no domínio de sistemas espaciais, são estudadas e avaliadas no escopo desse trabalho quanto a suas aplicabilidades em sistemas de controle de atitude e órbita de satélites. O trabalho utilizou-se de uma análise teórica das metodologias frente a exigências de teste da área espacial, e de um experimento prático, no qual as metodologias foram aplicadas em um protótipo de sistema de controle de atitude e órbita. Para este experimento a elicitação de requisitos a serem validados e a implementação do protótipo foram necessárias. A conclusão do estudo aponta forte contribuição das metodologias CoFI e InRob como guia para a construção de modelos comportamentais representativos de um sistema de controle de atitude e órbita, quando mecanismos de FDIR são devidamente considerados nos modelos.

# APLICATION OF MODEL BASED TEST METHODOLOGIES IN VERIFICATION AND VALIDATION OF FDIR MECHANISMS OF ATTITUDE AND ORBIT CONTROL SYSTEMS

## ABSTRACT

Attitude and orbit control systems are complex and critic systems for satellite mission execution, and they contain the central part of its logic implemented in software. The software needs to deal with fault issues of all control system related equipment, namely sensors, actuators, the onboard computer, and also the software itself. The capacity of recovering from faults is a major factor in the satellite's operational life span, with possibilities of pre mature death to mission time extension. The FDIR mechanisms are responsible for detecting and recovering the fault issues raised in operation. The verification and validation of these mechanisms pose a great challenge, because of software complexity, system dimension, effort necessary by the activity, and the required quality of the critic software product. Model based testing methodologies aim to standardize the test process, normalizing tests quality and also the activity effort. Two model based testing methodologies, named CoFI and InRob, developed inside the area of space systems are studied and evaluated by its applicability in satellites attitude and orbit control systems. The study used analysis of the theoretical capacities of the methodologies compared with the requirements for testing from the space area, and realized a practical experiment, applying the two methodologies in an attitude and orbit control system prototype. For this experiment to be carried out, system requirements elicitation and the implementation of an attitude and orbit control system prototype were necessary. The conclusion of the study shows strong contribution of the methodologies CoFI and InRob in guiding the construction of representative behavioral models of the attitude and orbit control system, when FDIR mechanisms are adequately considered in the models.

## LISTA DE FIGURAS

**Pág**.

# LISTA DE TABELAS

**Pág**.

## LISTA DE SIGLAS E ABREVIATURAS

| | |
|---|---|
| ADCS | Attitude Determination and Control System |
| AOCS | Attitude and Orbit Control System |
| CDB | Computador de Bordo |
| CoFI | Conformidade e Injeção de Falhas |
| CPU | Unidade de Processamento Central |
| CRC | Cyclic Redundancy Check |
| DDF | Detecção e Diagnóstico de Falhas |
| ECSS | European Cooperation for Space Standardization |
| ESA | Agência Espacial Européia |
| FDIR | Fault Detection, Isolation and Recovery |
| FSM | Finite State Machine |
| GNC | Guidance, Navigation and Control |
| GPS | Global Positioning System |
| HJ2IF | Hit or Jump to IF |
| HK | Housekeeping |
| IHM | Interface Homem Máquina |
| Inf | Infinito |
| INPE | Instituto Nacional de Pesquisas Espaciais |
| InRob | Interoperabilidade e Robustez de Sistemas Intensivos em Software |
| MBT | Testes Baseados em Modelos |
| MEF | Máquina de Estados Finitos |
| NaN | Not a Number |
| NASA | National Aeronautics and Space Administration |
| OBDH | Onboard Data Handler |
| PDU | Power Distribution Unit |
| PUS | Packet Utilization Standard |
| RB | Requisitos Base |
| RTOS | Real-Time Operating System |
| SID | Serviço de Informação e Documentação |

| | |
|---|---|
| SISCAO | Sistema de Controle de Atitude e Órbita |
| SPG | Serviço de Pós-Graduação |
| STPA | System-Theoretic Process Analysis |
| SUT | System Under Test (Sistema sob teste) |
| Sw | Software |
| TDI | Teses e Dissertações Internas |
| TIOA | Timed Input Output Automata |
| TMTC | Telemetria e Telecomando |
| TO | Time-Out |
| TS | Especificação Técnica de Software |
| TT&C | Telemetry, Tracking & Command |
| V&V | Verificação e Validação |

## SUMÁRIO

**Pág**.

# 1 INTRODUÇÃO

Todo satélite, exceto os muito simples, utilizam-se de sistemas de controle inerciais para controle de sua atitude e/ou órbita, sendo este sistema parte do Módulo de Serviço de um satélite. O Módulo de Serviço consta de uma plataforma de serviços básicos necessários para a operação no espaço, contendo os subsistemas de: Gestão de Bordo; Controle de Atitude e Órbita; Propulsão; Térmico; Telecomunicação de Serviço; Energia; e Estrutural e Mecânico. (1,2,3,4)

O subsistema de controle de atitude e órbita de satélites, do inglês AOCS – *Attitude and Orbit Control System*, é um sistema crítico do Módulo de Serviço do satélite. Seu correto funcionamento é imprescindível para manutenção do satélite em órbita e o controle de apontamento da carga útil, antenas e painéis solares.

Sistemas de controle inerciais são sistemas de alta multidisciplinaridade podendo conter componentes e equipamentos de diferentes especialidades (ex. software, mecânica, elétrica, química, etc.) e diferentes arquiteturas de operação. Em sistemas de controle inerciais o controle da planta é realizado por equipamentos que interagem com o ambiente via estímulos físicos, e tais estímulos de/para o ambiente necessitam de uma correta conversão, interpretação e tratamento para a representação virtual correta do estado do satélite e atuação de controle (1,5,6). Em tais sistemas é comum a existência de componentes de diferentes fornecedores operando de forma integrada, e mais importante que o correto funcionamento das partes é o funcionamento do sistema de forma integrada. Com isso a questão de integração do sistema de controle se torna de alta importância, devendo o sistema completo de controle desempenhar seu objetivo corretamente frente a todos possíveis intempéries que os componentes podem passar, de forma isolada e conjunta, das propriedades emergentes da integração e considerando as diferentes decisões de design e implementação dos componentes desenvolvidos pelos diferentes fornecedores.

Os subsistemas de controle de órbita e atitude de satélites possuem, atualmente, a lógica das funções de controle implementadas por software. Dada a natureza crítica do subsistema, os módulos de software contam com mecanismos capazes de detectar estados anormais de operação e automaticamente acionar uma ação de recuperação ou mitigação dos efeitos da anomalia, como chavear para um modo de operação alternativo ou para um hardware redundante. O conjunto dos mecanismos e lógicas para recuperação é chamado de FDIR, do inglês *Fault Detection, Isolation and Recovery*.

Os mecanismos FDIR no subsistema de controle de atitude e órbita são de alta importância para o cumprimento de seu objetivo. O controle inercial incorreto do satélite pode levá-lo rapidamente a um estado de instabilidade não mais recuperável e, portanto, à perda completa da missão e da espaçonave e por essa razão precisam ser verificadas cuidadosamente em nível de sistema. (4).

Um estudo de 2008 (7), no qual falhas de 129 diferentes satélites foram analisadas, mostrou que a maior parte das falhas em satélites que chegaram a entrar em órbita ocorreu no sistema de controle de atitude e órbita, como mostra a Figura 1.1.

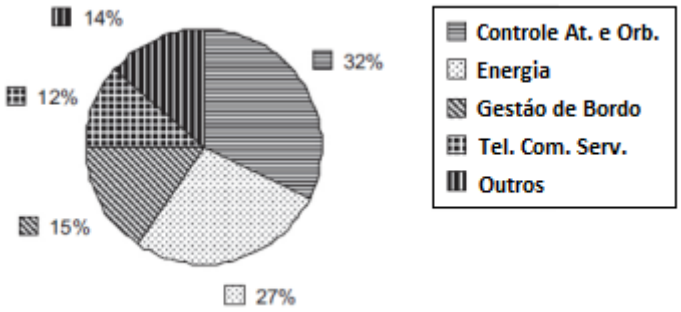


Figura 1.1. Falhas por subsistemas de satélites
Fonte: Adaptado de (7)

Dentre as falhas dos sistemas de controle de atitude e órbita não há predominância de um componente específico causador das falhas, como pode ser visto na Figura 1.2.

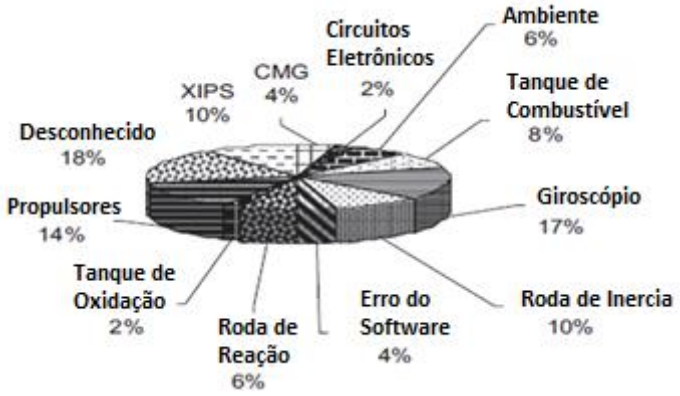


Figura 1.2. Falhas por componente

Fonte: Adaptado de (7)

O estudo (7) inicia afirmando que muitas espaçonaves mesmo com funções cuidadosamente projetadas e testadas para cumprir seu tempo de missão sofrem cedo de falhas não recuperáveis em órbita, enquanto que outras, mesmo sofrendo falhas severas são capazes de exceder seu tempo de vida quando procedimentos efetivos de recuperação de falhas são aplicados. E termina concluindo que além dos sistemas de recuperação convencionais a evolução de mecanismos FDIR pode aumentar consideravelmente a habilidade autônoma de detectar e recuperar falhas ocorrendo em órbita.

Casos reais exemplificam o efeito catastrófico da falência do serviço de controle inercial em sistemas espaciais, como apresentados na Tabela 1.1 (8,9).

Tabela 1.1. Alguns Casos de falência das capacidades de sistemas de controle inerciais em aplicações espaciais

| **Missão** | **Causa** | **Efeito** |
|---|---|---|
| Ariane 501 (1996) | “O relatório do acidente descreve o que foi chamado de “causa primaria” como a completa perda de guiagem e informação de atitude 37s depois da sequência de ignição dos motores principais (30 segundos depois de levantar voo)” (10) | Destruição do foguete e sua carga |

Continua

Tabela 1.1 - Conclusão

| | | |
|---|---|---|
| Mars Climate Orbiter (1999) | “perdida quando entrou na atmosfera de Marte em uma trajetória mais baixa que esperada. O conselho de investigação identificou o que chamou de causa “raiz” do acidente como a falha em usar unidades métricas na codificação de um arquivo de software solo usado nos modelos de trajetória (11). Dados de desempenho dos propulsores estavam em unidades inglesas” | Perda da sonda e missão |
| Mars Polar Lander (1999) | “Embora a causa da perda do MPL seja não conhecida, o cenário mais provável é que o problema tenha ocorrido durante a sequência de entrada, abertura e aterrissagem, quando as três pernas de aterrissagem eram para ser armadas de sua posição guardadas para posição de pouso.” (12)(13), “o software desligou os motores e o módulo caiu em queda livre até a superfície, impactando a uma velocidade de 22 metros por segundo e por isso destruído” | Perda da sonda e missão |
| Titan/ Centaur /Milstar (1999) | “Uma constante incorreta em filtro no eixo de rolagem zerou os dados de rolagem, resultando na perda do controle da rolagem e consequentemente controle de guinada e arfagem … colocando o satélite Milstar em uma incorreta e não útil órbita elíptica baixa” | Perda da missão |
| Genesis (2004) | “a causa aproximada ou direta do acidente foi que os sensores G-switch foram invertidos em orientação, por um *design* errôneo, e foram incapazes de realizar o sensoriamento da desaceleração da cápsula de retorno de amostras durante entrada atmosférica e iniciar desdobramento do paraquedas” | Perda da sonda, perda parcial das amostras científicas |

Fonte: Adaptado de (8,9)

O estudo e avanço de mecanismos FDIR para sistemas de controle de atitude e órbita de satélites se mostra de grande importância para a área espacial nos dias de hoje (7,14,15). E em consequência, atenção especial deverá ser dada à

verificação e validação dos mecanismos FDIR. O estudo (15) conclui que nos próximos anos deverá ser devotada mais atenção para o processo de desenvolvimento dos mecanismos FDIR, arquitetura, validação e testes, do que um maior incremento nas capacidades dos mecanismos existentes, os quais atendem aos requisitos de disponibilidade das missões.

Neste cenário temos as metodologias de testes, que visam disciplinar o processo de teste e proporcionam melhor qualidade ao produto final, enquanto facilitam o planejamento e a execução de cronogramas. Metodologias e técnicas de teste baseado em modelos, do inglês MBT – Model Based Testing, empregam métodos semi-formais, tais como modelos de estado, para modelar o comportamento esperado do sistema alvo de teste. Apesar de consideradas onerosas, seu uso tem se mostrado eficaz para a geração automática de casos de testes e a sistematização dos processos de V&V de sistemas críticos intensivos em software (16,17,18,19,20,21).

Visto a abrangência de disciplinas dos sistemas AOCS, o escopo do presente trabalho de mestrado se restringe à verificação e validação dos mecanismos de FDIR implementados no software do controlador de um Sistema AOCS, considerado um aplicativo de missão embarcado no computador de bordo da plataforma espacial. Questões de verificação e validação de equipamentos, do software de sensores ou atuadores e adversidades do compartilhamento de subsistemas em um único computador de bordo, ex. AOCS e Gestão de Bordo, não serão abordadas neste trabalho.

### 1.1. Objetivo da Dissertação

O objetivo desta dissertação é avaliar o potencial de duas metodologias de teste baseado em modelos para especificar casos de testes para verificação e validação de FDIR de Sistemas de controle de atitude e órbita para satélites. As duas metodologias de testes avaliadas são: CoFI - Conformidade e Injeção de Falhas (18); e InRob – Interoperabilidade e Robustez de Sistemas Intensivos e Software (17). Os aspectos avaliados são: (i) a adequação das metodologias de teste em verificar e validar o comportamento esperado de

mecanismos FDIR nas funções críticas realizadas pelo software de controle; e (ii) as comonalidades e complementariedades das metodologias na criação de casos de teste de conformidade e de interoperabilidade com foco na verificação e validação de requisitos de *safety*.

## 1.2. Metodologia

A realização desta dissertação apoiou-se na análise teórica da aplicabilidade e na experimentação prática das metodologias.

Na análise teórica da aplicabilidade avaliou-se as características das metodologias considerando suas especificidades face a conceitos de teste de software e das recomendações de guias e normas espaciais.

Na experimentação, um protótipo de AOCS com mecanismos de FDIR foram implementados e as metodologias CoFI e InRob foram aplicadas parcialmente. Para a realização deste experimento foi necessário o desenvolvimento de diversos artefatos de software.

A Figura 1.3 apresenta as atividades realizadas no contexto desta dissertação.

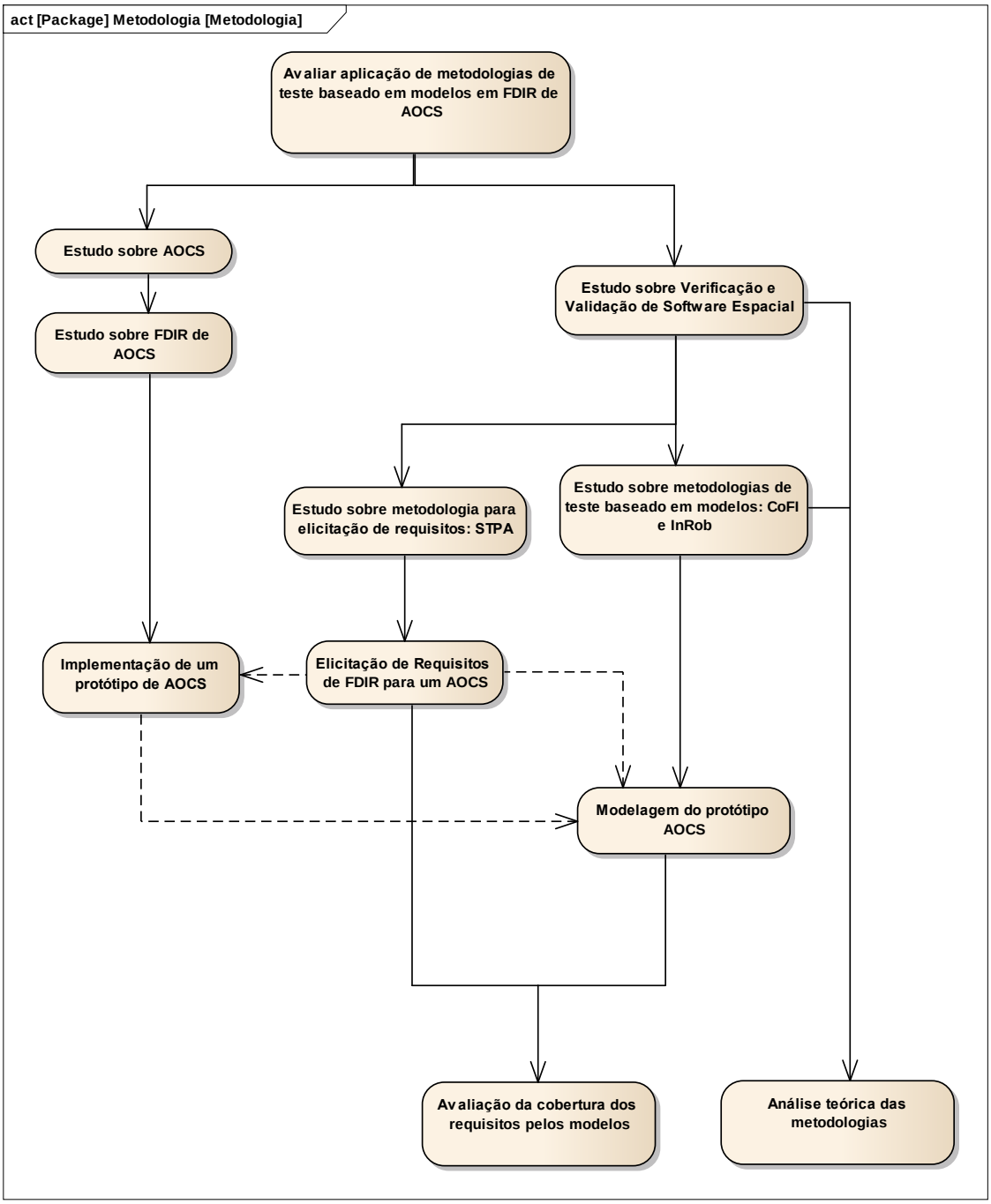


Figura 1.3. Atividades da dissertação

Inicialmente foram estudados os conceitos de FDIR de AOCS, como também conceitos sobre Verificação e Validação de software espacial.

Posteriormente as metodologias CoFI e InRob foram estudadas, e uma análise teórica das capacidades das metodologias foi realizada.

Seguindo, foi realizado o experimento prático que necessitou de desenvolvimento de artefatos de base para a aplicação das metodologias, consistindo em: um Protótipo AOCS e uma especificação de requisitos.

A metodologia STPA - *System-Theoretic Process Analysis* (22) foi estudada e utilizada para apoiar a elicitação de requisitos de *safety* para um sistema AOCS.

Após foram gerados os modelos do protótipo AOCS permitindo uma avaliação prática da aplicação das metodologias e a conclusão desta dissertação.

## 1.3. Organização da Dissertação

A dissertação está organizada da seguinte forma:

- Capítulo 2: Apresenta a revisão bibliográfica sobre AOCS, FDIR de AOCS e Verificação e Validação de software espacial. Inicialmente é feita uma contextualização de sistemas de controle inerciais, e apresentada uma revisão das técnicas de verificação e validação deste tipo de software, de acordo com normas espaciais. Em seguida, são apresentadas as formas de verificação e validação de FDIR de AOCS.

- Capítulo3: Apresenta conceitos sobre teste baseado em modelos e as metodologias de teste CoFI e InRob, utilizadas neste trabalho para especificação de testes de conformidade e de interoperabilidade, respectivamente. A técnica STPA, utilizada para elicitar requisitos de segurança (*safety*, em inglês) de um sistema AOCS, que envolvem os mecanismos FDIR, é também apresentada.

- Capítulo 4: Apresenta uma análise da aplicabilidade das metodologias de teste, baseada na análise teórica de suas capacidades.

- Capítulo 5: Apresenta o experimento prático. Inicialmente são apresentados os artefatos criados para a elicitação de requisitos para um sistema AOCS, de acordo com a metodologia STPA, que identifica

ameaças e trata questões de dependabilidade e *safety*. São apresentados também o Protótipo de AOCS e os modelos CoFI e InRob gerados. Por fim uma avaliação dos modelos gerados contra os requisitos é feita.

- Capítulo 6: Traz a conclusão do trabalho, avalia os resultados obtidos e aponta trabalhos futuros.

# 2 CONCEITOS DE SISTEMAS DE CONTROLE INERCIAIS E DE VERIFICAÇÃO E VALIDAÇÃO

O trabalho desenvolvido nesta dissertação, de forma geral, encontra-se na área de conhecimento de Engenharia de Software, e tem sua contribuição em Verificação e Validação de software espacial. Entretanto, concentra-se no software de um Sistema de Controle de Atitude e Órbita, o qual apresenta particularidades próprias de outras disciplinas, especialmente da Engenharia de Controle. Os trabalhos nesta dissertação não incluem pesquisa sobre desenvolvimento, verificação e validação dos aspectos da disciplina de engenharia de controle, que também são parte essencial do desenvolvimento de um sistema de controle de atitude e órbita. Por isso será apresentado neste capítulo somente uma breve revisão bibliográfica sobre verificação e validação da engenharia de controle para correto entendimento do escopo deste trabalho.

Este capítulo descreve conceitos de Sistemas Inerciais de modo geral, de Sistema de Controle de Atitude e Órbita para satélites, discorre sobre detecção, isolação e recuperação de falhas em tais sistemas, o que é chamado de mecanismos de FDIR e apresenta recomendações e conceitos de verificação e validação de sistemas de controle e de software em aplicações espaciais.

## 2.1. Sistemas de Controle Inerciais

Sistemas de controle inerciais são sistemas que controlam o deslocamento e a orientação de corpos no espaço. Esta categoria de sistemas está presente em diversas máquinas utilizadas pelo homem, como por exemplo: Satélites; Foguetes; Aviões; Submarinos; Navios; e Robôs. De forma mais experimental podemos encontrar estes sistemas também em automóveis.

A operação básica de um sistema de controle inercial consiste em adquirir informações de posição e movimento do corpo por meio de sensores, computar as ações de controle baseado no estado estimado e estado desejado para o corpo, e executar as ações de controle com uso de atuadores, que geram efeitos físicos de forças, em busca de levar o corpo ao estado desejado. Estes

sistemas de controle são cíclicos em sua atividade, tendo sempre uma malha fechada de atuação com o meio externo ao corpo.

De forma clássica sistemas de controle inerciais também são chamados de sistemas GNC – Guidance, Navigation and Control – pelas suas funcionalidades principais de controle, ilustradas na Tabela 2.1. “Um sistema que simplesmente indica a posição e velocidade de um veículo é um sistema de navegação. Se o sistema de navegação for colocado em uma malha fechada com os controles do veículo pelo computador de guiagem para controlar a posição e velocidade do veículo, ele é um sistema de Guiagem (e Controle) [GNC]” (23). “Porém os sistemas de controle inerciais atuais utilizam comumente mais funções para a realização de seus objetivos de controle, por exemplo mudança de modos de controle (interno ao controlador ou de sensores e atuadores), monitoramento do sistema de controle e situação da planta, atualização dos modelos, detecção de falhas, isolamento e recuperação” (5).

Tabela 2.1. Funções principais de um sistema GNC

| **Função** | **Descrição** |
|---|---|
| Guiagem *Guidance* | Refere-se a determinação da posição e orientação desejados (alvos) para o corpo. Consta de continuamente corrigir o vetor da velocidade do centro de massa do veículo, para que o veículo chegue a um ponto específico no espaço e tempo (23). |
| Navegação *Navigation* | Refere-se a determinação do estado estimado de um corpo frente as medidas dos sensores. |
| Controle *Control* | Refere-se a determinação das ações de controle para o instante frente as entradas de guiagem e navegação. |

Fonte: Adaptado de (5,23,24)

A Figura 2.1 apresenta a estrutura geral de um Sistema Controlado. O corpo ao qual o sistema de controle inercial controla é chamado de *planta* e em conjunto com ela forma o chamado *sistema controlado*. Os sistemas controlados têm como entrada os seus *objetivos de controle* e como saída o seu *desempenho de controle*. “Os operadores da planta controlada tem objetivos específicos. O propósito dos objetivos é ter um sistema de controle que dê à planta controlada um desempenho de controle especificado, mesmo com as interações com o seu ambiente externo.” (5).

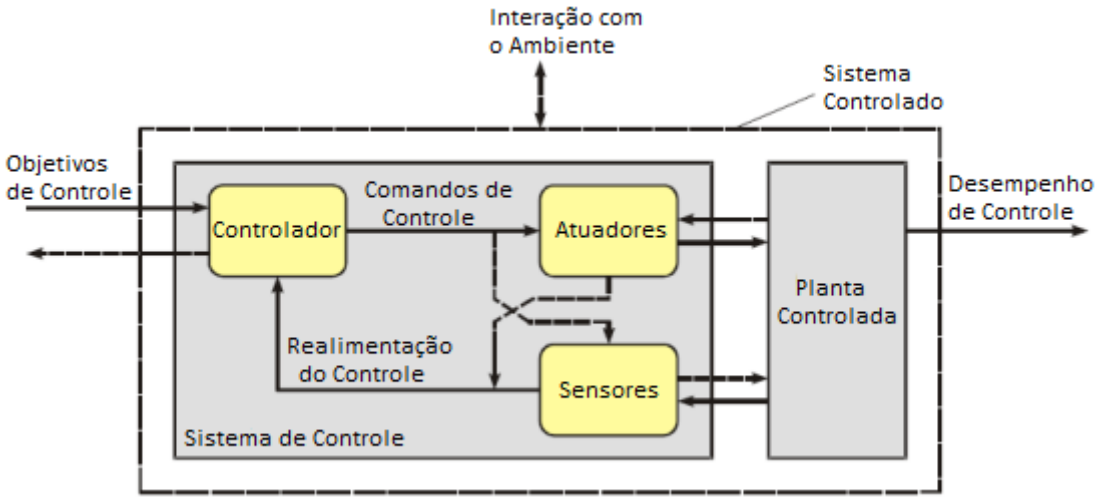


Figura 2.1. Estrutura Geral de um Sistema Controlado

Fonte: Adaptado de (5,24)

Os *objetivos de controle* são as referências de controle para o sistema controlado, e podem variar em abstração, podendo ser objetivos de alto nível, como realizar uma manobra ou adquirir um alvo, até ser comandos de baixo nível como valores de *setpoints* para as variáveis de estado do sistema. O *desempenho de controle*, por sua vez, depende dos *objetivos de controle* e visam caracterizar qualitativamente a realização destes objetivos. Várias podem ser as métricas na análise do desempenho de controle, ex. precisão, exatidão, agilidade de apontamento, amortecimento, etc., pertinentes ao objetivo, porém a mais básica característica pertinente a sistemas de controle é a estabilidade do sistema que diz respeito a sua capacidade de se manter controlável, “habilidade de um sistema submetido a distúrbios externos, entre

limites, de permanecer indefinidamente em um domínio limitado em torno de uma posição de equilíbrio ou de uma trajetória de equilíbrio" (6).

Na questão de estabilidade, destaca-se ainda a característica de **robustez** do sistema controlado, por estar relacionada a capacidade de manter estabilidade. A robustez representa a "habilidade de um sistema controlado de manter algumas características de desempenho, incluindo estabilidade, na presença de incertezas da planta, sensor, atuador e/ou ambiente" (6), sendo estas incertezas aproximações de projeto, variações de parâmetros de funcionamento dos equipamentos de controle e condições do ambiente ou falhas de operação dos equipamentos de controle. De forma mais abrangente podemos caracterizar não somente a estabilidade e robustez, mas todas as características de **dependabilidade** (25) de um sistema de controle inercial, como recomendado desde 1971 no guia de GNC da NASA: "Com o advento de sistemas aeroespaciais de larga escala, projetistas reconheceram a importância de especificar e realizar requisitos de projeto adicionais aos requisitos clássicos funcionais e de ambiente. Esses requisitos "adicionais" incluem **produtibilidade, safety, confiabilidade, qualidade, e manutenabilidade**. Essas características foram identificadas, cresceram em importância, e tornaram-se disciplinas próprias. Atualmente, não é concebível que qualquer especificação de requisitos de um sistema ou equipamento aeroespacial seja formulada sem consideração a essas características." (26).

Requisitos de dependabilidade visam garantir o correto funcionamento do sistema durante sua operação, ultimamente fornecendo as capacidades de funcionamento correto do sistema para seu sucesso durante seu tempo de missão.

Sistemas de controle modernos, como de veículos não tripulados ou sondas interplanetárias, também se caracterizam por sua capacidade de autonomia (27,28,29). Porém a capacidade de autonomia de espaçonaves orbitais é tradicionalmente baixa, sendo estas dependentes dos objetivos e decisões definidas pelo homem para a operação do sistema de controle. Esta característica não será abordada neste trabalho.

## 2.2. Sistemas de Controle Inerciais de Satélites

O sistema de controle inercial de satélites é chamado em português de SISCAO - Sistema de Controle de Atitude e Órbita, ou comumente pela sigla em inglês AOCS - Attitude and Orbit Control System. “O AOCS é um dos subsistemas de um satélite. É frequentemente o mais complexo subsistema do satélite. Sua função é controlar a atitude e órbita do satélite. Essa função é critica visto que sem adequado controle de atitude e órbita o satélite irá se tornar incapaz de cumprir os objetivos da missão” (30).

Nos satélites mais antigos, o subsistema AOCS era dividido em duas partes: (i) ADCS – Attitude Determination and Control Subsystem – que estabiliza o veículo e o orienta para sua direção especificada durante a missão, apesar de torques atuantes por distúrbios externos” (1) e internos; e (ii) GNC – Guidance, Navigation and Control – que determina a posição e velocidade do satélite ou, equivalentemente, seus elementos orbitais como uma função no tempo e ajusta a órbita para se aproximar das condições predeterminadas” (7), sendo o responsável por manobras orbitais, ou posicionamento fino de uma espaçonave no espaço. No sistema GNC da Space Shuttle, o processo [do subsistema GNC] é dividido em três passos: (a) cálculo da localização pretendida da espaçonave pelo software de guiagem, (b) estimativa da posição atual da espaçonave pela navegação e, (c) transporte da espaçonave em órbita para a localização requerida pelo controle de voo (31).

Atualmente, os sistemas AOCS além de integrarem ambas as funcionalidades de ADCS e GNC rumam a integração também com o subsistema de Gestão de Bordo, compartilhando de ambiente computacional único, graças ao avanço na área de circuitos integrados e software embarcado. Alguns exemplos de computadores de bordo com a filosofia de System-on-a-Chip (32) proporcionam uma capacidade de integração da computação de diversos subsistemas. Na área de software embarcado espacial o desenvolvimento de plataformas de software, ex. RTOS, e reuso de funcionalidades comuns, ex. PUS (33), geram modularização, legado de operação, facilidade de desenvolvimento e de verificação e validação dos aplicativos específicos da missão em um satélite.

Considerando seus componentes físicos, um sistema AOCS é composto por sensores, atuadores, um controlador e cabeamento de conexões, como ilustrado na Figura 2.1. O controlador contém a lógica de controle e interage com a planta controlada por sensores e atuadores. Os atuadores são dispositivos que convertem comandos do controlador em efeitos físicos na planta controlada, já os sensores são dispositivos que medem o estado da planta controlada e os provê como entrada em realimentação para o controlador (5).

A computação do controle inercial do satélite inclui hardware, software e operações humanas. Sua operação pode ser distribuída entre o segmento espacial e o segmento solo (5). O segmento solo supervisiona a operação do sistema AOCS via telemetria, e envia comandos ao mesmo que podem, por exemplo, alterar um modo, alterar objetivos de controle, comandar uma atuação de manobra orbital, etc. O segmento espacial realiza os *objetivos de controle* e executa os comandos pré-programados ou vindos do segmento solo.

O controlador é o componente central do sistema de controle nos satélites, nele é executado o software de controle que integra logicamente os diversos componentes, define o modo de operação, executa o controle da espaçonave, e além disso, contém a computação da funcionalidade de detecção, isolamento e recuperação de falhas, portanto é peça central na operação e recuperação do sistema, visto possível falha em qualquer um de seus componentes.

### 2.2.1. O Software Aplicativo do AOCS

O software do controlador do AOCS tem a função principal de comandar o controle inercial do satélite com o desempenho e dependabilidade especificada para a missão. O software contém o ***código das leis de controle*** e as ***funções complementares*** para o correto funcionamento do sistema. O “código dos algoritmos das leis de controle representa um pequeno subconjunto em [código de] uma aplicação complexa de controle (em controle de satélites estes códigos representam em torno de 20-30%). As funções complementares incluem técnicas para manusear o restante da aplicação de controle

embarcado (como interações com operador externo, detecção e recuperação de falhas, gerenciamento dos equipamentos, etc.)” (30).

As características de dependabilidade estão distribuídas em todo o código aplicativo do controlador, tanto na parte de algoritmos de controle quanto na parte das funções complementares da aplicação. O mau funcionamento de qualquer uma das partes do aplicativo pode piorar as características de dependabilidade do controle, enquanto que, uma implementação com o mínimo de defeitos e que adequadamente trate falhas, tanto do controlador quanto dos demais equipamentos de controle, aumenta o nível de dependabilidade do sistema e proporciona menor risco de perda da missão por culpa do sistema de controle.

## 2.3. Fault Detection, Isolation and Recovery

Em um satélite, o FDIR é o conjunto de mecanismos que funcionam de forma distribuída e hierárquica, implementados por hardware e/ou software, responsáveis por detectar, isolar e recuperar o satélite de situações de falha em sua operação. Em um sistema com alto nível de complexidade, como no caso de subsistemas AOCS, a funcionalidade do FDIR aumenta em importância pelo fato de sua responsabilidade de supervisionar a operação individual de componentes e do sistema integrado como um todo, mantendo-o em operação frente à miríade de cenários de perturbações e falhas que o sistema pode sofrer. As falhas podem ser tratadas local ou globalmente dependendo do impacto e das medidas de recuperação necessárias dos subsistemas do satélite (14,34,35).

Segundo (15,35), o desafio da construção de arquiteturas de FDIR de satélites é lidar com as falhas localmente, e no caso de insucesso tratá-las em um nível superior. Por exemplo: no caso de um erro de comunicação em um equipamento AOCS, o controlador do barramento pode inicialmente tentar novamente uma comunicação. Caso ocorra sucesso na nova tentativa, uma sinalização do evento é gerada e a operação pode proceder normalmente. Caso contrário, um nível de FDIR superior pode ser acionado para, por

exemplo, conferir o estado do equipamento, alterar seu modo de operação ou chavear para um equipamento redundante.

Para a discussão das características dos mecanismos de FDIR adotamos como terminologia de referência aquelas apresentadas pelo *SAFEPROCESS Technical Commitee of International Federation of Automatic Control IFAC e em* (36,37,38), que definem:

- **Funcionamento normal:** Um sistema está em funcionamento normal quando seus estados, suas entradas e saídas estão dentro da faixa de valores nominais.

- **Falha (*Fault*):** é um desvio de uma variável observada ou um parâmetro calculado para fora da faixa de valores considerados nominais para o processo e equipamentos.

- **Falência (*Failure*):** significa perda total ou significativa das capacidades de uma função do sistema.

- **Modelamento da falha:** determinação de um modelo que descreve os efeitos específicos de uma falha.

- **Resíduo:** valor que exprime a incoerência entre as informações ou valores aferidos e as fornecidas por um modelo teórico.

- **Detecção de falha:** determinação da presença de uma falha no sistema e instante de sua ocorrência.

- **Isolação[1] de falha:** determinação do tipo e localização da falha.

---

1 – O termo em inglês *Isolation* é traduzido de duas formas na literatura em português, como isolamento e isolação. Este trabalho adotou a tradução presente em trabalhos acadêmicos do INPE de Isolação (38).

- **Identificação de falha:** Determinação do tamanho e do comportamento temporal de uma falha.

- **Diagnóstico:** Determinação do tipo, tamanho, localização e tempo de detecção da falha. Inclui as tarefas de isolação e identificação da falha.

As funções de FDIR espacial constam de: Detecção; Isolação (e Identificação: Diagnóstico); e Recuperação.

A *detecção de falhas* pode ocorrer pela conferência de eventos discretos, por exemplo pelo monitoramento da palavra de status de equipamentos, sinalização de falha no processador, monitoramento de time-outs em comunicações e comandos, conferência de variáveis contra valores limites. E também pode ocorrer por análise de dados dinâmicos que resultam em resíduos que indicam falhas, por exemplo entre variáveis e parâmetros contra modelos de comportamento esperado no tempo (temporal) ou conferência cruzada de medidas (direta). Esta última forma é de grande importância no monitoramento de processos e sistemas dinâmicos, como é o caso de sistemas de controle inerciais AOCS.

De forma geral, a *detecção e diagnóstico de falhas* (DDF) pode ser descrita como em (39) por: um sistema $\boldsymbol{G}$ estimulado por uma entrada conhecida $\boldsymbol{u}$, ruído estocástico $\boldsymbol{v}$, perturbação determinística $\boldsymbol{w}$, e um sinal exógeno $\boldsymbol{f}$ representando o estímulo de uma falha. A arquitetura de diagnóstico de falha consiste de duas partes: um gerador de resíduo $\boldsymbol{F}$ e uma função de decisão $\boldsymbol{\bar{\sigma}}$. O gerador de resíduo usa a entrada $\boldsymbol{u}$ e a medida da saída $\boldsymbol{y}$ para produzir o residual $\boldsymbol{r}$, que carrega a informação de ocorrência de uma falha. A função de decisão $\boldsymbol{\bar{\sigma}}$ avalia o residual e determina o tipo de falha ocorrida caso ela exista, como representado na Figura 2.2.

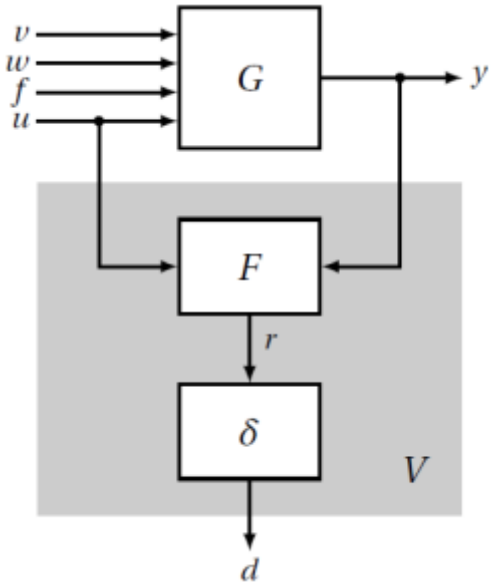


Figura 2.2. Arquitetura básica de um sistema de detecção e diagnóstico de falhas

Fonte: Adaptado de (39)

Em FDIR de sistemas aeroespaciais a mesma estrutura de DDF é utilizada, como na Figura 2.3 (14), ou estendida, como em (40) e (15).

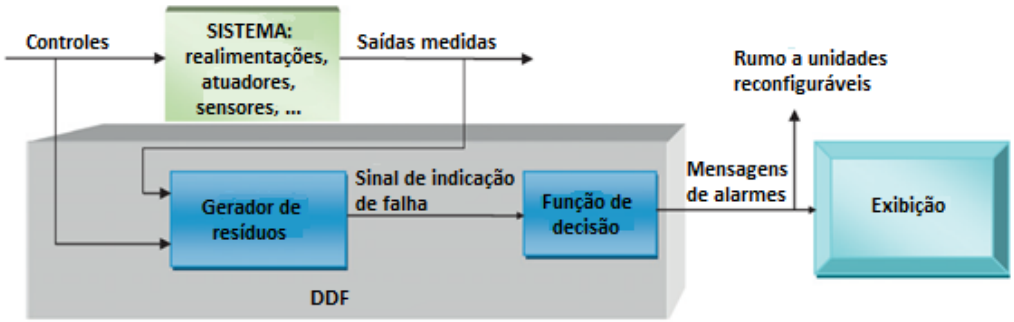


Figura 2.3. Estrutura básica de um DDF em aeroespacial

Fonte: Adaptado de (14)

A detecção e diagnóstico da falha em sistemas dinâmicos podem ser realizadas utilizando diversos métodos diferentes (41,42,43), como: *métodos quantitativos*, que usam modelos com equacionamento das dinâmicas esperadas do equipamento ou processo sendo monitorado; e *métodos qualitativos* onde modelos relacionando causas e efeitos de comportamento

são utilizados como base do monitoramento do sistema. Em ambos os casos pode ocorrer uma discrepância do comportamento do sistema no tempo comparando-se as saídas do sistema com o esperado pelo(s) modelo(s). Essa discrepância é referida por resíduo, que sinaliza uma possível falha no sistema, e o comportamento dessa diferença no tempo pode ser usado para o diagnóstico do problema.

A *recuperação* tem o objetivo de retirar ou mitigar a falha e seus efeitos no sistema. As ações de recuperação são dependentes da missão e são definidas por falhas ou pelos seus efeitos, com ações para falhas não identificáveis também existindo se for o caso. Em aplicações espaciais podemos citar duas estratégias de recuperação (15):

- Estratégia de Meio-Satélite: quando uma falha é detectada não há a etapa de isolação e a recuperação reconfigura completamente o satélite ativando os equipamentos redundantes. Neste caso, a equipe de operação do satélite em solo, é responsável por diagnosticar a falha e corrigir a anomalia. Esta estratégia é bem simples, requer um menor esforço de validação e mantém o satélite em um modo seguro, sendo aplicável a missões que requerem índices de disponibilidade baixos, como o caso de alguns satélites científicos.

- Estratégia de FDIR hierárquica: Tem como princípio manter o modo de operação do satélite de forma a propiciar uma alta disponibilidade da missão. Neste caso, as falhas são tratadas no nível mais baixo possível, com uma evolução gradual aos níveis superiores caso seja necessário. A isolação e recuperação são realizadas por processos locais, e os impactos mitigados mantém a operação mesmo que de forma degradada. As correções de dados são feitas localmente (ex. bit flip, CRC), o chaveamento para equipamentos redundantes é feito de forma individual, e o chaveamento para modo de segurança e interrupção da missão só ocorre em casos de falhas específicas críticas ou quando as recuperações locais não resolvem o problema.

### 2.3.1. FDIR para AOCS

Considerando exclusivamente subsistemas AOCS de satélites, alguns mecanismos de detecção de falhas são (15,41):

- **Conferência de Equipamentos**: quando o equipamento possui a funcionalidade de sinalizar falhas por sinais elétricos de status ou palavras de status.

- **Conferência de Transmissão de Dados**: quando protocolos de comunicação detectam falhas de transmissão de dados, como time-outs, checksums, acknowledges de comandos, etc.

- **Conferência de Consistência de Dados**: quando o dado útil é conferido para indicar falhas nos equipamentos e/ou perturbações na dinâmica dos equipamentos e satélite. Algumas conferências de consistência de dados são:

    - **Consistência de Dados de Equipamentos**: Confere a continuidade das medidas providas por um equipamento, normalmente sensor, mas também podem ser valores de operação de atuadores. Uma ruptura ou congelamento das medidas pode sinalizar uma falha.

    - **Consistência de Medidas**: Conferência cruzada entre sensores redundantes (*hardware redudancy*), ou conferência cruzada entre diferentes sensores (*analytical redundancy*), computando-se o resíduo das medidas. Uma discrepância entre medidas sinaliza uma falha em um dos sensores.

    - **Consistência de Comando**: Conferência dos efeitos da execução de um comando, normalmente validando o comportamento da cadeia de comando e atuação.

- **Monitoramento do Desempenho de Controle**: Monitora parâmetros de desempenho de controle e erro de atitude, uma anomalia nestes parâmetros sinaliza uma falha na operação do controle.

## 2.4. Verificação e Validação do Controle

Esta seção traz um resumo sobre verificação e validação dos aspectos de desempenho de controle, bem como do código aplicativo incluindo o código das leis de controle e as funções complementares do aplicativo. Os conceitos apresentados aqui são baseados nos guias de desenvolvimento de sistemas espaciais da *European Cooperation for Space Standardization* (ECSS) da Agência Espacial Europeia. Em geral, os conceitos de verificação e validação de controle e de software espacial serão apresentados, para situar corretamente o escopo de teste de software deste trabalho, principalmente porque diferentes testes são necessários para cobrir as diferentes disciplinas: Engenharia de Controle e Engenharia de Software.

A atividade de verificação e validação da Engenharia de Controle de sistemas de AOCS pode ser sumarizada em tarefas, como apresentado em (5):

- definição da estratégia de verificação e validação do controle (incluindo especificação dos requisitos do ambiente de testes),
- verificação preliminar do *desempenho de controle* por análise ou prototipação,
- verificação funcional e de desempenho por análise,
- verificação e validação final do sistema controlado (hardware, software e operação humana) por testes de integração tipo hardware-in-the-loop,
- validação do comportamento do sistema controlado em voo.

A atividade de verificação e validação do sistema de controle confere as funcionalidades e desempenho de controle do sistema durante todas suas

fases de desenvolvimento, depois na operação do sistema integrado para a aceitação do mesmo, e ainda com o sistema na fase de operação do satélite.

Segundo (4), a verificação dos componentes de hardware e software pode ser feita por quatro métodos diferentes: (i) análise, (ii) simulação em software, (iii) teste com hardware-in-the-loop e (iv) por testes em bancadas específicas como mesas de mancal a ar.

A Figura 2.4 apresenta os aspectos verificados e os métodos aplicados nas atividades de verificação de um sistema AOCS. Observa-se que os sistemas de controle inerciais atuam em malha fechada com o meio ambiente e os testes dos componentes de software se apoiam, em grande parte, em simulações do ambiente espacial.

| Verificação | Aspectos | Métodos |
|---|---|---|
| Hardware dos equipamentos AOCS | Desempenho especificado | Testes realizados pelo fornecedor |
| | Funções no satélite | Testes realizados após integração, qualitativamente |
| Interfaces de Hardware | Atribuição dos eixos, alinhamento | Medição na espaçonave |
| | Polaridade dos eixos | Teste na espaçonave |
| | Térmico | Análise e Teste Vacuo-Termico |
| | Elétrico | Testes e medições na espaçonave |
| EMC, EMI | Entre equipamentos AOCS | Testes e medições na espaçonave |
| | Entre AOCS e satélite | Testes e medições na espaçonave |
| | Possíveis configurações de hardware (redundância) | Testes e medições na espaçonave |
| Interfaces de Software | Protocolos, temporização, conversões | Simulador, testes na espaçonave |
| | AOCS TMTC | Simulador |
| Processos AOCS | | |
| Operações | Sequências operacionais | Simulador, Teste Hardware-in-the-loop |
| FDIR | Detecção de falhas, medidas de correção | Simulador, Teste Hardware-in-the-loop |
| Controle de Atitude | Estabilidade de processos | Análises, simulador, teste Hardware-in-the-loop, bancadas de teste |
| | Desvios de controle | Análises, simulador, teste Hardware-in-the-loop, bancadas de teste |
| | Curso e Duração dos processos | (Análises), simulador, teste Hardware-in-the-loop, bancadas de teste |

Figura 2.4. Verificação de um sistema de controle de atitude

Fonte: Adaptado de (4)

Para avaliar o desempenho do controle, Figura 2.5, o handbook de diretrizes sobre desempenho de controle da ECSS (45) indica três métodos de avaliar o desempenho do controle para sua verificação: (a) Experimentação, (b) Simulação numérica, (c) Análise de provisão de controle. Apesar das

diferenças no agrupamento dos métodos de teste dadas pelas fontes (4) e (45), podemos associar a equivalência dos grupos (i) a (c), (ii) a (b) e (iii,iv) a (a).

| | Propriedades de desempenho extrínsicas | Propriedades de desempenho instrínsicas |
|---|---|---|
| Propriedades de regime permanente | Erro de apontamento absoluto, estabilidade de apontamento, erro de medição absoluto, etc. | Trasmissão de ruído de medição, rejeição de distúrbios de torque externos, etc. |
| Propriedades de transição | Overshoot de atitude (entrando em um modo de controle a propulsão), tempo de tranquilização (fim do modo de controle a propulsão, modo de redução de velocidades), etc | Overshoot, tempo de resposta, tempo de acomodação, amortecimento, dacaimento logaritmo, etc. |
| Propriedades gerais | Consumo de combustível, média da iluminaçõa nos painéis,etc. | Estabilidade do sistema, Margens de estabilidade, robustez |

Figura 2.5. Exemplos de desempenhos de controle especificáveis

Fonte: Adaptado de (45)

Na Figura 2.5 pode-se observar que, algumas características de desempenho podem ser consideradas características de dependabilidade do controle, que neste caso, corresponde a robustez de aspectos do controle frente a condições inerciais momentâneas indesejadas do satélite, que aliado, por exemplo, a robustez de operação do software, fazem parte das características de dependabilidade do sistema AOCS como um todo, no cumprimento dos seus objetivos.

Resultados experimentais se valem do uso de prototipação ou equipamentos finais para testar as características de desempenho de um equipamento ou partes do sistema. Os testes são medições de uso e casos de testes representativos da operação do sistema de controle para a missão, que geram dados para apoio a design e verificação e validação do controle. Este tipo de teste pode ser combinado com simulação para fazer testes de simulação com hardware-in-the-loop.

Simulação numérica é utilizada para análise de alternativas desde conceitos e arquiteturas, refinar resultados de análises complexas e testar o controle em fases iniciais do projeto até fases finais de desenvolvimento de forma mais

detalhada, com modelos acumulando informações que se tornam disponíveis durante o desenvolvimento. Cada teste pode representar um cenário operacional de interesse para o desenvolvimento do controle. E devido ao crescente conhecimento adquirido de missões anteriores e exploração científica do espaço, as simulações são capazes de gerar resultados confiáveis com uma enorme flexibilidade de testes. São comuns simulações de pior-caso para teste dos casos limites dos parâmetros de operação e margens de controle, e simulações de Monte Carlo para avaliar distribuições de parâmetros tanto para design quanto para verificação do comportamento nas faixas de parâmetros do controle nos diferentes cenários de operação do controle.

A análise de provisão de controle utiliza a análise matemática para demonstrar a viabilidade do conceito de controle e as alternativas de arquitetura quanto a alcançar os objetivos de controle e satisfazer os requisitos de missão nas fases iniciais do desenvolvimento, para garantir que os requisitos da missão serão satisfeitos pelo *design* e nas decisões de projeto, e com a soma das contribuições de erro de controle por cada componente do sistema nas fases finais de desenvolvimento, sendo esta a análise de provisão de erros de controle.

Durante todo o desenvolvimento e a cada avanço no projeto do controle podemos dizer que é necessário a reverificação do desempenho de controle para assegurar que suas características continuam validas e aceitáveis mesmo visto a integração dos componentes (hardware, software, operador) e características intrínsecas aos mesmos. A validação do desempenho é feita bottom-up, onde cada componente do sistema de controle deve satisfazer seus requisitos e a integração dos componentes deve satisfazer os requisitos de maior nível hierárquico até serem satisfeitos os requisitos do sistema de controle como um todo.

## 2.5. Verificação e Validação de Software

Todo o código aplicativo AOCS, incluindo, tanto as funcionalidades de apoio ao sistema de controle quanto o código dos algoritmos de controle, por serem implementados em software, seguem as regras de verificação e validação de software embarcado. No código dos algoritmos de controle é preciso verificar tanto as características de controle quanto as características de implementação em software.

O processo de desenvolvimento de software pode ser dividido em nove grandes atividades, como agrupado em (46):

- **Requisitos de sistema relacionados a software:** estabelece os Requisitos Base (RB) de software no sistema espacial. Atividade de engenharia do sistema onde a especialidade de software apoia a engenharia de requisitos do sistema para definir as funções com seus desempenhos, e demais requisitos alocados a software, seus métodos de verificação e validação, restrições de desenvolvimento, etc.

- **Gerenciamento de software:** planejamento e organização da atividade de desenvolvimento de software, incluindo gerenciamento do ciclo de vida, revisões, gerenciamento de interfaces, provisões e margens e processo de desenvolvimento.

- **Requisitos de software e arquitetura de software:** elaboração da especificação técnica de software (TS) com os requisitos de software, derivados dos requisitos base, mais arquitetura ou modelo lógico funcional e comportamental de software, e demais modelos lógicos para apoio ao *design* de software.

- **Software *design* e implementação:** *design* do produto de software e sua implementação. Inclui codificação e teste de unidade e de integração do código de software em módulos.

- **Validação de software:** organização e execução das atividades de validação, validando o software quanto a sua especificação técnica (TS) e quanto os seus Requisitos base (RB).

- **Entrega e aceitação do software:** organização da atividade e entrega, instalação, treinamento e aceitação do software.

- **Verificação de software:** organização das atividades de verificação e execução da atividade, contemplando a verificação de todo o projeto, incluindo o produto de software.

- **Operação de software:** organização da atividade e operação do software, com apoio ao uso do mesmo.

- **Manutenção de software:** organização da atividade e manutenção de software, modificações de software, revisões de manutenção, migração de software e plano e execução do fim da operação do software.

É importante notar que não existe uma definição globalmente aceita sobre as atividades de verificação e validação, como explicado em (47): “deve ser notado que existe uma forte divergência de opinião sobre quais tipos de testes constituem validação. Algumas referências dizem que todos os testes são verificação e que validação é realizada quando requisitos são revistos e aprovados. Outras referências veem teste de unidade e integração como verificação e testes de mais alto nível como validação”. Não faz parte dos trabalhos de mestrado advogar sobre qual conceito é o mais correto, iremos usar os conceitos presentes no guia ECSS da ESA.

“*Validação* é definida como a confirmação, pelo provimento de evidencias objetivas que os requisitos para uma aplicação ou uso pretendido foram cumpridos. O processo de validação (para software) é o processo para confirmar se os requisitos foram corretamente e completamente implementados no produto final. Validação é, portanto, um teste “fim-a-fim” do produto.” (48).

Todos os requisitos devem ser validados, e cada requisito deve ter seu método, processo e critério de aceitação vinculado para ser validado. A validação pode utilizar qualquer método necessário e adequado para tal, havendo a preferência pelos guias da ESA por testes no produto final, como dito em (48). O guia (46) define que “validação deve ser realizada por testes”, mas ressalva que “caso possa ser justificado que não é possível a validação por testes, a validação pode ser realizada por análise, inspeção ou revisão de design.” O guia também define que para cada requisito de software, deve ser desenvolvido e documentado um conjunto de casos de testes (entradas, saídas e critério de aceitação) e um processo de teste, incluindo:

- Teste de estresse, limites e entradas singulares
- Teste da habilidade de isolar e mitigar os efeitos de falhas
- Teste da habilidade de operar com sucesso em um conjunto representativo do ambiente operacional
- Teste das interfaces externas incluindo limites, teste dos protocolos e testes dos tempos de comunicação
- Teste da interface homem máquina da aplicação

Mesmo a validação sendo um processo separado e dedicado, muito de seu trabalho pode se apoiar em resultados da verificação e teste da codificação para gerar evidências de que o software contém o requisito implementado corretamente. “Devido a crescente complexidade de software espacial, e com consequente explosão combinatória do comportamento do produto, as técnicas de verificação permitem a redução do esforço de validação. Como a validação representa uma atividade de alto custo, o plano de verificação e validação de software deve garantir um balanço de custo-benefício efetivo para o projeto” (48).

“O processo de *verificação* de software tem a intenção de confirmar que há especificação adequada e entradas para cada atividade e que as saídas das

atividades estão corretas e consistentes com as especificações e entradas. Este processo é concorrente com todas as atividades de software" (46). "Verificação é a confirmação, pelo provimento de evidencias objetivas que as exigências [de projeto] especificadas foram cumpridas" (48), ou seja, que as atividades do desenvolvimento de software estão completas e corretas para cada tipo de software sendo desenvolvido.

A expressão "para cada tipo de software sendo desenvolvido" indica que a verificação é dependente das características do software que influenciam as atividades de desenvolvimento. Como o processo de verificação tem o objetivo de verificar todas as atividades do desenvolvimento, ela depende das exigências necessárias, de cada atividade, para cada tipo de software. Por exemplo, cada software, ou componente de software, tem uma criticalidade, e cada nível de criticalidade acarreta exigências de desenvolvimento, como cobertura de código verificado. Se o software for de tempo-real faz-se necessário algumas tarefas, a especificação e *design* com análise de modelo de computação em tempo-real e escalonabilidade, e a atividade de verificação precisa verificar estas análises e a especificação destas características. Por isso é pedido que no planejamento da atividade de verificação seja definida a atividade baseando-se no tipo de software: "Atividades do ciclo de vida e produtos de software a ser verificados devem ser determinados baseados em análise de escopo, magnitude, complexidade, e criticalidade." (46).

A verificação pode ser realizada por diferentes formas (48), ex. testes, revisões, inspeção, análises, prototipação, auditorias, ferramentas específicas, listas de conferências, etc.. A verificação das atividades pode levar também em consideração não só aspectos técnicos da atividade, mas também aspectos gerenciais do desenvolvimento, como recursos utilizados, tempo de entrega, etc. se for de interesse para o projeto. Para cada atividade a ser verificada deve ser feita uma especificação dos itens a serem verificados. Um conjunto de itens de referência, não exaustivo, para cada atividade pode ser encontrado em (46) e (48).

### 2.5.1. Teste de Software

As tarefas de testes no desenvolvimento de software são distribuídas entre as atividades de implementação, verificação e validação, sempre com o intuito de demonstrar a corretude da implementação e encontrar os defeitos existentes. As atividades de testes tem grande importância no desenvolvimento de software espacial, sendo um método de verificação e validação largamente utilizado desde a geração da primeira unidade de software até a validação final do software e sistema espacial de mais alto nível.

Mesmo com as possibilidades de geração de código automática e modelo lógico de arquitetura executável, que fazem uso de testes mesmo antes da geração do código, iremos restringir a discussão aos testes padronizados e atualmente recomendados em (48) para o desenvolvimento de software espacial, os quais abordam os testes de características funcionais e não funcionais da implementação do software.

As características não funcionais de *Dependabilidade* nesse trabalho estão de acordo com a definição de (25), sendo eles: Disponibilidade – prontidão para o correto serviço; *Confiabilidade* – continuidade do correto serviço; *Safety* – ausência de consequências catastróficas para o usuário ou ambiente; *Integridade* – ausência de alteração imprópria do sistema; *Manutenabilidade* – habilidade de um processo de ser submetido a modificações ou reparos.

Ao longo do desenvolvimento do software, diferentes tipos de testes são adotados, de acordo com a fase de desenvolvimento do software: “Teste de unidade de software; teste de integração; teste de validação e teste de aceitação” (48). Testes específicos de validação podem ser subdivididos em *teste contra a especificação técnica de software*, *e teste contra os requisitos base*, sendo o último, o teste de qualificação do software.

As Figuras 2.6 e 2.7 (47) mostram a evolução dos testes aplicados ao software ao longo do seu desenvolvimento.

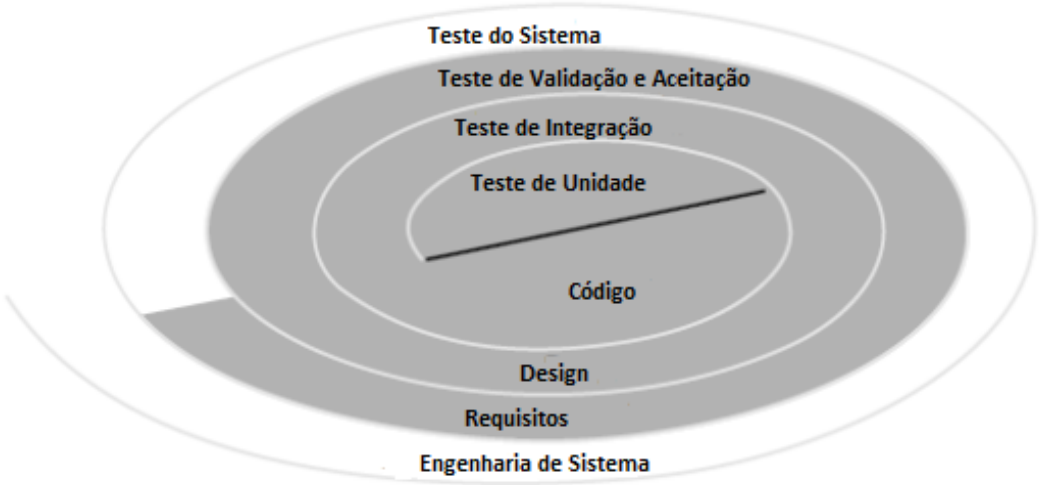


Figura 2.6. Evolução dos testes de software

Fonte: Adaptado de (47)

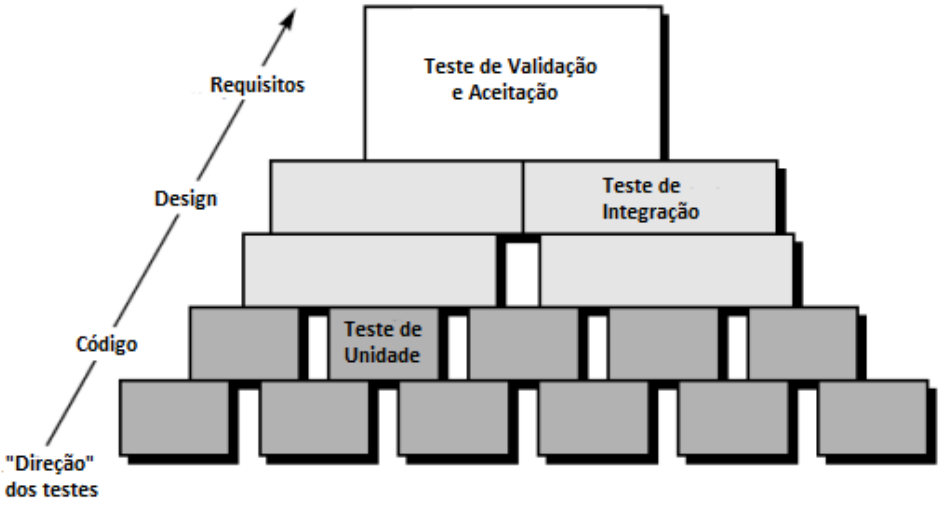


Figura 2.7. Hierarquia dos testes de software

Fonte: Adaptado de (47)

- **Teste de unidade:** Testes de unidade são feitos para a menor unidade codificada de software (47). Testes de unidade são executados o mais cedo possível para ganhar confiança suficiente no código (48). O guia (46) pede que toda unidade de software seja testada.

- **Teste de integração:** Os objetivos do teste de integração são: garantir a correta interação das unidades de software, considerando as interfaces externas, e que, de forma conjunta, interoperem de forma consistente como especificado. A integração consiste na montagem física e progressiva dos itens de software, correspondendo aos itens do design, até o software completo. (48)

  O teste de integração pode ocorrer de duas formas: (i) com a integração ocorrendo em um ambiente somente lógico, puramente de software, onde as funções se integram logicamente, referenciado como integração software-software; (ii) e com a integração ocorrendo já em ambiente protótipo ou real de operação, e portanto além das características lógicas do software se somam ao teste as influências do hardware alvo, referenciado por integração software-hardware. Normalmente testes de integração software-software ocorrem no inicio da integração. Porém, ultimamente é necessário que a integração ocorra com o hardware, considerando suas características no resultado do software.

- **Teste de validação (e verificação):** Testes de validação são testes para evidenciar a conformidade ou denotar defeitos do software frente as suas especificações. Testes de validação são basicamente testes caixa preta e da menor forma possível intrusivos na condução e observação dos resultados de testes.

  As necessidades para executar os testes de validação podem ser diferentes para cada caso. No caso de teste de software AOCS, por exemplo, a execução de testes de validação demanda ultimamente um ambiente de teste dedicado de simulação do ambiente espacial, para o sistema poder funcionar de forma completa.

  Testes de validação quanto aos requisitos base, são chamados de **testes de qualificação**. Testes de qualificação são testes contra os requisitos base de software, sendo executados com o produto de

software completo, usando as interfaces reais de operação, e em cenário operacional o mais próximo possível do real, no caso de AOCS, por exemplo, usando um simulador.

A divisão típica de revisões de um projeto espacial e suas entregas demonstra que a validação quanto à especificação técnica (TS) deve vir antes da qualificação do software (RB), porém “algumas vezes é difícil diferenciar testes realizados para validar o produto contra a TS dos realizados para validar o produto contra o RB.” (48). Portanto, se existir um adequado mapeamento de requisitos entre RB e TS (graças à rastreabilidade), é possível combinar as atividades de validação contra a TS e a qualificação em uma atividade única.

- **Teste de aceitação:** Testes de aceitação são testes de entrega do produto de software. Os testes podem ocorrer ainda em local de testes ou já implantado no local de operação. Os testes de aceitação constam de um conjunto de testes de validação dos requisitos base de software, onde o conjunto de requisitos a serem testados deve ser acordado entre as partes interessadas no desenvolvimento e o recebimento do produto de software. Os testes de aceitação devem considerar as configurações de código e as características da implantação do software e podem também ser utilizados para treinamento, validação de manuais e procedimentos.

Os quatro tipos de testes descritos acima formam o conjunto de testes padronizados e exigidos para desenvolvimento de software espacial tanto para validar requisitos funcionais e não funcionais do software como de dependabilidade. Todos os testes a serem executados no desenvolvimento de software devem ser planejados dentro de um destes grupos.

Diferentes abordagens e técnicas de geração de testes podem ser utilizadas dependendo do objetivo do teste. As abordagens mais comuns são: testes caixa-preta e testes caixa-branca. Segundo (48), teste caixa preta testa o software contra sua especificação para descobrir partes da especificação que

não foram satisfeitas; teste caixa branca testa a implementação para descobrir defeitos lógicos no código. Ambas as abordagens devem ser usadas para se testar um software.

Em (48) são destacados alguns testes que utilizam abordagens e técnicas apropriadas para verificar especificamente *características de dependabilidade* do produto de software, eficientes na descoberta de defeitos, sendo estes:

- **Teste de interface:** estes testes objetivam: (i) localizar erros de entradas e saídas que impeçam o sistema [de software] de operar como um todo; (ii) e localizar erros de tempo nas respostas das interfaces (48). O teste de interface exercita as interfaces do software com combinações de valores extremos e valores normais.

- **Teste de robustez:** visam demonstrar a habilidade do software de lidar com condições anormais de ambiente ou comportamento (entradas anormais ou condições de falha) (48). Testes de robustez exercitam o software com valores inválidos, valores fora dos limites, testam os mecanismos de proteção de erros, testam resposta a condições de erro de operação e inicialização.

- **Teste de desempenho:** estes testes focam os requisitos que incluem *throughput*, carga e tempos de resposta, podendo ser combinados com restrições no uso total de recursos do sistema (ex. uso de CPU, uso de memória, condições de tempo) (48). Os testes de desempenho exercitam o software determinando o uso de recursos por tarefa, contribuição de demanda de recurso em condições médias e piores casos, e piores casos de tempo de resposta e throughput.

É muito importante denotar aqui a diferença entre desempenho de controle e desempenho de software, enquanto o primeiro trabalha qualitativamente a capacidade de controle (ver seção 2.4), o segundo trabalha a capacidade do software de atender as suas demandas computacionais. Também é importante perceber que o não atendimento do desempenho de software em um software

de controle influencia o desempenho de controle, portanto, as atividades das diferentes disciplinas devem ser aliadas para o correto desenvolvimento, verificação e validação do sistema.

## 2.6. Verificação e Validação de FDIR de AOCS

No desenvolvimento do sistema de controle os algoritmos de FDIR, por exemplo, observadores e/ou estimadores são verificados e validados inicialmente pela atividade de Engenharia de Controle. Assegura-se assim que a lógica funcional e os cálculos para a solução de controle desenvolvidos tenham o desempenho de controle e as características de dependabilidade de controle esperados para a missão.

Atualmente, a lógica de FDIR dos controladores é largamente realizada em software, portanto as atividades de verificação e validação relacionadas a software são também essenciais no processo de verificação e validação de um sistema AOCS e dos mecanismos de FDIR.

Segundo (14) a validação de funcionalidades FDIR na área espacial assume teste de todas as possíveis combinações de caminhos das situações de falha, o que se torna custoso com o aumento da complexidade de hardware e software. Com isso, o processo de validação por diversas vezes limita a capacidade de implementar estratégias "*fail operational*" (manter a operação mesmo na ocorrência de falhas). Normatizar e priorizar testes eficazes das funções críticas da operação do AOCS é altamente desejável.

O avanço científico das abordagens de MBT vem ao encontro desses desafios na vertente de geração automática de casos de testes a partir de abstrações do comportamento esperado do sistema. Modelos de estado são especificados ressaltando as propriedades criticas do sistema alvo de teste. Ferramentas de testes utilizam tais modelos para a geração automática de casos de testes capazes de validar as características representadas nos modelos. Desta forma, a seleção de casos de testes de maior relevância (críticos) para o sistema se

torna possível sem o esforço de testar exaustivamente todo o sistema, diminuindo o custo e esforço do processo de V&V.

Portanto, esta alternativa será avaliada com o uso de duas metodologias de teste baseado em modelos para teste de software dos mecanismos FDIR de um AOCS.

# 3 METODOLOGIAS UTILIZADAS NO TRABALHO

Visando especificar testes para apoiar as atividades de verificação e validação de FDIR para AOCS, neste capítulo é apresentada uma revisão sobre teste baseado em modelos. Em seguida, é feita uma síntese das duas metodologias de teste baseado em modelos adotadas nesta dissertação: a metodologia CoFI, da sigla Conformidade e Injeção de Falhas, e a metodologia InRob, de Interoperabilidade e Robustez de Sistemas Intensivos em Software. E por fim, é apresentada a metodologia *System-Theoretic Process Analysis* – STPA, que foi utilizada para elicitar os requisitos, relacionados com FDIR de AOCS, a serem verificados e validados no experimento deste trabalho.

As metodologias CoFI e InRob são ambas metodologias de teste baseado em modelos criadas independentemente uma da outra e usadas para apoiar atividades de verificação e validação de software espacial. Cada metodologia já foi aplicada individualmente em estudos de casos de software embarcado em aplicações espaciais. A metodologia CoFI vem sendo aplicada em diferentes contextos de software crítico como registrado nas referências (18,19,20,49,50,51,52). A metodologia InRob, mais recentemente proposta, apresenta resultados de sua aplicação em software crítico encontrados em (16,17).

## 3.1. Teste Baseado em Modelos

Teste baseado em modelos, do inglês, Model-based Testing – MBT, permite a geração de casos de teste a partir de um *modelo abstrato* formal do sistema sob teste, do inglês, *System under Test* – SUT (21). O *modelo abstrato* ajuda a caracterizar aspectos importantes do sistema que devem ser testados, reduzindo a complexidade para especificar os testes do sistema. Os modelos podem descrever uma parte ou o comportamento completo do sistema sob teste.

De forma geral, a aplicação de teste baseado em modelos consiste das seguintes etapas (17,21):

- **Modelagem:** O comportamento do sistema deve ser modelado usando um padrão de modelagem, por exemplo: FSM – Finite State Machine, TIOA - Timed Input Output Automata, processos de Markov, etc., que represente as características a serem testadas.

- **Seleção do critério de geração de testes:** Escolha do algoritmo que determina o processo de derivar casos de teste a partir dos modelos, por exemplo: percursos nos estados e transições do modelo, execução dos modelos, objetivos de teste (*test purpose*), etc.

- **Geração dos casos de teste:** Processo automático de derivação de conjuntos de casos de testes por uma ferramenta de geração de casos de testes, utilizando como entrada os modelos e aplicando um critério de geração de testes. Cada caso de teste define uma sequência de entradas e saídas esperadas do sistema na execução do mesmo. Os casos de teste gerados normalmente representam um nível de abstração que impossibilita sua execução direta, precisando ser traduzidos em casos de teste executáveis.

- **Execução dos casos de teste:** Os casos de teste são executados por um Sistema de Teste e aplicados no Sistema sob teste (SUT). Todas as saídas observadas no SUT resultantes da execução do caso de teste são registradas.

- **Emissão de veredicto:** As saídas observadas no SUT registradas durante a execução dos casos de teste são comparadas com as saídas esperadas (aquelas que foram registradas no modelo), e uma conclusão é obtida. O resultado da comparação entre a saída esperada e a saída observada pode ser: (i) o comportamento registrado está conforme o esperado; (ii) o comportamento registrado apresenta um desvio com relação ao esperado e, portanto foi detectada uma inconformidade; (iii) ou alguma situação específica

aconteceu durante a execução, incapacitando um veredicto, tornando o teste inconclusivo.

Segundo (21), o principal benefício de teste baseado em modelos é a capacidade de geração automática de um grande número de testes cobrindo diversas características (complexidade) em um tempo relativamente curto. Mesmo que modelar tome um tempo considerável, este sempre será menor que gerar os mesmos casos de teste de forma manual.

A complexidade do sistema é tratada em MBT com o uso de abstrações, fazendo com que tanto a comunidade científica quanto o setor industrial recorram a metodologias de MBT (16).

Além disso, MBT traz o benefício de disciplinar o processo de teste, tanto na dimensão de qualidade dos testes gerados quanto de esforços para a geração. Também contribui na identificação de lacunas e ambiguidades nos requisitos e especificações como resultado do processo de modelagem do comportamento do sistema, melhorando assim questões relativas ao processo de desenvolvimento.

### 3.2. Metodologia CoFI

Como apresentado em (18), a CoFI é uma metodologia de teste de software que visa sistematizar a criação de casos de teste de sistema espaciais. A criação dos testes parte de modelos de estados que representam o comportamento do sistema descrito em especificações textuais. Quando os modelos puderem ser reutilizados de uma missão para outra, poder-se-á reduzir custos com os testes de software em missões espaciais. A metodologia é chamada CoFI pois combina teste de conformidade e validação por falhas injetáveis. Uma das principais características da metodologia é mapear falhas provocadas por problemas físicos causados pela radiação que afetam o processamento e a comunicação a bordo de satélites. A metodologia utiliza métodos formais e diagramas da UML na representação do comportamento do sistema visando facilitar a automação de seus passos. Como proposto em (18)

e (49) a CoFI compreende além de uma metodologia que guia o testador a criar modelos para serem usados na geração automática de casos de teste, um processo de teste. O processo CoFI, com relação à norma ECSS-E-40 Part B (53), está contido no escopo da atividade validação com relação à especificação técnica, que por sua vez, faz parte do Processo de Validação.

Em resumo, a metodologia se baseia na descrição formal da especificação de um SUT por meio de máquinas de estados finitos que representam o comportamento SUT frente à chegada de eventos (ou entradas). A CoFI recomenda a identificação de Serviços do SUT, os quais podem ser considerados como funções do sistema do ponto de vista de um usuário. Cada Serviço é descrito por um conjunto de máquinas de estados finitos, que, dependendo do tipo de comportamento que representam, são caracterizadas em quatro classes distintas. As quatro *classes de comportamento* construídas com modelos em máquinas de estados finitos da CoFI são: (i) comportamento nominal; (ii) comportamento de exceções especificadas; (iii) comportamento frente a eventos inesperados, aqui chamados de Caminhos Furtivos; (iv) comportamento frente a falhas de hardware que disparam mecanismos de Tolerância a Falhas. Sendo as três últimas classes, aquelas que representam o comportamento do sistema frente a situações de falha, as quais podem ter sido previstas na especificação textual fornecida ao testador ou não.

A aplicação da metodologia para geração dos casos de testes pode ser dividida em três fases, descritas a seguir:

- **Identificação:** Baseado na especificação do SUT devem ser identificados os Serviços, as falhas de hardware que podem acontecer, as facilidades e as restrições Sistema de Teste (compreendendo todas as ferramentas usadas para executar os testes), os Pontos de Controle e Observação (PCO) e os possíveis eventos (entradas) e ações (saídas) do SUT.
- **Modelagem:** Baseado na especificação do SUT, o comportamento dos Serviços deve ser representado em modelos de estado. Para cada serviço inicialmente deve ser feito, pelo menos um modelo do comportamento normal com os eventos esperados e refletindo as

reações do sistema frente a estes eventos. Aqui os modelos são chamados modelos normais. Posteriormente deve ser gerado um novo modelo contendo as exceções que se encontram especificadas no conjunto de requisitos do sistema. Tal conjunto de modelos é chamado de Exceções Especificadas. Em seguida, deve ser modelado o comportamento frente a eventos normais que ocorrem em momentos não esperados (Caminhos Furtivos). E por fim, deve ser gerado um ou mais modelos representando o comportamento do sistema quando as falhas de hardware acontecem. Estes modelos são chamados de tolerância a falhas, pois representam o comportamento dos mecanismos de tolerância a falhas de hardware que o sistema deve prover para garantir a dependabilidade.

- **Geração:** Com o auxílio de uma ferramenta de apoio a geração automática de testes para máquinas de estados finitos geram-se os conjuntos de casos de testes. Os casos de teste contêm caminhos da máquina de estado, ou seja, sequências de eventos (entradas) e respectivas ações (ou saídas esperadas) que caracterizam as transições das máquinas de estado.

A metodologia não se prende ao uso de uma ferramenta particular para a geração dos casos de testes. Dois exemplos de ferramentas já usadas com a CoFI para geração automática de casos de teste são a Condado (54) e a plataforma JPlavisFSM (19). A divisão da especificação do sistema em serviços (ou funções) e a representação do mesmo em classes de comportamento pelas máquinas de estados mitiga o problema da explosão do espaço de estados na geração dos testes. Após a geração dos casos de testes ocorre a execução e análise dos resultados de testes que pode ser considerado uma nova fase. Os casos de testes representam uma sequência de eventos e ações esperadas que o sistema sob teste deve responder e que demonstram a conformidade do serviço frente a sua especificação.

## 3.3. Metodologia InRob

Como apresentado em (16), a InRob é uma metodologia para testes de integração de sistemas de tempo real intensivos em software com foco em interoperabilidade e robustez.

O objetivo do teste de interoperabilidade é verificar que dois sistemas de software comunicantes, juntos tidos como o sistema sob teste, proveem corretamente os serviços descritos por sua especificação. A questão de robustez visa demonstrar que não só a operação nominal do serviço realizado conjuntamente é correta, mas que desvios de tempo no canal de comunicação interno ao sistema sob teste são tratados conforme especificados.

A metodologia combina técnicas de teste baseado em modelos com perfis de serviços para guiar a construção de modelos abstratos de interoperabilidade com base no serviço realizado conjuntamente por dois sistemas que se comunicam. O objetivo é testar aspectos de tempo real na integração destes sistemas. Para tanto, os modelos de serviços são expandidos com desvios de tempo para apoiar a geração de casos de testes de robustez.

A metodologia é composta de cinco componentes-chave: perfis dos serviços; modelo nominal do serviço de interesse; perigos quanto a tempo; modelo estendido do serviço; objetivos de teste. Estes elementos são encadeados em um processo para teste de integração que, assim como a metodologia CoFI, leva em consideração ferramentas para geração automática de testes e o ambiente de execução de testes (sistema de teste).

A metodologia combina três abordagens: (i) uma extensão de perfil operacional do serviço; (ii) uso do formalismo Timed Input and Output Automata (TIOA) para modelar o comportamento de um serviço colaborativo provido pelo sistema sendo integrado; (iii) geração automática de casos de teste usando objetivos de teste.

O cálculo do perfil operacional quantifica os serviços em termos de uso e criticidade em operação o que permite priorizar o esforço de modelagem dos perfis mais críticos em detrimento dos serviços menos críticos. O uso do formalismo TIOA permite representar o comportamento das interações de interesse nos modelos comunicantes por meio de transições específicas de entradas e saídas associadas a uma base de tempo, priorizando assim a modelagem das propriedades de tempo real associadas à interoperabilidade.

Finalmente, o formalismo de objetivos de testes permite guiar a geração de casos de testes de forma a cobrir apenas as propriedades do modelo especificadas no objetivo de teste (estratégia *test purpose*). Assim, a priorização dos casos de testes gerados ocorre em tempo de execução dos modelos.

Com o uso das abordagens de priorização mencionadas, a InRob evita o problema da explosão de caminhos na geração automática de casos de testes a partir de modelos de estado. O conceito de estágio de integração é adotado pela metodologia InRob para destacar as interfaces dos componentes em integração nos modelos de sistemas comunicantes, contornando assim a questão de complexidade na modelagem completa do comportamento de dois software comunicantes. A InRob orienta a geração de casos de teste viáveis e eficazes a cada estágio de integração do sistema de software, produzindo conjuntos de cenários de testes que venham a exercitar o sistema de forma a ter um bom esforço-benefício de teste.

Os artefatos a serem encadeados em um processo de teste proposto na metodologia são:

- **Perfil de serviço:** o perfil de serviços mapeia quais componentes de software participam de forma conjunta na execução de um dado serviço. Com isso pode-se calcular a dependência dos serviços quanto aos componentes e interfaces de software utilizadas e o resultado apoia a priorização de testes. Os serviços com maior dependência são priorizados.
- **Modelo nominal do serviço:** modelo representando as interações entre dois componentes de software que se comunicam para realizar um dado serviço. O modelo de estado é representado no formalismo TIOA e serve como especificação nominal das interações. Guardas temporais são utilizadas para representar intervalos de tempo entre transições de estado.
- **Perigos quanto a tempo:** desvios de tempo que podem ocorrer na interação entre componentes. Os desvios são agrupados em classes de equivalência de causa-efeito para minimizar a geração de casos de teste equivalentes.

- **Modelo estendido do serviço:** modelo contendo a representação do serviço nominal com extensões das interações que representam desvios de tempo.
- **Objetivo de teste:** conjunto de propriedades verificáveis no modelo. As propriedades são expressões formais que visto a linguagem formal do modelo podem ser verificadas em tempo de execução do modelo como presentes ou não para fins de geração dos casos de testes.

O processo de teste guiado pela metodologia InRob é dividido em três fases: A, B e C. Na **fase A** ocorre o modelamento do sistema sob teste, portanto a geração do perfil de serviços, geração dos modelos nominais dos serviços de interesse, a análise do modelo nominal em termos dos perigos relacionado a tempo, e a geração dos modelos estendidos do serviço é realizada.

Na **fase B** são definidos os objetivos de teste baseados nas propriedades de verificação que se desejam testar. A partir dos objetivos de testes a geração dos casos de teste é feita usando uma ferramenta de *model checking* para TIOA. A metodologia não se prende a uma ferramenta específica, qualquer ferramenta que possa gerar casos de teste baseado em objetivo de teste em modelos TIOA pode ser utilizada. Como exemplo a ferramenta HJ2IF (55) foi utilizada em (16).

Na **fase C** ocorre a execução dos testes no sistema sob teste e análise dos resultados. Para tanto recomenda-se escolher na fase B ferramentas de geração automática cuja saída são casos de testes descritos na forma de scripts que possam ser importados, com a devida conversão da estrutura de dados, no ambiente de execução dos testes.

A metodologia se apoia na capacidade do sistema de testes conseguir emular os desvios temporais para a execução dos testes. Se o ambiente de testes não tiver originalmente esta capacidade, dispositivos de apoio podem ser adicionados para emular este comportamento, como descrito em (16) e (17).

## 3.4. Metodologia STPA

A metodologia *System-Theoretic Process Analysis* (STPA) (22,44) é utilizada para desenvolvimento de sistemas *safety-critical*, e portanto, não é uma metodologia para testes baseado em modelos. Neste trabalho ela foi aplicada para elicitar requisitos do subsistema AOCS usados tanto para o desenvolvimento do protótipo de AOCS, como para apoio a análise dos resultados da aplicação das metodologias de teste CoFI e InRob.

A metodologia STPA utiliza uma técnica de análise de ameaças (*hazards*) com um novo modelo de causalidade de acidente baseado em teoria de sistemas ao invés de teoria de confiabilidade (22,44).

A técnica tem os mesmos objetivos que as outras técnicas de análise de ameaças, que é identificar cenários que levam a ameaças identificadas e, portanto, a perdas, para que possam ser eliminadas ou controladas no sistema. O resultado da aplicação desta técnica é a identificação de um conjunto maior de causas (das ameaças), muitas delas não envolvendo falhas ou falta de confiabilidade que previne apenas os acidentes de falha de componente (22). A STPA considera acidentes advindos de interações entre componentes, que podem ser fruto de erros no projeto ou interações não seguras entre componentes não falhos. Muitas dessas causas adicionais estão relacionadas a novos tipos de tecnologia (como computadores e sistemas digitais) e maiores níveis de complexidade dos sistemas construídos hoje (44).

No modelo de causalidade de acidentes da STPA, *safety* é uma propriedade emergente que se forma quando componentes interagem de forma adequada entre si em um ambiente de operação. Existe um conjunto de requisitos de *safety* relacionados aos componentes que asseguram as propriedades de *safety* do sistema. **Acidentes ocorrem quando interações violam os requisitos de *safety*, ou as restrições adequadas para a interação não foram corretamente impostas.** Portanto a essência da STPA é migrar a ênfase de prevenção de falhas de componentes para a imposição de requisitos de *safety* no sistema e seus componentes.

De forma resumida, a técnica assume que acidentes ocorrem quando a estrutura do sistema não realiza adequadamente o controle dos requisitos de *safety* (desde o nível organizacional até os componentes tecnológicos), e *estados perigosos* ocorrem devido a:

- Perturbações ou condições externas não tratadas
- Falhas de componentes não corretamente tratadas
- Interações perigosas entre componentes
- Ações de controle de múltiplos controladores de forma inadequada

O processo de aplicação da técnica é dividido em quatro etapas:

1. Estabelecimento da base da análise - definição dos acidentes ou perdas; identificação das ameaças associadas ao acidente; especificação dos requisitos de *safety* de alto nível; definição preliminar da arquitetura funcional do sistema;
2. Identificação de **ações de controle perigosas** - identificação de ações de controle perigosas na arquitetura preliminar funcional do sistema. Os tipos de ações de controle possivelmente perigosas são: (i) ausência de ação de controle; (ii) ação de controle indevida; (iii) ação de controle provida muito tarde ou muito cedo; (iv) ação de controle fora da ordem correta; (v) ação de controle aplicada por um tempo mais longo ou mais curto que o correto (tanto em sistemas discretos como contínuos);
3. Derivação de requisitos de *safety* - os requisitos desta etapa representam um entendimento melhor do sistema, dada uma arquitetura preliminar, e das ameaças de *safety*; tais requisitos devem satisfazer os requisitos de *safety* declarados na primeira etapa.

4. Apoio ao projeto do sistema - estende o entendimento das causas que levam a ocorrência de uma ação de controle perigosa. Nesta etapa mais uma causa de perigo é considerada: (vi) ação de controle não executada adequadamente, devido à arquitetura e a componentes da solução. São descritos cenários que podem levar a ocorrência da ação perigosa. O entendimento das causas pode elicitar requisitos adicionais, e as informações desta etapa apoiam o desenvolvimento do sistema, para mitigação ou eliminação de potenciais causas de ameaças. Nesta etapa cenários que levam as ameaças são descritos, baseado largamente em experiência prévia na área e conhecimento do sistema e das tecnologias utilizadas na solução. Existe uma lista de causas comuns que podem levar a ameaças que pode ser usada também para apoiar esta etapa.

# 4 AVALIAÇÃO TEÓRICA DAS METODOLOGIAS COFI E INROB

Neste capítulo, visando avaliar o potencial combinado das metodologias de teste baseado em modelos CoFI e InRob, quando aplicadas na verificação e validação das funcionalidades de FDIR de um subsistema AOCS, apresenta-se uma análise das metodologias frente aos requisitos de verificação e validação de um software espacial estabelecidos nos guias ECSS, em relação a teste para validação de *características de dependabilidade*.

## 4.1. Análise da Aplicabilidade das Metodologias

Como explicado no Capítulo 2, teste é um método de verificação do comportamento esperado do software, o qual também apoia a validação do software tanto com relação a sua especificação técnica (TS) quanto os seus requisitos base (RB).

As metodologias de teste baseados em modelos, CoFI e InRob, objeto de análise desse trabalho, se propõem a orientar a criação de modelos, a partir dos quais, casos de testes podem ser gerados automaticamente, para validar os requisitos de um sistema sob teste. As metodologias orientam que, a partir dos requisitos especificados, serviços providos pelo software sejam identificados e sejam representados no formalismo dos modelos adotados por estas metodologias, com as devidas abstrações.

Cabe destacar que, cada metodologia analisada guia a geração de casos de testes visando validar aspectos específicos. No caso da CoFI visa-se validar a conformidade do comportamento (normal e anormal) do SUT com relação ao comportamento esperado, descrito nos requisitos especificados, bem como, a conformidade do comportamento frente a eventos inoportunos e falhas de hardware. No caso da InRob o foco é validar a interoperabilidade de dois software de tempo real comunicantes sob a ótica de aspectos de robustez especificados, por exemplo time-out que podem ser violados por defeitos no canal de comunicação ou por não cumprimento do desempenho esperado pelos componentes em integração.

Considerando os tipos de testes recomendados pelos guias de desenvolvimento (47) e (48), apresentados no Capítulo 2, a Tabela 4.1 mostra a aplicabilidade de cada metodologia com relação aos tipos de teste relacionados com a fase de desenvolvimento do software.

Tabela 4.1. Metodologias vs. tipos de testes relacionados com as fases de desenvolvimento do software

| | **Teste de Unidade** | **Teste de Integração** | **Teste de Validação** | **Teste de Aceitação** |
|---|---|---|---|---|
| **CoFI** | X | | X | X |
| **InRob** | X | X | X | |

Para os Testes de Integração, de Validação, e de Aceitação, as metodologias se mostram **complementares**, sendo a metodologia InRob direcionada para o teste de integração pois foca a interoperabilidade entre componentes, e também apoiando a validação do software, a metodologia CoFI para os testes de validação e de aceitação, quando os componentes atuam de forma integrada para as funcionalidades de alto nível do software, num contexto de teste do tipo caixa-preta.

O Teste de Unidade é considerado pelo guia (46) como parte da atividade de desenvolvimento de software, esses testes no contexto de projetos de software embarcado espaciais são excluídos do escopo das atividades de V&V por fazerem parte da implementação. Entretanto, caso haja interesse em aplicar as metodologias no contexto do fornecimento das unidades, pode-se dizer que a metodologia CoFI pode ser aplicada em Teste de Unidade se a unidade tiver um comportamento especificável em máquina de estado. A CoFI pode ser aplicada parcialmente. Caso a unidade não apresente característica de software embarcado ou não tenha requisitos de tolerância a falhas, a modelagem do comportamento frente a falhas de hardware pode ser omitida, como em (56). Outras simplificações também podem ser adotadas, como ilustrado na referência (51). A metodologia InRob pode ser aplicada no escopo

de unidades, nas perspectivas de verificação de comunicação de componentes de um módulo de software sendo considerado uma unidade.

Quanto aos tipos de teste relacionados com a validação das *características de dependabilidade* de software destacadas pelo guia (48), ver seção 2.5.1, a Tabela 4.2 apresenta o potencial das metodologias analisadas para estes tipos de teste.

Tabela 4.2. Metodologias vs. testes focando dependabilidade.

| | **Teste de Interface** | **Teste de Robustez** | **Teste de Desempenho** |
|---|---|---|---|
| **CoFI** | X | X | |
| **InRob** | X | X | X |

Considerando os testes cujo foco é verificar características de dependabilidade de software descritos na revisão bibliográfica (seção 2.5.1), as metodologias demonstram **similaridades** no escopo com relação à cobertura dos tipos de testes da Tabela 4.2. Nesse contexto, de acordo com (25), cobertura refere-se a uma medida de representatividade das situações para as quais o sistema é submetido durante sua validação comparada com as situações reais com as quais será confrontado durante sua vida operacional. Observa-se comonalidades nas duas metodologias para testes de interface e robustez. Em termos da cobertura de Testes de Desempenho, a metodologia InRob se aplica com certas restrições, ela pode ser aplicada apenas para validar requisitos que incluem throughput e tempo de resposta de funções especificadas, as quais são verificáveis na fase de integração.

Apesar da similaridade na cobertura de testes focando dependabilidade de software apresentada na Tabela 4.2, especificidades são percebidas quando as metodologias são analisadas em termos do potencial dos modelos CoFI e InRob em cobrir as seis **ações de controle perigosas** (ver seção 3.4) que podem levar um sistema a entrar em um estado perigoso, de acordo com a

metodologia STPA. A Tabela 4.3 apresenta as especificidades identificadas para as ações de controle perigosas.

Tabela 4.3. Metodologias vs. ações de controle perigosas

| | **CoFI** | **InRob** |
|---|---|---|
| **(i) Ausência de ações de controle** | X | X |
| **(ii) Ações de controle indevidas** | X | |
| **(iii) Ação de controle muito tarde ou muito cedo** | | X |
| **(iv) Ações de controle fora de ordem** | X | X |
| **(v) Tempo de aplicação da ação muito longo ou curto** | | X |
| **(vi) Ações de controle não executadas adequadamente** | X | |

Na CoFI, (i) ausência de ações de comandos e (ii) ações de controle indevidas podem ser indicadas nas classes de modelos de Exceção Especificada (caso tenha sido previsto o comportamento do SUT nestas situações) e na classe Caminhos Furtivos (caso a ausência de ações de controle e as ações indevidas sejam consequência de eventos que acontecem em momentos inesperados) ou ainda na classe Tolerância a Falhas (caso sejam consequência de uma falha de hardware). As (iv) ações de controle fora de ordem são sempre modeladas nos modelos de Caminhos Furtivos. A metodologia CoFI recomenda modelos que representem o comportamento do SUT frente a falhas de hardware, uma das principais causas para execução de (vi) ações de controle não executadas de forma inadequada.

Na InRob o caso de ausência de comandos (i) é modelado com a perda de mensagens, e as ações de controle muito tarde ou muito cedo (iii) são modelados com a temporização de mensagens, por meio de atrasos e antecipações de mensagens, que podem cobrir também a situação fora de ordem (iv). Estas situações são emuladas no sistema de teste por um emulador de defeitos do canal de comunicação. A InRob é capaz de gerar testes que tratam bem as questões de temporização de mensagens de sistemas

comunicantes, podendo contemplar aplicações cujo tempo de execução é longo ou curto (v).

## 4.2. Conclusão sobre o Potencial das Metodologias de Teste

A análise comparativa nos permite afirmar que as metodologias se mostram teoricamente adequadas para seu uso combinado apresentando forte complementariedade, e cobrindo grande parte do escopo das atividades de verificação e validação de software e de características de dependabilidade de software principalmente nos testes de interface e robustez. Quanto à verificação de desempenho de software, este fica parcialmente coberto pela metodologia InRob.

# 5 APLICAÇÃO DAS METODOLOGIAS EM UM PROTÓTIPO DE AOCS

Este capítulo apresenta a aplicação das metodologias InRob e CoFI em um experimento prático. Um subconjunto de funcionalidades de um subsistema AOCS de satélite foi especificado e prototipado com o propósito de evidenciar a importância dos mecanismos de detecção, isolamento e recuperação de falhas (FDIR) no funcionamento seguro (*safety*) do AOCS e a necessidade de um processo sistematizado para testar tais mecanismos.

Inicialmente foi desenvolvida uma especificação de requisitos para um sistema de controle para satélites (AOCS), e um protótipo deste sistema foi implementado. Em seguida, modelos comportamentais dos serviços providos pelo AOCS prototipado foram construídos, guiados pelas metodologias InRob e CoFI.

Dada a enorme dimensão das atividades de V&V de sistemas espaciais, o experimento se restringiu à verificação de propriedades de dependabilidade do software dos mecanismos de FDIR de um AOCS, nas fases de integração, validação, aceitação. Focou-se aspectos de robustez do software, características de tempo real e interoperabilidade entre os componentes. Desempenho de software não foi abordado no experimento, por limitação de tempo e restrição de sistema de teste com capacidade de testes de carga disponível.

O experimento se divide nas seguintes atividades:

a) elicitação de requisitos de FDIR, usando a metodologia SPTA,

b) implementação do protótipo de AOCS,

c) aplicação das metodologias CoFI e InRob ao protótipo de AOCS,

d) avaliação dos resultados do experimento prático.

A Figura 5.1 apresenta uma visão mais detalhada das atividades realizadas na aplicação das metodologias.

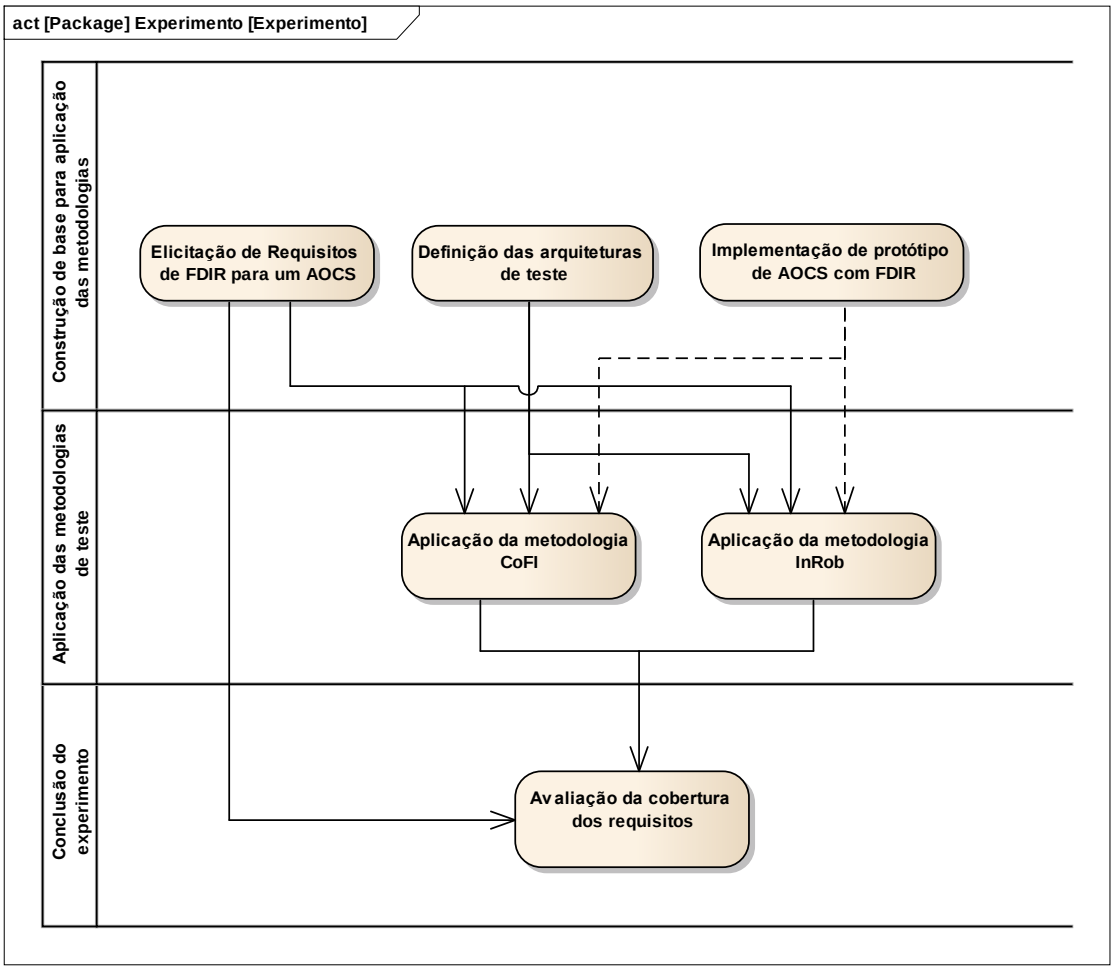


Figura 5.1. Aplicação das metodologias de teste

## 5.1. Elicitação de Requisitos de FDIR

A técnica STPA foi utilizada para a elicitação de requisitos das funcionalidades de FDIR de um sistema de controle para satélites. Como a STPA orienta a elicitação de requisitos de *safety*, ela pode ser utilizada para a elicitação dos requisitos de FDIR.

Considerando que o objetivo aqui é a obtenção de requisitos, apenas as três primeiras etapas da técnica foram aplicadas, quais sejam:

1. Estabelecimento da base da análise
2. Identificação de ações de controle perigosas

3. Derivação de requisitos de safety

### 5.1.1. Estabelecimento da Base da Análise

Na etapa 1, as seguintes perguntas precisam ser respondidas:

- Quais acidentes ou perdas queremos evitar?
- Quais ameaças podem levar ao acidente ou perda?

Considerando um sistema AOCS para satélites os acidentes e perdas levantadas são apresentadas na Tabela 5.1.

Tabela 5.1. Acidentes e perdas AOCS

| **Acidentes ou perdas a serem evitadas** |
|---|
| 1- Perda da capacidade de controle de apontamento da espaçonave necessária para continuidade da missão |
| 2- Perda da capacidade de correção de órbita |
| 3- Acidente pela ocorrência de estado de esgotamento da energia da espaçonave |

Para cada acidente ou perda, listamos as ameaças que podem levar a ela. A definição de ameaça corresponde às condições ou estados não desejados, que em uma condição de pior-caso, levam aos acidentes ou perdas. As perdas e respectivas ameaças para o sistema AOCS são apresentadas nas Tabelas 5.2, 5.3 e 5.4.

Tabela 5.2. Perda 1 e respectivas ameaças

| **Perda da capacidade de controle de apontamento** |
|---|
| 1.a- Estado de deficiência da espaçonave de estimar seu estado corrente de apontamento |
| 1.b- Estado de deficiência da espaçonave de atuar sobre seu apontamento |
| 1.c- Estado de controle de atuação sobre apontamento incorreto ou fora do desempenho de controle |
| 1.d- Condição de perturbações e ruídos da operação do satélite no ambiente espacial |

Tabela 5.3. Perda 2 e respectivas ameaças

| **Perda da capacidade de correção de órbita** |
|---|
| 2.a- Estado de incapacidade da espaçonave de controlar seu apontamento |
| 2.b- Estado de deficiência da espaçonave de atuar sobre elementos de sua órbita |

Tabela 5.4. Perda 3 e respectivas ameaças

| **Acidente de esgotamento da energia da espaçonave** |
|---|
| 3.a- Estado de incapacidade da espaçonave de controlar seu apontamento |
| 3.b- Estado de deficiência da espaçonave de atuar sobre os painéis solares |

Podemos notar que as ameaças 2.a e 3.a das Tabela 5.3 e Tabela 5.4 coincidem com a Perda 1: “Perda da capacidade de controle de apontamento” sendo portanto contempladas nesse escopo.

Importante ressaltar que a análise está considerando apenas *safety* para o subsistema AOCS, portanto casos de danificação das células dos painéis

solares ou problemas elétricos no satélite, responsabilidade de outros subsistemas, não foram considerados.

Foi considerado também na especificação das ameaças que os comandos de mudança de órbita são enviados de um sistema em solo e, portanto há uma malha fechada com solo, nunca corrigidos autonomamente. E que os painéis solares precisam ser apontados ativamente durante a operação.

Uma vez identificadas as ameaças, os requisitos de safety de alto nível do sistema são elicitados e correspondem a declarações de que as ameaças devem ser evitadas ou proibidas no sistema.

Como apresentado em (44), a Tabela 5.5 relaciona as ameaças e os respectivos requisitos de safety de alto nível.

Tabela 5.5. Requisitos de alto nível

| **Ameaça** | **Requisito de Safety** |
|---|---|
| 1.a | A espaçonave deve ser capaz de estimar seu estado corrente de apontamento mesmo na existência de falha simples de sensores, com desempenho para realização da missão. |
| 1.b | A espaçonave deve manter-se em um estado seguro de apontamento em caso de falha de atuação. |
| 1.c | A espaçonave deve ser capaz de monitorar seu desempenho de controle e manter-se em um estado seguro de apontamento em caso de falha no controle. |
| 1.d | A espaçonave deve ser capaz de manter-se controlada sob condições de ruído e perturbação condizentes com a especificada para missão. (deve haver aqui uma ligação a especificação destes parâmetros em um caso real) |
| 2.b | A espaçonave deve cessar a manobra e manter-se em um estado seguro de apontamento em caso de falha na execução de manobra. |
| 3.b | A espaçonave deve ser manter-se em um estado seguro de apontamento e consumo de energia em caso de falha na atuação do painel solar. |

Estes requisitos serão utilizados mais a diante para derivar os requisitos dos componentes do sistema.

Para derivar os requisitos de *safety* de componentes, foi utilizada a estrutura funcional preliminar do sistema que consta dos seguintes módulos de serviços de uma plataforma orbital:

- TT&C – Telemetry, Tracking & Command: Telecomunicação de Serviço
- OBDH – Onboard Data Handler: Gestão de Dados de Bordo
- PDU – Power Distribution Unit: Unidade de Distribuição de Energia
- AOCS – Attitude and Orbit Control System: Subsistema de Controle de Atitude e Órbita

A Figura 5.2 ilustra a estrutura funcional preliminar do sistema, em termos de fluxo de controle operacional envolvendo outros subsistemas do segmento espacial, na ótica da operação do AOCS.

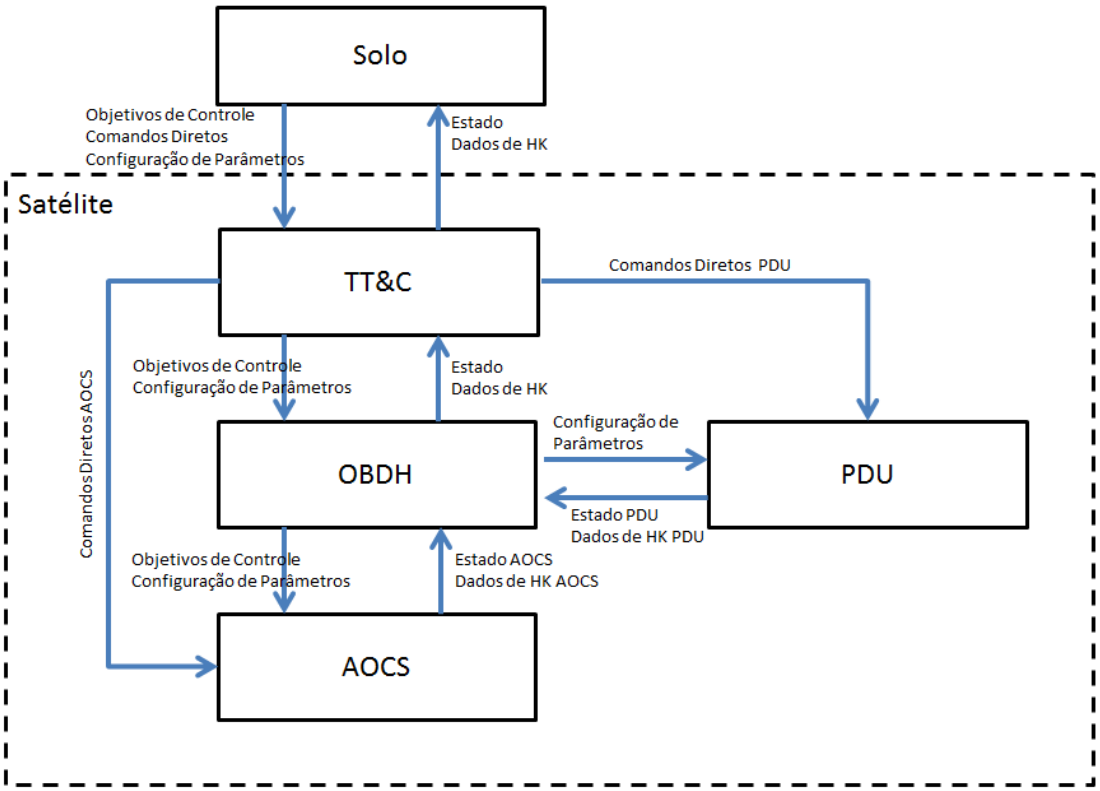


Figura 5.2. Arquitetura Funcional, visão geral

A Figura 5.3 ilustra a arquitetura funcional de um subsistema AOCS, conforme conceitos apresentados no Capítulo 2.

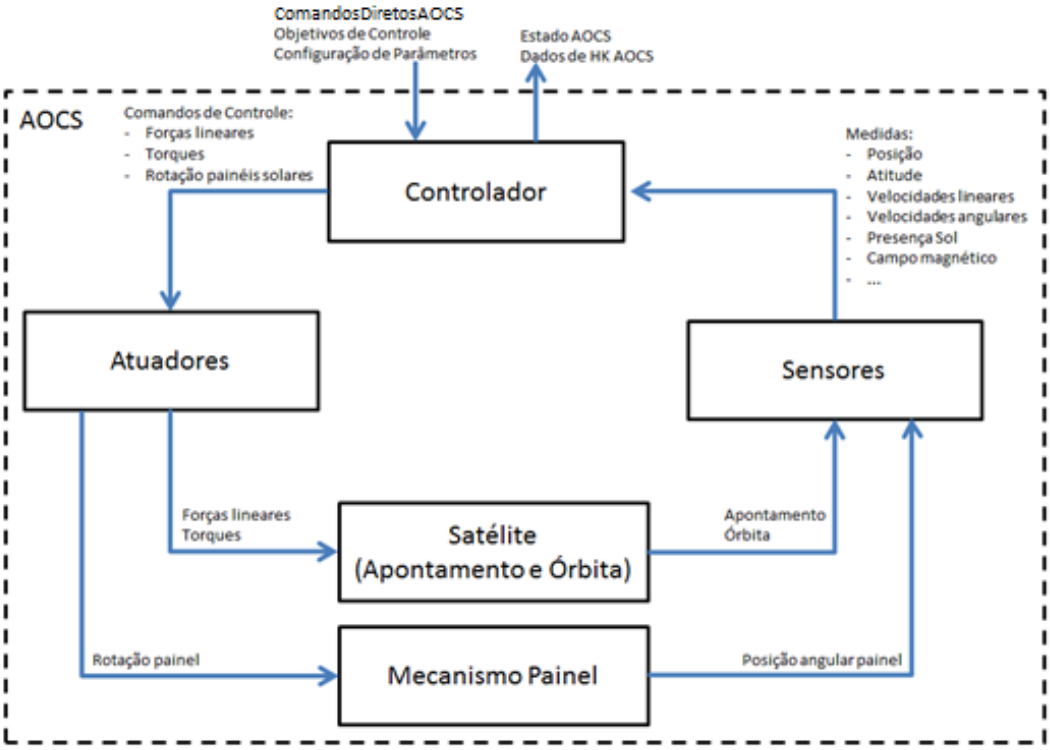


Figura 5.3. Arquitetura Funcional, visão AOCS

As responsabilidades e o modelo de processo do controlador são descritos na Figura 5.4.

**Controlador**

Responsabilidade:
- Definir o modo de controle
- Estimar a atitude
- Determinar os comandos de controle
- Detectar falhas , isola-las e recuperar o controle

Modelo do Processo do Controlador:
- Modo: Emergência, Nominal, Manobra orbital, ...
- Erro apontamento: Apontado, Não apontado
- Estado equipamentos: Falha, Funcional
- Apontamento Painéis Solares: Apontados, Não apontados

Figura 5.4. Descrição do Controlador

Na descrição das responsabilidades e no modelo de processo do controlador foram utilizados alguns recursos previstos em (44), como Simplificação Lógica e Discretização de Variáveis Contínuas.

O processo da operação de um controlador de sistema inercial é altamente complexo, contendo diversos parâmetros que influenciam no resultado de um ciclo e vários estados dentro de um modo, como por exemplo, os estados acompanhando um alvo, realizando trajetória, etc., não apresentados na Figura. Porém o Modelo do Processo deve explicitar as variáveis do processo que podem diferenciar os efeitos das ações de controle entre segura e não segura, e não as variáveis utilizadas internamente na operação. No experimento estas variáveis foram consideradas relevantes para derivação dos requisitos de *safety* dos componentes.

Uma vez concluída a especificação dos artefatos base da metodologia, foi realizada a análise das ações de controle.

### 5.1.2. Identificação de Ações de Controle Perigosas

Para cada ameaça descrita nas Tabelas 5.2, 5.3 e 5.4 é apresentada a análise das ações de controle. As Tabelas 5.6 a 5.11 indicam se as ações de controle, considerando as causas de perigo, podem ou não tornar a ameaça realidade.

Tabela 5.6. Ameaça 1.a: Deficiência para estimar estado de apontamento

| Ação de Controle | Não prover | Prover | Ordem ou cedo/tarde | Tempo de aplicação errado |
|---|---|---|---|---|
| **Forças lineares** | X | SIM, mudança indevida de órbita pode tirar capacidade do controlador de se encontrar no espaço | SIM, se Erro Apontamento for "Não apontado" na correção de órbita irá mudar para órbita indevida<br><br>SIM, se tempos forem errados, o apontamento pode alterar e mudar para órbita indevida | SIM, muita aplicação pode levar a uma órbita indevida e impedir controlador de se encontrar no espaço |
| **Torques** | SIM, se sistema permitir caminhada para fora de faixa de operação dos sensores | SIM, se sistema rumar para fora de faixa de operação dos sensores | SIM, se atuações e sensoriamentos começarem a se interferir incorretamente levando a estado incorreto de sensoriamento (ex. magnetômetro com magnetotorquer) | SIM, se sistema rumar para fora de faixa de operação dos sensores |
| **Rotação painéis solares** | X | X | X | X |

Esta tabela mostra que principalmente ações de controle não coordenadas podem prejudicar a capacidade do satélite de estimar seu estado de apontamento.

Tabela 5.7. Ameaça 1.b: Deficiência sobre atuação no apontamento

| Ação de Controle | Não prover | Prover | Ordem ou cedo/tarde | Tempo de aplicação errado |
|---|---|---|---|---|
| **Forças lineares** | X | SIM, equipamento em falha pode gerar torques indevidos e causar instabilidade | X | SIM, muita aplicação pode ser levada a instabilidade |
| **Torques** | SIM, se permitir caminhada para instabilidade | SIM, equipamento em falha pode levar a instabilidade | Sim, ordem de atuação incorreta ou atuações em tempos errados podem levar a instabilidade | Sim, pode levar a instabilidade por excesso de atuação |
| **Rotação painéis solares** | X | X | X | X |

Esta tabela mostra que principalmente falhas em equipamentos e problemas comandando os equipamentos (comunicação) são perigosos podendo levar o sistema a um estado de instabilidade.

Tabela 5.8. Ameaça 1.c: Controle incorreto ou perda de desempenho

| Ação de Controle | Não prover | Prover | Ordem ou cedo/tarde | Tempo de aplicação errado |
|---|---|---|---|---|
| **Forças lineares** | X | SIM, se fora do modo adequado irá conflitar com controle de atitude<br><br>SIM, se equipamento em falha pode gerar torques indevidos | X | X |
| **Torques** | SIM, degrada desempenho | SIM, controle indevido se equipamento em falha | SIM, degrada desempenho | SIM, degrada desempenho |
| **Rotação painéis solares** | SIM, degradação de energia pode levar a necessidade de desligamento de equipamentos | SIM, degradação de energia pode levar a necessidade de desligamento de equipamentos | SIM, degradação de energia pode levar a necessidade de desligamento de equipamentos | SIM, degradação de energia pode levar a necessidade de desligamento de equipamentos |

Esta tabela mostra principalmente que não realizar as ações de controle dentro de seu ciclo de controle degrada o desempenho, também que equipamentos em falha e apontamento incorreto do painel podem levar a modos degradados de controle.

Tabela 5.9. Ameaça 1.d: Condição de perturbações e ruídos da operação do satélite no ambiente espacial

| Ação de Controle | Não prover | Prover | Ordem ou cedo/tarde | Tempo de aplicação errado |
|---|---|---|---|---|
| **Forças lineares** | X | SIM, se levar a órbita indevida com perturbações inadequadas a missão | SIM, se levar a órbita indevida com perturbações inadequadas a missão | SIM, se levar a órbita indevida com perturbações inadequadas a missão |
| **Torques** | SIM, não tratar as perturbações pode acumular seus efeitos e aumentar seu potencial destrutivo a missão | X | X | X |
| **Rotação painéis solares** | X | X | X | X |

Esta tabela mostra principalmente que é preciso tratar perturbações e ruídos durante toda operação, e que ações de correção indevidas podem levar a uma condição ambiental desfavorável a missão.

Tabela 5.10. Ameaça 2.b: Deficiência da espaçonave de atuar sobre elementos de sua órbita

| Ação de Controle | Não prover | Prover | Ordem ou cedo/tarde | Tempo de aplicação errado |
|---|---|---|---|---|
| **Forças lineares** | SIM, se órbita não corrigida, capacidade de operação do satélite pode ser perdida por órbita indevida | SIM, se equipamento em falha, fora de modo coreto, ou ainda não apontado pode levar a órbita indevida | SIM, pode levar a órbita indevida e perda de capacidade de operação | SIM, pode levar a órbita indevida e perda de capacidade de operação e esgotamento de combustível |
| **Torques** | X | SIM, se gerar torque indevido durante a execução de manobra orbital pode levar a órbita indevida | X | X |
| **Rotação painéis solares** | X | X | X | X |

Esta tabela mostra principalmente que ações de correção de órbita não coordenadas podem levar o satélite a uma órbita indevida, podendo causar perda de comunicação ou condição inadequada de operação.

Tabela 5.11. Ameaça 3.b: Deficiência da espaçonave de atuar sobre os painéis solares

| Ação de Controle | Não prover | Prover | Ordem ou cedo/tarde | Tempo de aplicação errado |
|---|---|---|---|---|
| **Forças lineares** | X | X | X | X |
| **Torques** | X | X | X | X |
| **Rotação painéis solares** | X | X | X | X |

Esta tabela mostra que a ameaça de deficiência para atuar sobre os painéis vem de fatores externos e falhas nos componentes e não de ações de controle. Nenhuma ação de controle leva a uma incapacidade de atuação nos painéis, dado que não se mostra estranho já que em tabelas anteriores foi mostrado que apontamentos errados podem incapacitar o painel de obter energia, que é a perda sendo evitada. Portanto, o que esta tabela mostra é que esta ameaça não é desencadeada diretamente pela execução de uma das ações de controle possíveis.

### 5.1.3. Derivação dos Requisitos de *Safety*

A derivação dos requisitos dos componentes de *safety* se dá com a definição de cenários de possíveis ocorrências das ações de controle não seguras e suas possíveis causas, e em seguida relacionam-se requisitos de *safety*.

A lista de cenários e causas é apresentada, como em (44). A Tabela 5.12 mostra os cenários e requisitos.

Tabela 5.12. Tabela de Cenários e Requisitos

| Cenário | Causa | Requisito | Alocação |
|---|---|---|---|
| Manobra orbital comandada e realizada fora de modo de operação adequado ocorrendo conflito com controle corrente da espaçonave levando a órbita indevida | Controlador recebe comando de execução de manobra orbital em modo incorreto mas manobra é iniciada | 1) O controlador não deve permitir uma ação de controle de manobra orbital ser executada fora do modo correto | Controlador |
| Ação de manobra orbital iniciada porém inibição da força ocorrendo muito tarde | Controlador não é capaz de sustentar operação em tempo real e perde deadline de desligar atuação | 2) O controlador deve em um pior-cenário de requisições de uso de CPU da missão garantir o tempo de execução das tarefas de controle | Controlador, Desenvolv. Sw |

Continua

Tabela 5.12 - Continuação

| | | | |
|---|---|---|---|
| Ação de manobra orbital iniciada porém não finalizada no tempo correto devido a erro de ordem do controlador | Controlador inicia manobra orbital, porém evento de bordo altera modo, e forças são desligadas fora do tempo correto | 3) O controlador deve mesmo na ocorrência de falhas finalizar atuação sobre órbita antes de alterar o modo<br>4) A lógica e sequência de ações na operação de uma manobra orbital deve garantir uma finalização segura da manobra mesmo no caso de falhas relevantes a mesma | Controlador,<br>Operação |
| Ação de manobra orbital iniciada porém inibição da força ocorrendo muito tarde devido a falha de atuador | Controlador comanda ação, porém equipamento não inibe atuação | 5) Os atuadores de força lineares devem inibir a atuação mesmo na ocorrência de falha simples no atuador | Atuador |
| Comando de torque é enviado mas não executado | Atuador em falha | 6) O controlador deve detectar desvios de comportamento dos atuadores | Controlador, Desenvolv. Cntr. |
| Comando de torque é enviado mas não aceito | Atuador em falha | 7) O controlador deve detectar erros no comando (comunicação) dos atuadores | Controlador, Atuadores |
| Comando de torque é enviado mas ocorre em tempo incorreto | Atuador em falha temporária | 8) O controlador deve detectar erros nos tempos de comandos (comunicação) | Controlador, Atuadores |
| Comando de torque não é enviado porém é executado | Atuador respondendo comando em time-out | 9) O controlador deve comandar um estado seguro do atuador caso este sinalize atuação em momento indevido | Controlador, Atuador |
| Comando de torque não é enviado porém é executado silenciosamente | Atuador em falha | 10) O controlador deve detectar atuação espúria visto desvios da dinâmica esperada para a planta | Controlador, Desenvolv. Cntr. |

Continua

Tabela 5.12 - Continuação

| | | | |
|---|---|---|---|
| Comandos de torque executados, porém com desempenho incorreto | Atuador em falha | 11) O controlador deve levar a um apontamento seguro caso detecte degradação de desempenho do controle | Controlador, Desenvolv. Cntr. |
| Comandos de torque não enviados devido a problemas no controlador | Controlador em falha | 12) O controlador deve detectar erros de hardware e software durante sua execução e chavear para modo de operação seguro em caso de falha. Obs. Erros de hardware: erro de memória, erro de processador. Erros de software: Divisões por zero, Valores numéricos fora de faixa (NaN, +-inf), erro na obtenção de recursos<br>13) O Controlador deve levar a um apontamento seguro caso haja perda dos tempos de execução das tarefas | Controlador |
| Sistema tem degradação de energia | Mecanismos de atuação dos painéis em falha | 14) O controlador deve levar a um apontamento seguro caso haja degradação de energia no satélite | Controlador |
| Estimativa de estado incoerente com a dinâmica esperada | Sensor em falha | 15) O controlador deve detectar leituras não coerentes de sensores. Obs. Descontinuidades e congelamento de valor | Controlador |
| Degradação de estimativa de estado devido a atrasos de dados de sensores | Sensor em falha | 16) O controlador deve detectar erros nos tempos dos dados dos sensores (comunicação) | Controlador |
| Degradação de estimativa de estado devido a falta de dados | Sensor em falha | 17) O controlador deve ser chavear para modo seguro após evoluir a dinâmica do sistema mesmo com sensor em falha por até X tempo | Controlador, Desenvolv. Cntr. |
| Degradação de estimativa de estado devido a erros múltiplos de sensores | Sensores em falha | 18) O controlador deve entrar em modo seguro de apontamento se houver simultaneamente mais de uma falha em sensores | Controlador |

Continua

Tabela 5.12 - Conclusão

| | | | |
|---|---|---|---|
| Degradação de desempenho de controle devido a ruídos e perturbações | Condições ambientais fora do esperado ou deterioração de equipamentos | 19) O controlador deve entrar em modo seguro caso o comportamento da planta comece a divergir do esperado mesmo se não houver falhas detectadas de equipamentos | Controlador |
| Degradação de mensagens de sensores devido a ruídos na comunicação | Condições ambientais fora do esperado ou deterioração de equipamentos | 20) O controlador deve detectar comunicação com dados corrompidos nos sensores digitais | Controlador |

A lista de requisitos elicitados foi considerada adequada por comparação com questões de FDIR discutidas na bibliografia de referência de sistemas AOCS. Dentre os 20 requisitos de *safety* acima listados um subconjunto de 17 requisitos será validado durante o experimento. A seleção do subconjunto de requisitos validados depende das funcionalidades implementadas no protótipo, **sistema sob teste,** e das facilidades de controle e observação providas pelo **sistema de teste**.

Os requisitos não validados são os não testáveis, como o requisito 2 que necessita de uma análise de escalonabilidade do sistema, e os não relacionados com o controlador, como o requisito 4 que é operacional e o 5 que é de atuador.

## 5.2. O Sistema sob Teste: Protótipo de AOCS

Esta seção descreve brevemente a construção de um protótipo de sistema de controle de atitude e órbita com FDIR.

O protótipo do AOCS, assim como todo sistema de controle de atitude e órbita tem como principal função específica e dedicada, a de controlar a atitude e executar manobras orbitais. Esta função é repetitiva (cíclica) enquanto o sistema estiver operando.

O protótipo foi desenvolvido para ser executado em um computador de bordo PC104 com arquitetura x86, Figura 5.5. O desenvolvimento levou em consideração a existência de um sistema de software proprietário desenvolvido para aplicações espaciais. A camada de software básico do sistema, contendo o sistema operacional e os serviços independentes de missão, foi reutilizada deste sistema proprietário. A parte lógica do protótipo AOCS, a qual é dependente da missão foi desenvolvida especificamente para este trabalho de mestrado.


Figura 5.5. Computador de Bordo com arquitetura x86

O protótipo AOCS possui as seguintes características:

- Três modos de operação: Nominal, Manobra Orbital e Emergência
- Estratégias de recuperação do FDIR do tipo Meio-satélite

O controlador inicia em modo de emergência (E), e um telecomando (TcN) irá chavear para o modo nominal (N), neste ponto telecomandos podem alterar entre o modo de manobra orbital (M) e nominal (N) até a ocorrência de evento de falha (ef) que provoca um chaveamento para o modo emergência (E). A

partir dai um novo telecomando pode alterar o modo para o nominal. Como apresenta a Figura 5.6.

O modo de emergência provê um apontamento seguro e é o início necessário do controle para o apontamento nominal.

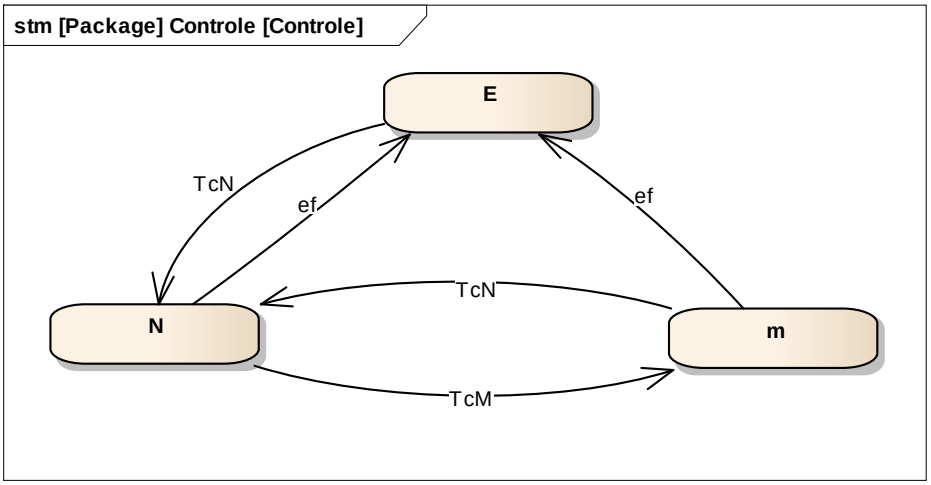


Figura 5.6. Descrição de mais alto nível do Protótipo de AOCS

Para facilitar o desenvolvimento do protótipo AOCS, já que este não é foco principal do trabalho, assumiu-se uma redundância idêntica dos equipamentos do satélite. Portanto, somente um conjunto de algoritmos de controle de atitude foi desenvolvido. Assumiu-se um sistema de AOCS com os seguintes equipamentos para cada um dos conjuntos redundantes:

- 1 Controlador
- 8 Sensores Solares – Analógicos
- 1 Sensor de Estrelas – Digital Serial
- 1 Unidade Girométrica de 3 eixos – Digital Serial
- 1 Magnetômetro de 3 eixos – Digital Serial
- 1 GPS – Digital Serial

- 3 Rodas de Reação, uma por eixo – Digital Serial
- 8 Propulsores, todos em um mesmo plano do satélite – Digital Bilevel
- 3 Magnetotorquers, um por eixo – Analógicos
- 2 Mecanismos de rotação de painel solar – Digital Serial
- 1 Sensor de sinalização de baixa energia, representando o mecanismo de monitoramento da energia do satélite – Digital Bilevel

A arquitetura de FDIR do protótipo AOCS seguiu a proposta de (40), na qual todos os eventos de falha discretos, como por exemplo: erro de comunicação; erro de memória; erro de software; quanto os erros comportamentais no tempo (contínuos), como por exemplo: desvio da dinâmica medida em relação à dinâmica esperada do satélite; se tornam entrada para uma função de diagnóstico que toma a decisão de recuperação de forma centralizada.

O desenvolvimento dos mecanismos de FDIR seguiram propostas tradicionais descritas na revisão bibliográfica, incluindo:

- Detecção de falhas na comunicação com equipamentos, falhas de tempo e integridade da informação
- Observadores para detecção de falhas na dinâmica de equipamentos
- Observador para detecção de falha na dinâmica do satélite
- Detecção de erro de hardware com indicação interna ao computador de bordo
- Detecção de erros de software com eventos gerados pelo software

Os mecanismos FDIR desenvolvidos no protótipo AOCS podem ser relacionados com os requisitos, como apresentado na Tabela 5.13.

Tabela 5.13. Requisitos vs. Mecanismos FDIR.

| Requisitos | Mecanismo FDIR |
|---|---|
| 1) O controlador não deve permitir uma ação de controle de manobra orbital ser executada fora do modo correto | a) Conferência do modo antes da execução de comando de manobra orbital |
| 2) O controlador deve em um pior-cenário de requisições de uso de CPU da missão garantir o tempo de execução das tarefas de controle | b) NA |
| 3) O controlador deve mesmo na ocorrência de falhas finalizar atuação sobre órbita antes de alterar o modo | c) Sinalização de falhas de Sw e Hw durante a execução de manobra e finalização de manobra em caso de falha |
| 4) A lógica e sequência de ações na operação de uma manobra orbital deve garantir uma finalização segura da manobra mesmo no caso de falhas relevantes a mesma | d) NA |
| 5) Os atuadores de força lineares devem inibir a atuação mesmo na ocorrência de falha simples no atuador | e) NA |
| 6) O controlador deve detectar desvios de comportamento dos atuadores | f) Observação da dinâmica dos atuadores |
| 7) O controlador deve detectar erros no comando (comunicação) dos atuadores | g) Detecção de mensagem corrompida por campo de validação |
| 8) O controlador deve detectar erros nos tempos de comandos (comunicação) | h) Time-out de aceitação de comando pelo atuador |
| 9) O controlador deve comandar um estado seguro do atuador caso este sinalize atuação em momento indevido | i) Detecção de comunicação indevida (ex. mensagem fora de hora, latch-up de porta) |
| 10) O controlador deve detectar atuação espúria visto desvios da dinâmica esperada para a planta | j) Observação da dinâmica da planta |
| 11) O controlador deve levar a um apontamento seguro caso detecte degradação de desempenho do controle | k) Cálculo de parâmetros de desempenho e conferência se dentro de faixa nominal |

Continua

Tabela 5.13 - Conclusão

| | |
|---|---|
| 12) O controlador deve detectar erros de hardware e software durante sua execução e chavear para modo de operação seguro em caso de falha. Obs. Erros de hardware: erro de memória, erro de processador. Erros de software: Divisões por zero, Valores numéricos fora de faixa (NaN, +-inf), erro na obtenção de recursos | l) Sinalização de evento de falha pelo hardware ou software do AOCS |
| 13) O Controlador deve levar a um apontamento seguro caso haja perda dos tempos de execução das tarefas | m) Sinalização de evento de perda de período de execução pelo software do AOCS |
| 14) O controlador deve levar a um apontamento seguro caso haja degradação de energia no satélite | n) Conferência do nível de energia do satélite, se encontra dentro do nominal |
| 15) O controlador deve detectar leituras não coerentes de sensores. Obs. Descontinuidades e congelamento de valor | o) Observação da dinâmica de sensores |
| 16) O controlador deve detectar erros nos tempos dos dados dos sensores (comunicação) | p) Time-out de recebimento de mensagem de sensor |
| 17) O controlador deve chavear para modo seguro após evoluir a dinâmica do sistema mesmo com sensor em falha por até X tempo | q) Detecção e contagem de mensagem perdidas ou inválidas de sensores |
| 18) O controlador deve entrar em modo seguro de apontamento se houver simultaneamente mais de uma falha em sensores | r) Conferência de estado de sensores |
| 19) O controlador deve entrar em modo seguro caso o comportamento da planta comece a divergir do esperado mesmo se não houver falhas detectadas de equipamentos | s) Observação da dinâmica da planta |
| 20) O controlador deve detectar comunicação com dados corrompidos nos sensores digitais | t) Detecção de mensagem corrompida por campo de validação |

Os mecanismos FDIR e as falhas que levam ao seu disparo nos modelos são apresentados na tabela 5.14.

Tabela 5.14. Mecanismos FDIR e Falhas relacionadas

| Mecanismo FDIR | Falha Sensor | Falha Controlador | Falha Atuador | Falha Sensor - Controlador | Falha Controlador - Atuador |
|---|---|---|---|---|---|
| a) Conferência do modo antes da execução de comando de manobra orbital | | (i) Controlador recebe comando de execução de manobra fora do modo correto<br>ou modo altera antes da execução | | | |
| b) NA | | | | | |
| c) Sinalização de falhas de Sw e Hw durante a execução de manobra e finalização de manobra em caso de falha | | (i) Falha de operação do software (miss-deadline, erro de obtenção de recurso, etc.)<br>(ii) Falha de operação do hardware (bitflip, erro de processador, etc.) | | | |
| d) NA | | | | | |
| e) NA | | | | | |
| f) Observadores da dinâmica dos atuadores | | | (i) Mau funcionamento do atuador atua incorretamente | | (i) Atrasos no canal de comunicação causam atrasos na dinâmica do atuador |
| g) Detecção de mensagem corrompida por campo de validação | | | (i) Dado corrompido por mau funcionamento do atuador | | (i) Dado corrompido por mau funcionamento do atuador ou canal |
| h) Time-out de aceitação de comando pelo atuador | | | (i) Atuador não responde | | (i) Atrasos no canal de comunicação causam atraso fora da janela de aceitação<br>(ii) Perda da mensagem no canal |
| i) Detecção de comunicação indevida (ex. mensagem fora de hora, latch-up de porta) | | | (i) Mau funcionamento do atuador atua em momento incorreto | | (i) Atrasos no canal de comunicação fazem resposta chegar em momento inesperado após time-out |
| j) Observador da dinâmica da planta | | | (i) Atuador atua em momento incorreto | | (i) Pequenos atrasos fazem a dinâmica da planta ter atraso de comportamento |

Continua

Tabela 5.13 - Continuação

| | | | | | |
|---|---|---|---|---|---|
| k) Cálculo de parâmetros de desempenho e conferência se dentro de faixa nominal | (i) Falha não detectável de sensores | (i) Falha nos cálculos de comandos de atuação<br>(ii) Ruídos e perturbações externas | (i) Falha não detectável de atuadores | | |
| l) Sinalização de evento de falha pelo hardware ou software base do AOCS | | (i) Falha do hardware OBC detectáveis<br>(ii) Falha no software causado por falha silenciosa de hardware ou degradação de condições de operação do hardware/software | | | |
| m) Sinalização de evento de perda de período de execução pelo software do AOCS | | (i) Falha de hardware ou software leva a perda de deadline de tarefa | | | |
| n) Conferência do nível de energia do satélite se dentro do nominal | | (i) Sinalização de baixa energia do satélite | | | |
| o) Observadores da dinâmica de sensores | (i) Mau funcionamento do sensor lê valores incorretos<br>(ii) Ruídos de leitura | | | (i) Atrasos no canal de comunicação causa erro da dinâmica do sensor | |
| p) Time-out de recebimento de mensagem de sensor | (i) Sensor não envia dado | | | (i) Atrasos no canal de comunicação causam atraso fora da janela de aceitação do dado<br>(ii) Perda da mensagem no canal | |
| q) Detecção e contagem de mensagem perdidas ou inválidas de sensores | (i) Dado corrompido por mau funcionamento do sensor<br>(ii) Sensor não envia dado | | | (i) Atrasos no canal de comunicação causam atraso fora da janela de aceitação do dado<br>(ii) Perda da mensagem no canal<br>(iii) Dado corrompido por mau funcionamento do sensor ou canal | |
| r) Conferência de estado de sensores | (i) Mau funcionamento de mais de um sensor | | | (i) Atrasos em canais de mais de um sensor<br>(ii) Perda da mensagem em mais de um canal de sensores | |

Continua

Tabela 5.13 - Conclusão

| | | | | | |
|---|---|---|---|---|---|
| s) Observador da dinâmica da planta | (i) Falha não detectável de sensores | (i) Falha nos cálculos de comandos de atuação<br>(ii) Ruídos e perturbações externas | (i) Falha não detectável de atuadores | | |
| t) Detecção de mensagem corrompida por campo de validação | (i) Dado corrompido por mau funcionamento do sensor | | | (i) Dado corrompido por mau funcionamento do sensor ou canal | |

## 5.3. O Sistema de Teste

A arquitetura geral do sistema de teste é apresentada aqui devido a sua importância para identificação dos Pontos de Observação e Controle (PCO), os quais são elementos fundamentais na execução dos testes e, são importantes para aplicação das metodologias de teste.

O sistema de teste adotado se baseia em um sistema comum para teste de aplicações AOCS, ver seção 2.4., consistindo de simuladores do hardware (sensores e atuadores) e de um ambiente que se comunica com o sistema sob teste (o computador de bordo com o software AOCS embarcado). No contexto de testes de integração alguns simuladores de hardware deverão ser substituídos pelo próprio hardware do equipamento real. A Figura 5.7 apresenta os elementos arquiteturais do sistema de teste e do protótipo AOCS. Destacam-se os Pontos de Observação e Controle sendo: Operação de simulação e Emulador da Estação Solo. Cada elemento (quadrado) da Figura 5.7 representa um computador/equipamento distinto, e os Sensores e Atuadores reais podem estar conectados ao Computador de Bordo ou não.

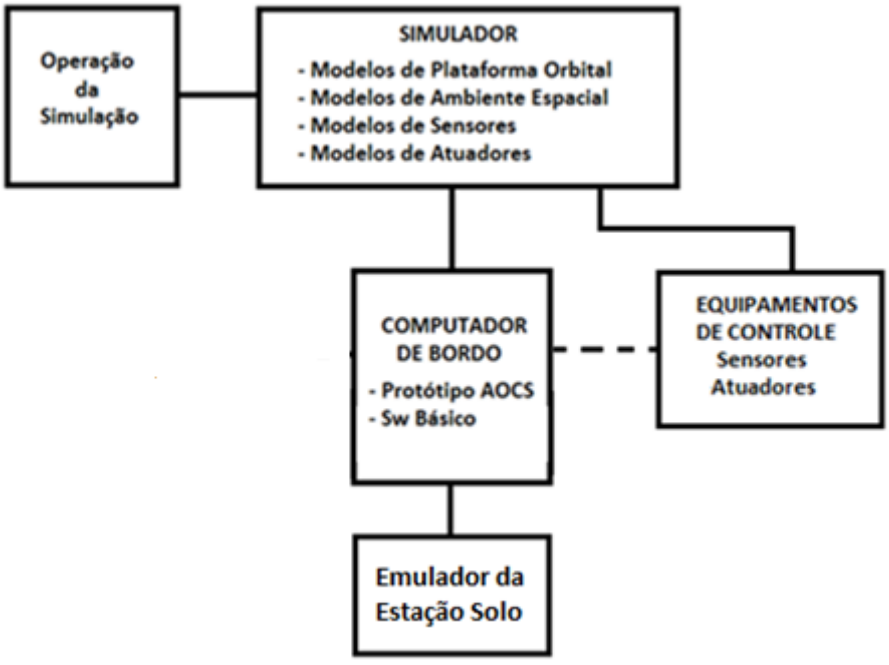


Figura 5.7. Integração do Sistema de Teste e Sistema sob Teste.

## 5.4. Aplicação da Metodologia CoFI

A primeira atividade realizada para aplicação da metodologia COFI foi estudar os requisitos e entender o contexto e o comportamento do protótipo AOCS.

No passo de **Identificação** foram identificados os serviços relacionados ao tratamento de FDIR, no sentido de uma função a ser executada pelo protótipo AOCS, os seguintes **serviços** foram identificados:

(i) Serviço de Mudança de Modo,

(ii) Serviço de Controle de Atitude (Modo Nominal)

(iii) Serviço de Manobra Orbital (Modo Manobra Orbital)

(iv) Serviço de Monitoramento de Saúde

Outra identificação necessária na metodologia CoFI é o **Sistema de Teste**, compreendendo as ferramentas planejadas para executar os testes, tal sistema é mostrado de forma geral na Figura 5.7, mas a Figura 5.8 ilustra o sistema de teste indicando os Pontos de Controle e Observação, e as entradas e saídas para realização dos testes de conformidade pela metodologia CoFI. O sistema de teste contém facilidades para simular os equipamentos sensores e atuadores, gerando valores como se estivessem operando em ambiente espacial, e a estação solo envia telecomandos e recebe telemetria do Protótipo AOCS.

Os Pontos de Controle e Observação na Figura 5.8 são indicados como PC – pontos de controle, através dos quais são inseridos os eventos possíveis do sistema (entradas) e PO – pontos de observação, por onde se observa as ações (ou saídas) geradas pelo SUT.

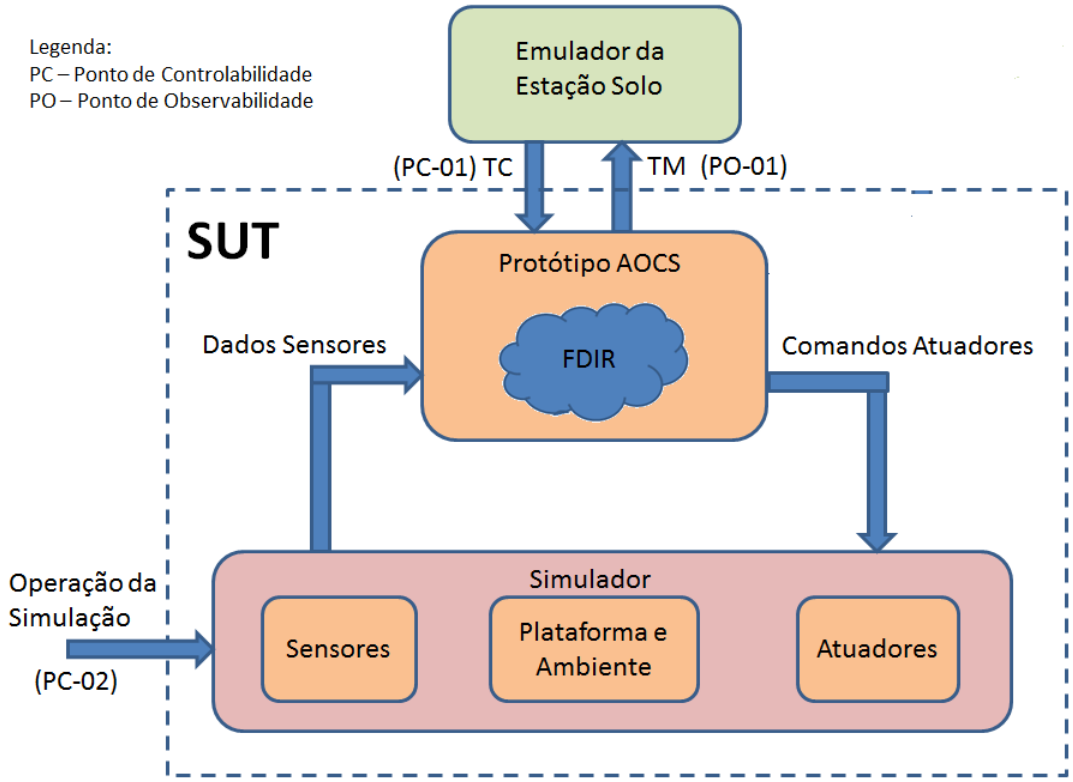


Figura 5.8. Sistema sob teste e Sistema de teste para CoFI

Nessa configuração observa-se que o protótipo AOCS bem como as funções FDIR são consideradas como caixa preta. Cabe observar também que o Sistema de Teste não provê facilidades para provocar falha de hardware no OBC, entretanto alguns modelos ilustram casos em que eventos de falhas de hardware são considerados para apresentar o comportamento do sistema frente a tais eventos. Os eventos de entradas e saídas identificados do conjunto de sistema de teste e sistema sob teste são apresentados nas tabelas a seguir.

São apresentadas as tabelas parciais de Entradas, Saídas, Falhas de Hardware e Exceções Especificadas. As tabelas completas se encontram no APÊNDICE B. E em seguida são apresentados os serviços.

Tabela 5.15. Tabela parcial das entradas

| PC | Evento de entrada | Descrição |
|---|---|---|
| PC-02 | Injeta_Falha_Dinamica_Sensor | Insere falha de dinâmica em um sensor (Descontinuidade ou congelamento de dado) |
| | Injeta_Falha_Dados_Sensor | Insere falha nos dados de um sensor por X ciclos |
| | Injeta_Falha_Sensores | Insere falha em dois sensores em um mesmo ciclo |
| | Injeta_Falha_Dinamica_Planta | Insere falha de dinâmica da planta |
| | Injeta_Falha_Energia_Planta | Comanda sinalização de baixa energia do satélite |
| | Injeta_Falha_Dinamica_Atuador | Insere falha da dinâmica de um atuador |
| Interno | Início_de_ciclo | Início de ciclo de controle, gerado por temporizador interno do sistema |
| | Fim_monitora_sensores_ok | Finalização do monitoramento de sensores sem encontrar falhas |
| | Fim_monitora_sensores_nok | Finalização do monitoramento de sensores com a descoberta de falha |
| | Fim_monitora_planta_ok | Finalização do monitoramento da planta sem encontrar falhas |
| | Fim_monitora_planta_nok | Finalização do monitoramento da planta com a descoberta de falha |
| | Fim_monitora_atuadores_ok | Finalização do monitoramento de atuadores sem encontrar falhas |
| | Fim_monitora_atuadores_nok | Finalização do monitoramento de atuadores com a descoberta de falha |
| | Fim_monitora_energia_ok | Finalização do monitoramento da energia do satélite sem encontrar falhas |
| | Fim_monitora_energia_nok | Finalização do monitoramento da energia do satélite com a descoberta de falha |

Tabela 5.16. Tabela parcial das falhas de hardware

| Falha | Evento | Descrição |
|---|---|---|
| Falha de Hardware OBC | Injeta_Falha_ HardwareOBC | Falhas no Hardware do Computador de Bordo sinalizadas, como bitflip em memória e erro de CPU. |

Tabela 5.17. Tabela parcial das saídas

| PO | Saídas | Descrição |
|---|---|---|
| PO-01 | Requere_Modo_ Emergência | Indicação por TM de requisição interna para mudança para modo de operação emergência |
| | Sinaliza_Falha_Din_ Sensor | Sinalização por TM de falha na dinâmica de um sensor |
| | Sinaliza_Falha_Dado_ Defasado | Sinalização por TM de falha por dado de sensor muito defasado (mais de x ciclos) |
| | Sinaliza_Falha_na_ Dinamica_do_Satélite | Sinalização por TM de falha na dinâmica do satélite |
| | Sinaliza_Falha_na_ Dinamica_do_Atuador | Sinalização por TM de falha na dinâmica de um atuador |
| | Sinaliza_Degradação_ Nível_Energia | Sinalização por TM de falha por nível de energia do satélite abaixo do nominal |
| | Sinaliza_Falha_ Simultanea_de_ Sensores | Sinalização por TM de falha simultânea em mais de um sensor |
| | Sinaliza_Falha_ HardwareOBC | Sinalização por TM de falha interna ao hardware do OBC |
| | Nenhuma_Saída | Sucesso (nenhuma falha) na execução do controle |

Tabela 5.18. Tabela parcial das exceções especificadas do sistema

| Requisito | Descrição | Máquina COFI |
|---|---|---|
| 6 | O controlador deve detectar desvios de comportamento dos atuadores | S_MS_EE |
| 10 | O controlador deve detectar atuação espúria visto desvios da dinâmica esperada para a planta | S_MS_EE |
| 12 | O controlador deve detectar erros de hardware e software durante sua execução e chavear para modo de operação seguro em caso de falha. Obs. Erros de hardware: erro de memória, erro de processador. Erros de software: Divisões por zero, Valores numéricos fora de faixa (NaN, +-inf), erro na obtenção de recursos | S_MS_FH |
| 14 | O controlador deve levar a um apontamento seguro caso haja degradação de energia no satélite | S_MS_EE |
| 15 | O controlador deve detectar leituras não coerentes de sensores. Obs. Descontinuidades e congelamento de valor | S_MS_EE |
| 17 | O controlador deve ser chavear para modo seguro após evoluir a dinâmica do sistema mesmo com sensor em falha por até X tempo | S_MS_EE |
| 18 | O controlador deve entrar em modo seguro de apontamento se houver simultaneamente mais de uma falha em sensores | S_MS_EE |
| 19 | O controlador deve entrar em modo seguro caso o comportamento da planta comece a divergir do esperado mesmo se não houver falhas detectadas de equipamentos | S_MS_EE |

Após a fase de identificação, inicia-se a fase de Modelagem. A modelagem do protótipo AOCS de acordo com a metodologia CoFI levou em consideração os requisitos do software definidos segundo a STPA, ver seção 5.1.

Observa-se que segundo a metodologia, apenas os eventos previstos e as falhas que podem ser executados através do Sistema de teste disponível são usados na modelagem, isso evita a geração de casos de teste não executáveis.

Antes de iniciar a modelagem, alguns tipos de eventos foram padronizados:

I. **Evento Fim de Função:** Em um sistema ativo, existem encadeamentos de funções que são executados em um ciclo. Quando o fim da computação de uma função é o fator de início da próxima função, ele foi descrito como um evento. Por exemplo, o fim da etapa de sensoriamento inicia a etapa de computação do controle, que quando finaliza inicia a etapa de atuação. O resultado destas etapas é modelado com um evento representando o resultado da execução.

II. **Evento de início de ciclo:** Como o sistema é ativo, ele irá começar a executar funções dado um período de tempo decorrido desde o último início de suas funções. Para representar essa funcionalidade nos modelos foi utilizado o evento de início de ciclo.

III. **Evento "Nenhuma saída" de Sucesso**: Quando um ciclo de controle termina havendo sucesso em todas as funções nenhum evento de sinalização de fim do ciclo ou sucesso é gerado, simplesmente o software entra em modo de espera do próximo período de execução. A ausência de sinalizações de falhas no fim do ciclo é uma indicação de que o processamento terminou corretamente. Portanto, o evento "Nenhuma saída" foi criado como um evento de saída "visualizável" de fim de um ciclo com sucesso, este evento é representado pela não ocorrência de nenhum evento de falha. Este evento poderia ser confundido com um travamento do software, porém este segundo causaria um reinício do sistema e portanto eles são distinguíveis.

Como o escopo deste trabalho no sistema de controle testa basicamente a parte do sistema que é a ativa, que força uma sequência pré-definida das

funções, estas não ocorrem fora de ordem, por isso os modelos CoFI de caminhos furtivos, que representam caminhos de execução fora da ordem normal não foram criados para a maioria dos serviços. Por exemplo, no serviço de controle de atitude não acontece de uma função de atuação ocorrer antes das funções de cálculo de controle. Portanto as funções ativas deste sistema se mostraram por especificação naturalmente resistentes a esse tipo de falha.

A seguir são apresentados os modelos normal, de exceções especificadas, e de tolerância a falhas de hardware do Serviço Monitoramento de Saúde do protótipo AOCS.

O modelo normal da COFI para o serviço Monitoramento de Saúde é apresentado na Figura 5.9.

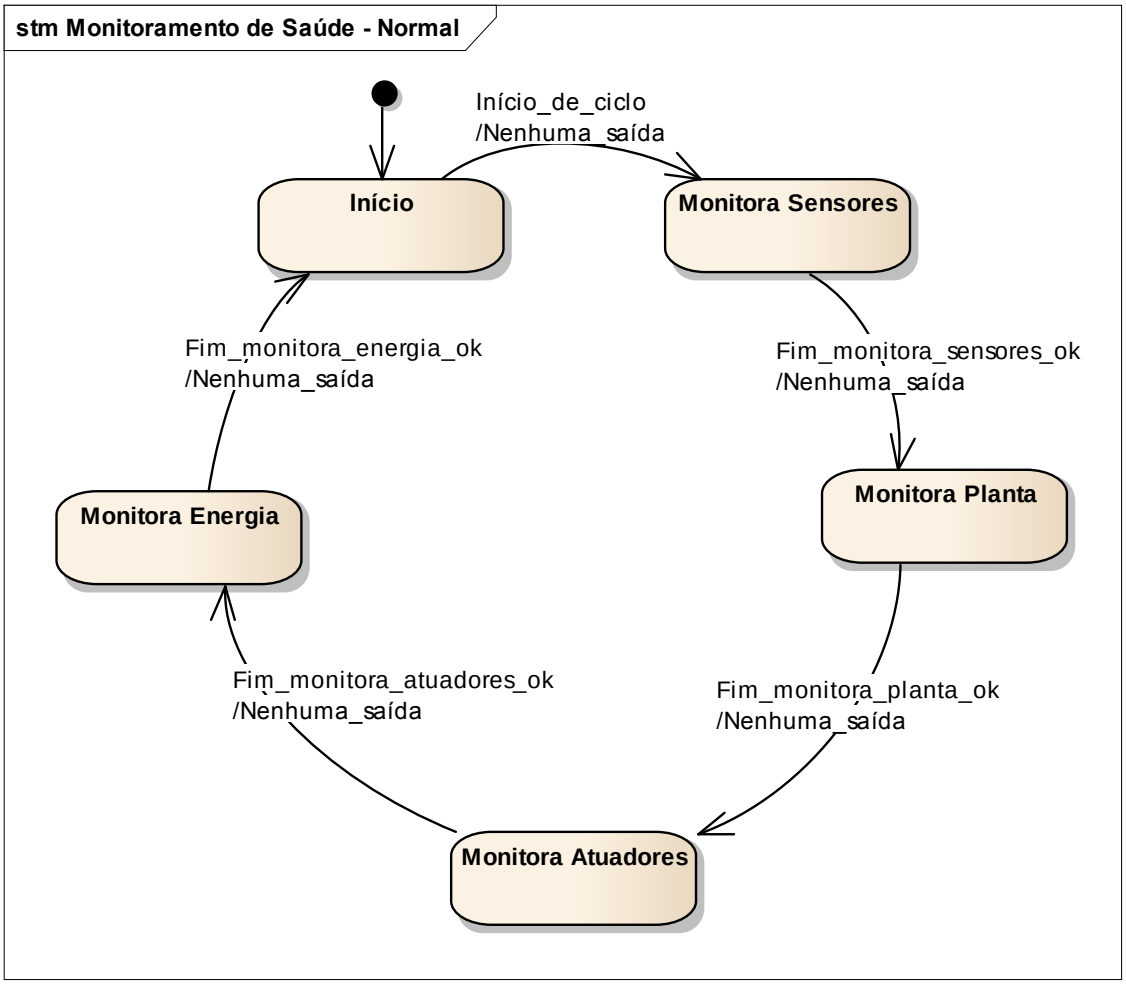


Figura 5.9. S_MS_N: Modelo Normal do Serviço Monitoramento de Saúde

O controle da atitude inicia com a ocorrência do evento cíclico, e o fim da execução das funções de controle geram os eventos de entrada e saída da máquina, sendo as saídas nulas representando sucesso da função. No caso de sucesso em toda operação do modo de operação do controle nenhuma saída visível é observável. O sucesso significa que a atitude está sendo controlada como esperado e o sistema AOCS irá dormir esperando o próximo ciclo para realizar as mesmas funções novamente.

A Figura 5.10 apresenta o modelo de exceções especificadas.

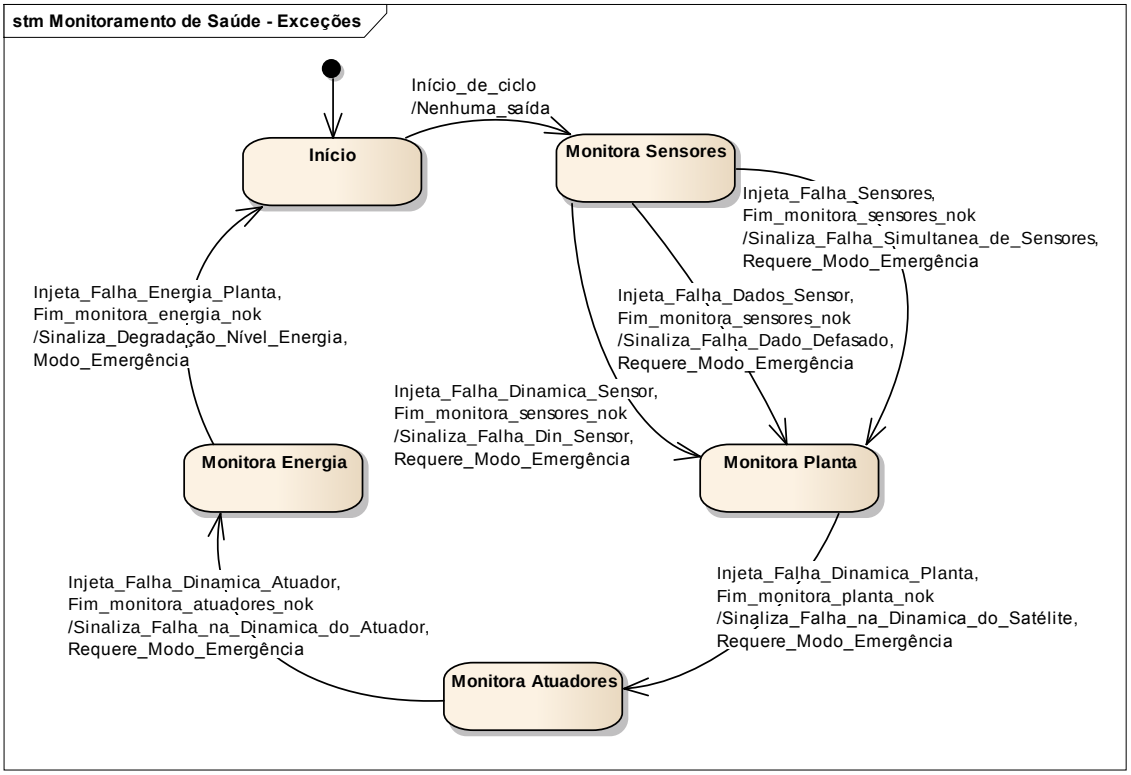


Figura 5.10. S_MS_EE: Modelo de Exceções Especificadas do Serviço Monitoramento de Saúde

Outras características da modelagem citadas anteriormente podem ser vistas, como o sequenciamento de funções mesmo com os diferentes resultados de sucesso ou falha nas funções predecessoras.

A Figura 5.11 apresenta o modelo de tolerância a falhas de hardware.

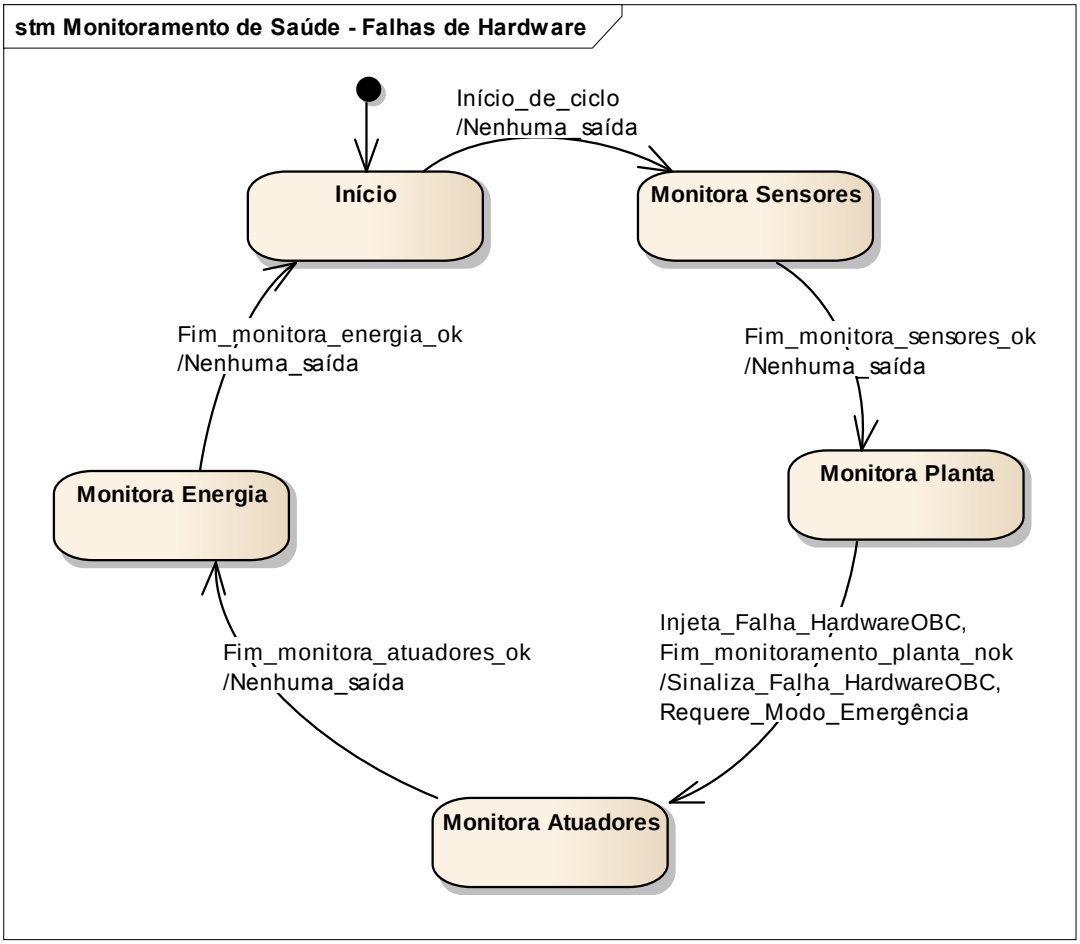


Figura 5.11. S_MS_TF: Modelo de Tolerância a Falhas de Hardware do Serviço Monitoramento de Saúde

Os modelos criados para os demais serviços: Serviço Mudança de Modo; Serviço Controle de Atitude; e Serviço Manobra Orbital; são apresentados no Apêndice B e os resultados da aplicabilidade da CoFI são discutidos na seção 5.6.

## 5.5. Aplicação da Metodologia InRob

A aplicação da metodologia InRob nesse trabalho visa apoiar a verificação e validação dos mecanismos FDIR do protótipo AOCS no tocante aos requisitos de tempo real. Enquadra-se no contexto de integração de subsistemas para validar as propriedades de tempo real do AOCS associadas à interoperabilidade do protótipo AOCS com os equipamentos reais sensores e

atuadores. Apesar de também aplicável na integração de componentes internos do software AOCS, integração software-software, ver seção 2.5.1, é importante destacar que esforço adicional na arquitetura de teste seria necessário para prover injetores de defeitos nos canais de comunicação internos ao protótipo, como abordado em (57).

Os componentes do sistema modelados no nível de abstração de subsistemas interoperantes são: protótipo software AOCS (Controlador), Sensores, Atuadores, e suas interfaces de comunicação providas com emuladores de defeitos (FEM), como mostra a Figura 5.12.

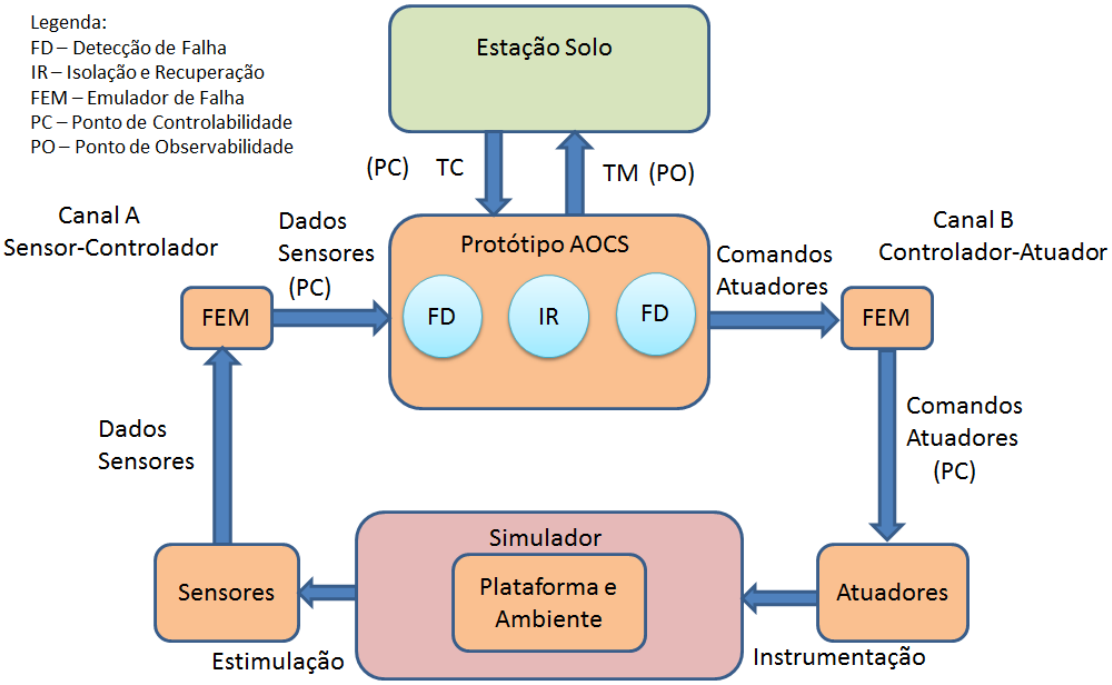


Figura 5.12. Sistema sob teste e Sistema de teste para InRob

Conforme recomendado pela InRob, um emulador de defeito no canal de comunicação (FEM) faz-se necessário como ponto de controle (PC) na interface inferior, sendo fundamental para a execução dos casos de testes. Na terminologia da InRob, as entradas e saídas via Estação Solo representam as interfaces superiores e são utilizadas como pontos de controle (PC) e

observação (PO). O simulador tem a função de simular o ambiente espacial e a plataforma do satélite, alvo de controle do AOCS.

### 5.5.1. Modelagem InRob

A modelagem, segundo a metodologia InRob, é parte da **Fase A,** a qual compreende a geração do **perfil de serviços**, geração dos **modelos nominais** dos serviços de interesse, a análise do modelo nominal em termos dos **perigos relacionado a tempo**, e a extensão de cada modelo nominal com os perigos associados ao serviço gerando o **modelo aumentado**.

No presente trabalho, o passo de mapeamento do perfil dos serviços recomendado na InRob não foi necessário, dado o escopo reduzido das funcionalidades implementadas no protótipo AOCS ter priorizado os serviços mais críticos do sistema em termos de segurança (safety), a saber: (i) controlar atitude do satélite e (ii) realizar manobra orbital, realizados respectivamente nos modos de operação Nominal (N) e Manobra Orbital (M) no diagrama da Figura 5.6. Na InRob, o passo de perfil operacional apoia a priorização de esforços na construção dos modelos dos serviços mais críticos realizados conjuntamente pelos componentes em integração. No caso do protótipo AOCS, ambos os serviços (i) e (ii) são realizados de forma colaborativa pelos componentes do sistema AOCS alvo da integração e, por essa razão, as questões críticas de tempo real associadas à interoperabilidade em ambos os serviços foram modeladas.

A modelagem do comportamento esperado dos componentes **Controlador** (Software AOCS), **Sensores** e **Atuadores** na realização dos dois serviços levou em consideração os requisitos do software definidos segundo a STPA, ver seção 5.1.

Para cada serviço foram especificados:

- Um cenário comportamental nominal de interoperabilidade dos componentes envolvidos, contemplando o funcionamento nominal dos componentes. Tais cenários nominais foram descritos utilizando o

formalismo TIOA (Timed Input Output Automata) recomendado pela InRob e compõem **os modelos nominais do serviço.**

- Cada cenário nominal foi analisado em termos de situações perigosas em relação a **desvios de tempo**: desvios insignificantes (minor Deviation) e desvios significativos (Major Deviation).

- Cada modelo nominal foi estendido com os desvios de tempo formando o **conjunto de modelos aumentados do serviço**.

A seguir os 3 passos acima descritos que compreendem a Fase A, de construção dos modelos InRob, serão exemplificados na modelagem do serviço de controle de atitude AOCS.

#### 5.5.1.1. Serviço de Controle de Atitude

Um **cenário nominal** de interoperabilidade do **controlador** com **um atuador digital** real é modelado com o envio de um comando (!Acomm) do controlador para o atuador digital, por meio do canal de comunicação (B) da Figura 5.12. Esse comando deverá ser recebido pelo atuador digital (?Acomm) que depois de validá-lo deverá retornar uma resposta OK (!Comm_ok) ou (!Comm_nok) ao controlador. A resposta deverá ser recebida pelo controlador. No caso de ser nok, o mecanismo de detecção de falha (FDIR) no controlador deve sinalizar uma falha de operação (!FDet_Op). A Figura 5.13 apresenta o modelo nominal do comportamento de interoperabilidade especificado para os dois componentes envolvidos utilizando o formalismo TIOA, conforme recomendado pela InRob. Para destacar o comportamento esperado do mecanismo FDIR abordado no cenário, optou-se por separá-lo em uma terceira entidade no modelo. Os modelos do controlador e FDIR poderão ser compostos por meio do produto síncrono das duas máquinas TIOA.

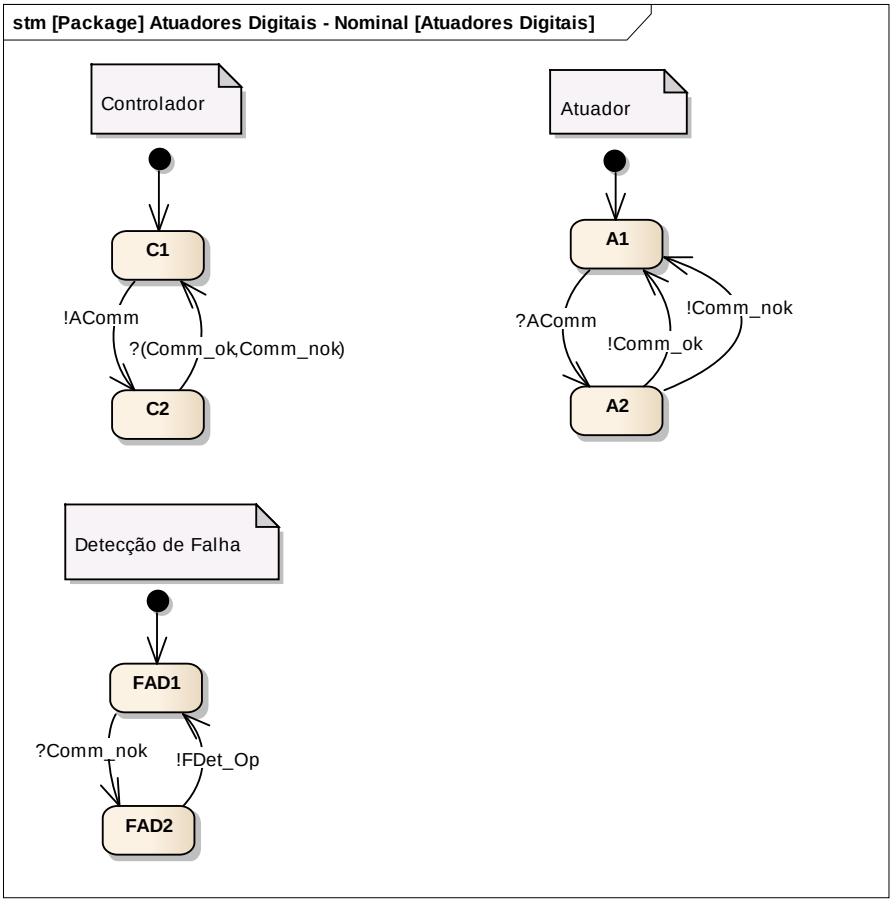


Figura 5.13. Modelo Nominal InRob CN_CAD_1: Controlador e Atuadores Digitais

Os seguintes desvios de tempo são identificados para o modelo:

Tabela 5.19. Desvio do Modelo InRob

| **Classe do desvio** | **Perigos** |
|---|---|
| Maior (majD) | Atrasos que extrapolam a janela de aceite do comando de atuação |
| Menor (minD) | Atrasos que não extrapolam a janela de aceite do comando de atuação |
| Perda (TO) | Perda da mensagem, considerada um atraso infinito |

Neste ponto é interessante perceber que os desvios podem ocorrer na mensagem de comando de atuação do controlador para o atuador, ou também

ocorrer da mensagem de confirmação do atuador para o controlador. Neste caso, em ambos cenários o resultado é idêntico, o controlador irá perceber um desvio do tempo na recepção da aceitação do comando, que pode causar uma falha de comunicação, portanto os cenários tem equivalência de causa-efeito. Com isso foi modelado somente o desvio em uma componente, como pede a metodologia InRob.

Também é importante destacar a necessidade de representar no modelo o comportamento esperado no caso de atraso infinito na recepção da resposta. Para gerenciar essa situação um mecanismo FDIR capaz de identificar o tempo esgotado (*time-out*) é comumente utilizado para garantir robustez na comunicação de sistemas de tempo real e seu funcionamento correto deve ser verificado na integração. Portanto, a análise de perigo evidencia a presença desse mecanismo FDIR nos modelos aumentados referentes ao Controlador, Atuador e Detecção de Falha com os respectivos eventos: !Acomm, !A_TO, e FDet_Op.

A metodologia InRob, com o formalismo TIOA, utiliza de um relógio global entre as máquinas para trabalhar a questão do tempo real. As máquinas TIOA podem então associar o instante de ocorrência aos eventos representados nas transições. Para lidar com *time-outs*, transições iniciam ou reiniciam contadores de tempo baseado no relógio global entre as máquinas. Caso um contador estoure um período de tempo (*time-out*) definido pela especificação, a representação no modelo é um evento de quiescência (mimica de um alarme recebido pelo relógio global) representado como transição na máquina que iniciou o contador. Este evento pode representar *time-outs* de comunicação entre componentes do sistema, ou caracterizar requisitos de tempo de operação das máquinas TIOA, ex. uma máquina pode ter um requisito de tempo para alcançar um determinado estado, que representa a necessidade da operação ocorrer em um determinado tempo. Esta modelagem de clock possibilita a metodologia InRob gerar casos de teste que considerem questões de tempo real na integração de sistemas. Na Figura 5.14 o evento A_TO

representa o *time-out* da comunicação do controlador com o equipamento atuador.

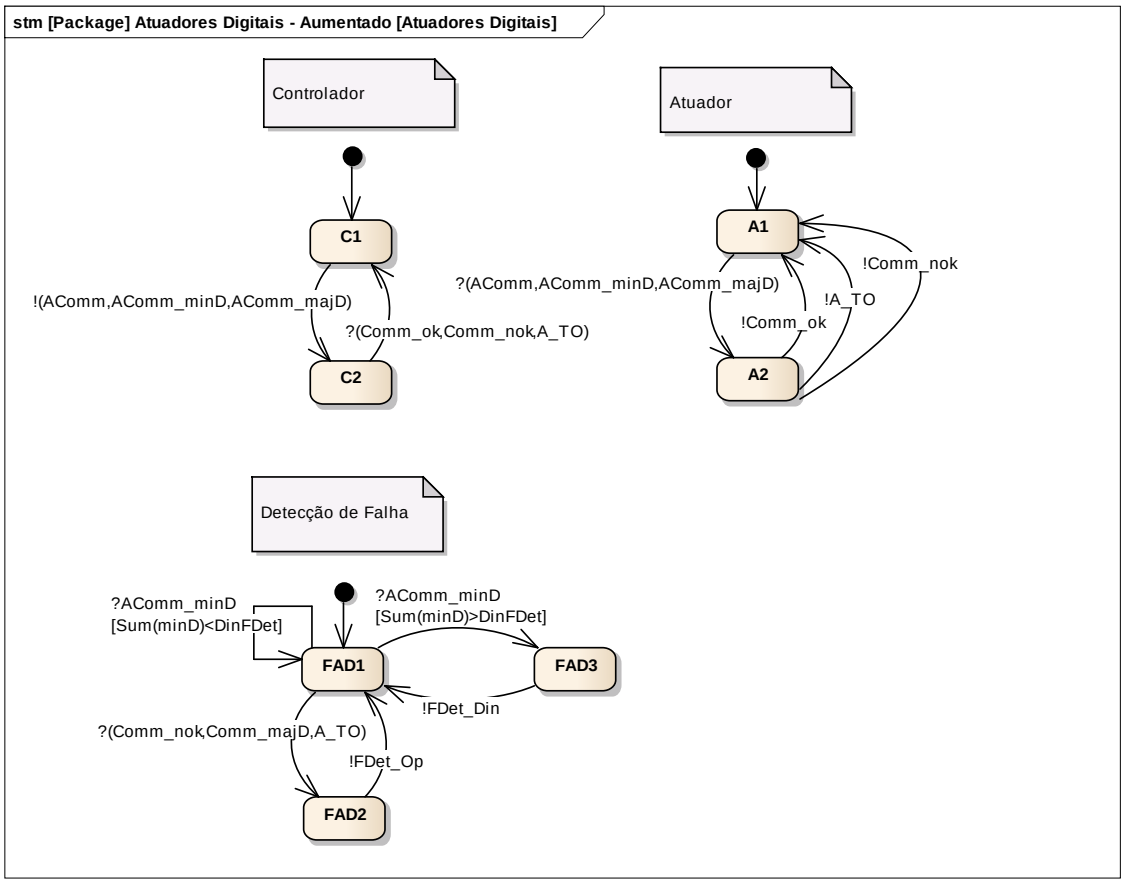


Figura 5.14. Modelo Aumentado InRob CA_CAD_1: Controlado e Atuadores Digitais

O **modelo aumentado** com os perigos de tempo é apresentado na Figura 5.14, geram casos contendo os perigos de tempo.

O modelo apresenta os desvios e as transições resultantes desta extensão. No caso de perda de mensagem (!A_TO) uma falha de operação (!FDet_Op) é detectada.

No caso de um comando do controlador sofrer um grande atraso (!AComm_majD), o atuador irá responder, pois é indiferente aos tempos de recebimento dos comandos (?AComm_majD) e responde com as respostas idênticas as nominais (!Comm_ok ou !Comm_nok), porém haverá o estouro da

janela de tempo de recebimento de resposta por parte do controlador, disparando a detecção de uma falha de operação (!FDet_Op).

No caso de um comando do controlador sofrer um pequeno atraso (!AComm_minD), inicialmente nada ocorre, pois as respostas do atuador ficam dentro da janela de tempo de recebimento de resposta por parte do controlador. Porém caso pequenos atrasos ocorram repetidamente no sistema, a dinâmica do atuador irá ser influenciada pele acumulo de atrasos, até o momento que ocorre uma falha da dinâmica do atuador, que desvia acima de um limite, do esperado (!FDet_Din).

O exemplo de aplicação da metodologia InRob descrito anteriormente cobre apenas um subconjunto de cenários dos serviços AOCS modelados pela InRob. Os demais modelos referentes aos cenários InRob encontram-se no APÊNDICE A.

É notável que o comportamento de interoperabilidade do controlador e atuador digital nos cenários modelados no exemplo acima ocorre nos dois serviços (i) controle de atitude e (ii) controle de manobra. Portanto, os modelos acima serão reutilizados na geração de casos de testes de ambos os serviços.

## 5.6. Avaliação da Aplicabilidade das Metodologias

A modelagem por ambas as metodologias foi realizada com sucesso. A caracterização dos mecanismos FDIR nos modelos foi possível, condição essencial para que os casos de testes gerados a partir dos modelos cobrissem as propriedades de FDIR.

No caso da metodologia CoFI, a ferramenta Condado (58), que gera testes a partir de MEFs, garante que casos de teste cubram os mecanismos modelados.

No caso da metodologia InRob, uma ferramenta que implemente, por exemplo, a técnica *model-checking*, permitirá gerar automaticamente casos de testes por meio da verificação de propriedades especificadas (descritas como propósito

de teste) no modelo InRob. Em (17) foi utilizada a ferramenta HJ2IF que simula a execução dos modelos InRob descritos na linguagem IF verificando os objetivos de teste (subconjunto de propriedades do modelo) descrito como Propósito de Teste (*Test Purpose*). O caminho utilizado pela ferramenta para verificar o conjunto de propriedades buscadas é registrado como caso de teste. Desta forma, torna-se possível a geração automática de casos de teste que cubram os mecanismos de FDIR representados nos modelos InRob.

As metodologias não foram aplicadas em sua totalidade, algumas adequações ao escopo do trabalho foram feitas, com isso o potencial total das metodologias não foi exercitado e conclusões relacionados aos passos não executados ficam fora da discussão deste trabalho. Como por exemplo, as classes de modelos de caminhos furtivos e tolerância a falhas de hardware não tiveram os insumos normalmente disponíveis para sua geração na aplicação da metodologia CoFI, na maioria dos serviços testados, dada a característica de escopo das funções escolhidas do sistema sob teste. No caso da InRob, o mapeamento de perfil operacional não foi necessário dado a priorização já feita na escolha dos serviços críticos implementados no protótipo AOCS.

Algumas diferenças entre as duas metodologias podem ser denotadas. Os testes produzidos com a aplicação da metodologia CoFI são do tipo caixa preta. Já a metodologia InRob cobre aspectos de interoperabilidade, portanto explicita as entidades interoperantes, controlador e equipamentos sensores e atuadores. A metodologia InRob permite uma modelagem de um componente em mais de uma máquina TIOA, o que pode ser visto na Figura 5.13. A metodologia InRob também demanda a utilização de um equipamento capaz de atuar no canal de comunicação para introduzir falhas (FEM) para a execução dos testes. Equipamentos com esta capacidade podem ser realizados como apresentado em (59) e (57).

### 5.6.1. Cobertura dos Mecanismos FDIR

Esta seção apresenta uma análise da cobertura dos requisitos elicitados, observando se o mecanismo FDIR utilizado para detectar a falha associada ao requisito é coberto pelos modelos. Com isso avalia-se a capacidade das metodologias de gerar testes para os requisitos, apoiando suas verificações e validações.

Inicialmente apresentamos o agrupamento dos modelos por serviço e os mecanismos FDIR cobertos por cada metodologia. A Tabela 5.20 apresenta a cobertura dos mecanismos FDIR pela metodologia CoFI.

Tabela 5.20. Serviços CoFI e Modelos vs. Mecanismos FDIR cobertos

| Serviços | Modelos CoFI | Mecanismo FDIR coberto |
|---|---|---|
| Serviço Mudança de Modo | S_MN_N | - |
| | S_MN_CF | - |
| Serviço Controle de Atitude | S_CA_N | - |
| | S_CA_EE | a, g, h, i, p, t |
| Serviço Manobra Orbital | S_MO_N | - |
| | S_MO_EE | c, g, h, i, p, t |
| Serviço Monitoramento de Saúde | S_MS_N | - |
| | S_MS_EE | f, j, n, o, q, r, s |
| | S_MS_FH | - |

A Tabela 5.21 apresenta a cobertura dos mecanismos FDIR pela metodologia InRob.

Tabela 5.21. Serviços InRob e Modelos vs. Mecanismos FDIR cobertos

| Serviços | Modelos InRob | Mecanismo FDIR coberto |
|---|---|---|
| Serviço Controle de Atitude | CN_CSA_1 | - |
| | CA_CSA_1 | o, r |
| | CN_CSDA_1 | - |
| | CA_CSDA_1 | o, p, q, r, t |
| | CN_CSDR_1 | - |
| | CA_CSDR_1 | o, p, q, r, t |
| | CN_CAAB_1 | - |
| | CA_CAAB_1 | j |
| | CN_CAD_1 | - |
| | CA_CAD_1 | f, g, h, i |
| | CN_CNTR_1 | - |
| | CN_MNOM_1 | - |
| | CA_NMON_1 | - |
| Serviço Manobra Orbital | CN_CSA_1 | - |
| | CA_CSA_1 | o, r |
| | CN_CSDA_1 | - |
| | CA_CSDA_1 | o, p, q, r, t |
| | CN_CSDR_1 | - |
| | CA_CSDR_1 | o, p, q, r, t |
| | CN_CAAB_1 | - |
| | CA_CAAB_1 | j |
| | CN_CAD_1 | - |
| | CA_CAD_1 | f, g, h, i |
| | CN_CNTR_1 | - |
| | CN_MMANO_1 | - |
| | CA_NMANO_1 | - |

A Tabela 5.22 apresenta um resumo dos requisitos elicitados e se existe modelo InRob ou CoFI para testá-lo.

Tabela 5.22. Cobertura de Requisitos por Modelos CoFI e/ou InRob

| Requisito | Mecanismo FDIR | Modelos InRob | Modelos CoFI |
|---|---|---|---|
| 1) O controlador não deve permitir uma ação de controle de manobra orbital ser executada fora do modo correto | a | - | X |
| 2) O controlador deve em um pior-cenário de requisições de uso de CPU da missão garantir o tempo de execução das tarefas de controle | b | NA | NA |
| 3) O controlador deve mesmo na ocorrência de falhas finalizar atuação sobre órbita antes de alterar o modo | c | - | X |
| 4) A lógica e sequência de ações na operação de uma manobra orbital deve garantir uma finalização segura da manobra mesmo no caso de falhas relevantes a mesma | d | NA | NA |
| 5) Os atuadores de força lineares devem inibir a atuação mesmo na ocorrência de falha simples no atuador | e | NA | NA |
| 6) O controlador deve detectar desvios de comportamento dos atuadores | f | X | X |
| 7) O controlador deve detectar erros no comando (comunicação) dos atuadores | g | X | X |
| 8) O controlador deve detectar erros nos tempos de comandos (comunicação) | h | X | X |
| 9) O controlador deve comandar um estado seguro do atuador caso este sinalize atuação em momento indevido | i | X | X |
| 10) O controlador deve detectar atuação espúria visto desvios da dinâmica esperada para a planta | j | X | X |
| 11) O controlador deve levar a um apontamento seguro caso detecte degradação de desempenho do controle | k | - | - |

Continua

Tabela 5.22 - Conclusão

| | | | |
|---|---|---|---|
| 12) O controlador deve detectar erros de hardware e software durante sua execução e chavear para modo de operação seguro em caso de falha. Obs. Erros de hardware: erro de memória, erro de processador. Erros de software: Divisões por zero, Valores numéricos fora de faixa (NaN, +-inf), erro na obtenção de recursos | l | - | - |
| 13) O Controlador deve levar a um apontamento seguro caso haja perda dos tempos de execução das tarefas | m | - | - |
| 14) O controlador deve levar a um apontamento seguro caso haja degradação de energia no satélite | n | - | X |
| 15) O controlador deve detectar leituras não coerentes de sensores. Obs. Descontinuidades e congelamento de valor | o | X | X |
| 16) O controlador deve detectar erros nos tempos dos dados dos sensores (comunicação) | p | X | X |
| 17) O controlador deve chavear para modo seguro após evoluir a dinâmica do sistema mesmo com sensor em falha por até X tempo | q | X | X |
| 18) O controlador deve entrar em modo seguro de apontamento se houver simultaneamente mais de uma falha em sensores | r | X | X |
| 19) O controlador deve entrar em modo seguro caso o comportamento da planta comece a divergir do esperado mesmo se não houver falhas detectadas de equipamentos | s | - | X |
| 20) O controlador deve detectar comunicação com dados corrompidos nos sensores digitais | t | X | X |

A Tabela 5.22 mostra que a metodologia CoFI gera modelos da maioria dos mecanismos relacionados aos requisitos. Esta informação é condizente com o esperado para uma metodologia que se posiciona para a fase de validação no processo de V&V onde todos os requisitos devem ser validados. É importante notar que, os modelos CoFI apresentam transições com informações em um

alto nível de abstração, portanto o próximo passo seria uma tradução das sequências de teste em casos de teste executáveis. Além disso, uma sequência de teste CoFI pode se desdobrar em vários casos de teste executáveis, considerando que uma transição pode representar diferentes estímulos.

A metodologia InRob trata de interoperabilidade de sistemas de tempo real e foca perigos de tempo, principalmente os relacionados à fase de integração de sistemas no processo de V&V. Os modelos InRob contém representações das interações entre os sistemas do SUT e detalham o comportamento das mensagens trocadas nos canais de comunicação como causa-efeito.

Alguns casos de teste gerados pela metodologia InRob com desvios de tempo não são encontrados diretamente equivalentes na metodologia CoFI, como por exemplo os casos de acúmulo de pequenos atrasos. A metodologia InRob tem a capacidade de trabalhar com contadores de tempo (*clocks*) síncrono entre as máquinas interoperantes, apoiando o teste de requisitos de tempo-real de forma bem estruturada. Portanto, é importante observar que embora ambas as metodologias testem alguns mesmos requisitos, os casos de testes gerados serão diferentes devido as diferenças e especificidades na modelagem do sistema sob teste (SUT) requeridos em cada metodologia.

# 6 CONCLUSÃO

Este trabalho de mestrado se propôs a avaliar: (i) a adequação das metodologias de teste em verificar e validar o comportamento esperado de mecanismos FDIR nas funções críticas realizadas pelo software de controle; e (ii) as comonalidades e complementariedades das metodologias no processo de teste de conformidade e interoperabilidade.

Importante lembrar, como dito no início do capítulo 2, que os trabalhos realizados nesta dissertação concentram-se na área de conhecimento de engenharia de software, contribuindo para verificação e validação de software espacial. Portanto, as conclusões se restringem ao escopo desta disciplina. Em um desenvolvimento completo de um subsistema de controle de atitude e órbita as atividades pesquisadas neste trabalho devem ser aliadas às atividades das outras disciplinas como engenharia de controle, hardware, elétrica, etc. para a conclusão do projeto de um sistema complexo como o de um AOCS.

Com base na  análise teórica, na execução do experimento prático, e nos resultados obtidos, pode-se concluir que as metodologias de teste baseado em modelos CoFI e InRob são aplicáveis e adequadas para formalizar e normatizar a atividade de testes para software de sistemas de controle de atitude e órbita.

O resultado do experimento teórico mostrou que as duas metodologias aplicadas de forma combinada cobrem todas as atividades de teste de verificação e validação de software, sendo elas: teste de unidade, teste de integração, teste de validação e testes de aceitação. As metodologias também cobrem todos os focos de teste de dependabilidade de software recomendados por guias de desenvolvimento espacial: teste de interface, teste de robustez, e teste de desempenho. Ficando parcialmente descoberto neste grupo somente testes de desempenho de software.

As metodologias têm propostas diferentes, e as comonalidades puderam ser percebidas em ambos os experimentos teóricos e práticos. Um foco especial em robustez é dado em ambas as metodologias, porém estas geram casos de

teste diferentes que podem se completar para verificar e validar um requisito como apresentado pela Tabela 4.3.

O experimento prático mostrou que as metodologias geram modelos representativos de mecanismo de FDIR de um sistema AOCS, sendo obtidos modelos para os serviços que cobrem os requisitos de *safety*, Tabela 5.22, elicitados durante o experimento. Estes modelos podem gerar casos de testes que testam os requisitos relacionados a estes mecanismos, apoiando assim a verificação e validação do sistema.

As especificidades na aplicação das metodologias em um sistema de controle de atitude e órbita foram denotadas durante a descrição do experimento, o que pode ser usado de guia para trabalhos em aplicações reais.

Os objetivos enunciados no início dos trabalhos puderam ser cumpridos.

Em resumo, o resultado obtido para o objetivo (ii) mostra que as metodologias, principalmente utilizadas de forma combinadas, se apresentam como uma proposta que cobre todas as fases da atividade de V&V de software e que são altamente complementares no teste de características ligadas a dependabilidade do software.

Em resumo, o resultado obtido para o objetivo (i) mostra que as metodologias se mostram adequadas ao teste do software de sistemas de controle de atitude e orbita, sendo capazes de gerar modelos de forma representativa dos mecanismos de FDIR de um sistema AOCS para gerar automaticamente testes, e assim apoiar a verificação e validação de seus requisitos.

Assim, o estudo realizado aponta forte contribuição das metodologias CoFI e InRob como guia para a construção de modelos comportamentais representativos de um sistema de controle de atitude e órbita, quando mecanismos de FDIR são devidamente considerados nos modelos. Nesta questão, o apoio da metodologia SPTA na elicitação dos requisitos de safety contribuiu significativamente para focar os aspectos críticos associados aos

serviços do software AOCS, destacando as propriedades relevantes para validação dos mecanismos FDIR.

## 6.1. Trabalhos Futuros

Uma vez que os modelos do protótipo AOCS, criados com base em ambas as metodologias, foram elaborados, o próximo passo é explorar aspectos da geração automática e seleção de casos de teste usando as ferramentas adequadas para cada metodologia. Outro trabalho a ser realizado é a execução destes testes contra o protótipo AOCS.

Uma oportunidade especial de trabalho é a possibilidade de integração das metodologias CoFI e InRob com outra metodologia de teste baseado em modelos que realize testes de desempenho de software, ou até mesmo investigar possíveis expansões das metodologias para cobrir outros aspectos de teste de dependabilidade, uma vez que testes de desempenho de software são cobertos de forma parcial pelas metodologias.

Outra oportunidade de trabalho futuro é comparar as metodologias de teste baseado em modelos com outros métodos de verificação de software, como métodos formais, para avaliar as capacidades de uso conjunto e potencial de encontrar defeitos de cada método durante um desenvolvimento.

O estudo do potencial da formalização e utilização de um processo combinado de especificação, desenvolvimento, verificação e validação utilizando as metodologias CoFI, InRob e STPA também se mostra como um ponto de trabalho futuro.

## REFERÊNCIAS BIBLIOGRÁFICAS

(1) LARSON, W. J.; WERTZ, J. R. **Space mission analysis and design.** Dordrecht, NL: Kluwer Academic, 1999. 969306936 ISBN 1-881883-10-8.

(2) WERTZ, J. R. **Spacecraft attitude determination and control**. Dordrecht, NL: D. Reidel, 1978. 858 p. (Astrophysics and Space Science Library, 73) ISBN 90-277-0959-9.

(3) Larson, W.; Kirkpatrick, D.; Sellers, J. J.; Thomas, D.; Verma, D. **Applied space systems engineering.** McGraw Hill, New York, 2009. (Space Technology Series) ISBN 978-0-073-40886-6.

(4) LEY, W.; WITTMANN, K.; HALLMANN, W. (Ed.). **Handbook of space technology.** Chichester, England : John Wiley, 2009. 882 ISBN 978-0-470-69739-9.

(5) EUROPEAN SPACE AGENCY (ESA). **ECSS-E-60A:** control engineering – space engineering. NL, 14 September 2004.

(6) EUROPEAN SPACE AGENCY (ESA). **ECSS-E-ST-60-10C:** control performance – space engineering. NL, 15 November 2008.

(7) TAFAZOLI, M. A study of on-orbit spacecraft failures. **Acta Astronautica**. v. 64, p. 195-205, 2009.

(8) LEVESON, N. G. Role of Software in Spacecraft Accidents, **Journal of Spacecraft and Rockets**, v. 41, n. 4, p. 564-575, 2004.

(9) NATIONAL AERONAUTICS AND SPACE ADMINISTRATION (NASA). **Genesis mishap investigation board report.** Volume I. Washington, 2005.

(10) LIONS, J. L. (Chairman). **Ariane 5:** Flight 501 failure - Report by the Inquiry Board. Paris: ESA-CNES, FR, 19 July 1996.

(11) STEPHENSON, A. G.; MULVILLE, D. R.; BAUER, F. H.; DUKEMAN, G. A.; NORVIG, P.; LAPIANA L. S.; RUTLEDGE P. J.; FOLTA D.; SACKHEIM, R. **Mars climate orbiter:** mishap investigation board report. Washington, DC: NASA, 10 November 1999.

(12) JPL SPECIAL REVIEW BOARD; CASANI, J. **Report on the loss of the Mars polar lander and deep space 2 missions.** Pasadena: NASA Jet Propulsion Laboratory, California Institute of Technology , 22 March 2000.

(13) YOUNG, T. (CHAIRMAN). **Mars program independent assessment team report.** Washington: NASA STI/Recon Technical Report N, 14 March 2000.

(14) ZOLGHADRI, A. Advanced model-based fdir techniques for aerospace systems: Today challenges and opportunities. **Progress in Aerospace Sciences,** v. 53, p. 18-29, 2012.

(15) OLIVE, X. FDI (R) for satellites: How to deal with high availability and robustness in the space domain?. **International Journal of Applied Mathematics and Computer Science**, v. 22, n. 1, p. 99–107, 2012.

(16) MATTIELLO-FRANCISCO, M. F.; MARTINS, E.; CAVALLI, A. R.; YANO, E. T. InRob: An approach for testing interoperability and robustness of real-time embedded software. **Journal of Systems and Software**, v. 85, n. 1, p. 3-15, Jan. 2012. doi: <10.1016/j.jss.2011.02.034>.

(17) MATTIELLO-FRANCISCO, M. F. **InRob -** uma abordagem para testes de interoperabilidade e de robustez de subsistemas de tempo-real intensivos em software. 2009. Tese de Doutorado - ITA, São José dos Campos - SP, 2009. Disponível em: <http://www.bd.bibl.ita.br/tesesdigitais/lista_resumo.php?num_tese=000556379>. Acesso em: 06 jan. 2014.

(18) AMBROSIO, A. M. **CoFI:** uma abordagem combinando teste de conformidade e injeção de falhas para validação de software em aplicações espaciais. 2005. 209 f. Tese (Doutorado em Computação Aplicada) - INPE, São José dos Campos, SP, 2005 São José dos Campos, SP : INPE, 2005. 209 (INPE-13264-TDI/1031) Disponível em: <http://urlib.net/rep/sid.inpe.br/MTC-m13@80/2005/09.06.13.34> Acesso em: 6 jan. 2014

(19) PINHEIRO, A. C. **Subsídios para a aplicação de métodos de geração de casos de testes baseados em máquinas de estados.** 2012. Dissertação (Mestrado em Ciências de Computação e Matemática Computacional) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2012.

(20) AMBROSIO, A. M.; MARTINS, E.; VIJAYKUMAR, N. L.; CARVALHO, S. V. Systematic generation of test and fault cases for space application validation. In: DATA SYSTEM IN AEROESPACE CONFERENCE, 9. (DASIA), Edinburg, Scotland. **Proceedings...** 2005. p. 12. Papel. (INPE-13072-PRE/8336). Disponível em: <http://urlib.net/sid.inpe.br/iris@1916/2005/11.21.18.43>. Acesso em: 06 jan. 2014.

(21) CHERUKURI, V. K., GUPTA, P. **Model based testing for non-functional requirements**. Master Thesis in Software Engineering School of Innovation, Design and Engineering Mälardalen University Västerås, Sweden. June, 2010.

(22) LEVESON, N. **Engineering a safer world:** systems thinking applied to safety. Cambridge, Massachusetts: MIT Press, 2011.

(23) FUNCAN, R. C.; GUNNERSEN JR, A. S. Inertial guidance, navigation and control systems. **Journal of Spacecraft and Rockets,** v. 1, n. 6, NASA Manned Spacecraft Center, Houston, TX, Nov-Dec 1964.

(24) EUROPEAN SPACE AGENCY (ESA). **ECSS-E-HB-60A:** control engineering handbook – space engineering. NL, 14 Decemeber 2010.

(25) AVIZIENIS, A.; LAPRIE, J. C.; RANDEL, B.; LANDWEHR, C. Basic concepts and taxonomy of dependable and secure computing. **IEEE Transactions on Dependable and Secure Computing**, v.1, n.1, p.11-33, 2004.

(26) NATIONAL AERONAUTICS AND SPACE ADMINISTRATION (NASA). **A guide to onboard checkout.** Volume 1: guidance, navigation and control. Huntsville: International Business Machines Corp., Sep. 1971. (NASA-CRE-115261).

(27) HUANG, H. M. **Autonomy levels for unmanned systems (ALFUS) framework volume i:** terminology, version 2.0. Gaithersburg, MD: NIST Special Publication 1011-I-2.0, pages 1–47, 2008. Contributed by the Ad Hoc ALFUS Working Group Participants.

(28) DEPARTMENT OF THE NAVY - USA. **The Navy Unmanned Surface Vehicle (USV) master plan**. 23 July 2007. Disponível em http://www.navy.mil/navydata/technology/usvmppr.pdf, acesso em 23/01/2014.

(29) KENDOUL, F. Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. **Journal of Field Robotics**, v. 29, n. 2, p. 315–378, 2012.

(30) PASETTI, A. **Software frameworks and embedded control systems.** Berlin, Germany: Springer, 2002. 293 p. (Lecture Notes in Computer Science 2231) ISBN 0-540-43189-6.

(31) NATIONAL AERONAUTICS AND SPACE ADMINISTRATION (NASA). **NSTS 1988 news reference manual –** guidance, navigation and control. September, 1988. Disponível em: http://science.ksc.nasa.gov/shuttle/technology/sts-newsref/stsref-toc.html, acesso em 23/01/2014.

(32) KOEBEL, F.; COLDEFY, J. F. SCOC3: a space computer on a chip: an example of successful development of a highly integrated innovative ASIC. In: CONFERENCE ON DESIGN, AUTOMATION AND TEST IN EUROPE (DATE), 2010, Dresden. **Proceedings…** Dresden: European Design and Automation Association, 2010. p. 1345-1348.

(33) EUROPEAN SPACE AGENCY (ESA). **ECSS-E-70-41A:** ground systems and operations – telemetry and telecommand packet utilization. NL, 30 January 2003.

(34) EICKHOFF, J. **Simulating spacecraft systems.** Heidelberg, Germany : Springer, 2009. 353 p. (Springer Aerospace Technology) ISBN 978-3-642-01275-4.

(35) EICKHOFF, J. **Onboard computers, onboard software and satellite operations**. Springer Science & Business Media, 2011, 300 p. ISBN 9783642251702.

(36) ISERMANN, R.; BALLE, P. Trends in the application of model-based fault detection and diagnosis of technical processes. **Control Engineering Practice**, n. 5, p. 709-719, 1997.

(37) SALLEM, F. **Détection et isolation de défauts actionneurs basées sur un modèle de l'organe de commande.** PhD diss., Université Paul Sabatier-Toulouse III, 2013

(38) LEITE, A. C. **Detecção e diagnóstico de falhas em sensores e atuadores da plataforma multi-missão**. 2007. 374 p. (INPE-15219-TDI/1313). Tese (Doutorado em Mecânica Espacial e Controle) - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2007. Disponível em: <http://urlib.net/sid.inpe.br/mtc-m17@80/2007/12.07.10.48>. Acesso em: 06 jan. 2014.

(39) WHEELER, T. J. **Probabilistic performance analysis of fault diagnosis schemes.** PhD diss., University of California, Berkeley, 2011

(40) BAYOUDH, M.; TRAVÉ-MASSUYES, L.; OLIVE, X.; THALES ALENIA SPACE. Hybrid systems diagnosis by coupling continuous and discrete event techniques. In: IFAC WORLD CONGRESS, 2008, Seoul, Korea. **Proceedings…** Seoul: IFAC, 2008. p. 7265-7270.

(41) VENKATASUBRAMANIAN, V.; RENGASWAMY, R.; YIN, K.; KAVURI, S. N. A review of process fault detection and diagnosis: Part I: Quantitative model-based methods. **Computers & chemical engineering 27**, n. 3, p. 293-311, 2003.

(42) VENKATASUBRAMANIAN, V.; RENGASWAMY, R.; YIN, K.; KAVURI, S. N. A review of process fault detection and diagnosis: Part II: Qualitative models and search strategies. **Computers & Chemical Engineering 27**, n. 3, , p. 313-326, 2003.

(43) VENKATASUBRAMANIAN, V.; RENGASWAMY, R.; YIN, K.; KAVURI, S. N. A review of process fault detection and diagnosis: Part III: Process history based methods. **Computers & chemical engineering 27**, n. 3, p. 327-346, 2003.

(44) ____. **An STPA primer**, Version 1. MIT, August 2013. Disponível em: http://sunnyday.mit.edu/STPA-Primer-v0.pdf. Acesso em: 06 jan. 2014.

(45) EUROPEAN SPACE AGENCY (ESA). **ECSS-E-HB-60-10A:** control performance guidelines. NL, 14 December 2010.

(46) EUROPEAN SPACE AGENCY (ESA). **ECSS-E-ST-40C:** software. NL, 6 March 2009.

(47) PRESSMAN, R. S. **Software engineering, a practitioner's approach.** Boston, MA: McGraw-Hill, 2009. 895 p. ISBN 978-0-073-37597-7.

(48) EUROPEAN SPACE AGENCY (ESA). **ECSS-E-HB-40A PR-Draft 1:** software engineering handbook. NL, 1 October 2012.

(49) AMBROSIO, A. M.; MARTINS, E.; VIJAYKUMAR, N. L.; CARVALHO, S. V. A conformance testing process for space applications software services. **Journal of Aerospace Computing, Information, and Communication**, v. 3, n. 4, p. 146-158, Apr. 2006. (INPE-14067-PRE/9236).

(50) MORAIS, M. H. E.; AMBRÓSIO, A. M. A new model-based approach for analysis and refinement of requirement specification to space operations. In: SPACEOPS 2010 CONFERENCE, 2010, Huntsville. **Proceedings...** Hunstsville: AIAA, 2010. Papel.

(51) ANJOS, J. M. S.; GRIPP, J.; PONTES, R. P.; VILLANI, E. Applying the CoFI testing methodology to a multifunctional robot end-effector. In: LATIN-AMERICAN SYMPOSIUM ON DEPENDABLE COMPUTING (LADC), 5., 2011, São José dos Campos (Brasil). **Proceedings…** São José dos Campos: INPE, 2011. v. Suplemental. CD, On-line. Disponível em: <http://urlib.net/8JMKD3MGP8W/39FG232>. Acesso em: 07 ago. 2014.

(52) FRANCISCO, M. F. M.; VILLANI, E.; MARTINS, E.; DUTRA, T.; COELHO, B.; AMBROSIO, A. M. An Experience on Technology Transfer of CoFI Methodology to Automotive Domain. In: LATIN-AMERICAN SYMPOSIUM ON DEPENDABLE COMPUTING, 6. (LADC'2013), 2013, Rio de Janeiro. **Proceedings...** 2013. DVD. ISBN 978-85-7669-274-4.

(53) EUROPEAN SPACE AGENCY (ESA). **ECSS-E-40 Part 1B:** software – part1: principles and requirements. NL, 28 November 2003.

(54) MARTINS, E. **ATIFS:** um ambiente de teste baseado em injeção de falhas de software. Campinas: UNICAMP, 1995. 32p. (DCC-95-24).

(55) CAVALLI, A.; LEE, D.; RINDERKNECHT, C.; ZAÏDI, F. Hit-or-Jump: An algorithm for embedded testing with applications to IN services. **In Formal Methods for Protocol Engineering And Distributed Systems IFIP Advances in Information and Communication Technology,** v. 28, p. 41-56. Springer US, 1999.

(56) PINHEIRO, A. C.; AMBROSIO, A. M. **Modelagem do módulo de comunicação do satélite ITASAT segundo a metodologia COFI**. São José dos Campos: INPE, 2013. 40 p. (sid.inpe.br/mtc-m19/2013/10.04.19.58-NTC). Disponível em: <http://urlib.net/8JMKD3MGP7W/3EUF338>. Acesso em: 06 jan. 2014.

(57) CORSETTI, A. FPGA-based fault injection architecture for real time software dependability testing. In: LATIN-AMERICAN SYMPOSIUM ON DEPENDABLE COMPUTING, 5. (LADC), 2011, São José dos Campos. **Proceedings...** São José dos Campos: INPE, 2011. v. Suplemental. CD, On-line. Disponível em: <http://urlib.net/8JMKD3MGP8W/39FGQSP>. Acesso em: 06 ago. 2014.

(58) SABIÃO, S. B.; MARTINS, E. Condado: uma ferramenta para geração de testes de protocolos combinando controle e dados. In: SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES, 16., 1998, Rio de Janeiro: UFF, 1998. p. 404-423.

(59) MATTIELLO-FRANCISCO, F.; MARTINS, E.; CORSETTI, A.; CAVALLI, A. R.; YANO, E. Extended interoperability models for timed system robustness testing. In: COMMUNICATIONS, 2009. LATINCOM'09. IEEE LATIN-AMERICAN CONFERENCE, 2009, Medellin, Colombia. **Proceedings...** Medellin: IEEE, 2009. p. 1-6.

## APÊNDICE A – ARTEFATOS DA METODOLOGIA INROB

Este apêndice apresenta os artefatos criados na aplicação da metodologia InRob ao Protótipo AOCS.

Os modelos InRob apresentados no apêndice são os modelos nominais e os modelos aumentados com os perigos de tempo.

- **Modelo Sensores Analógicos**

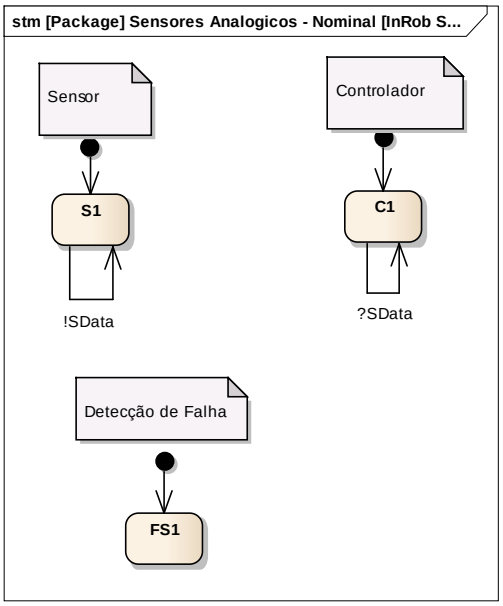


Figura A.1. Modelo InRob Nominal CN_CSA_1: Controlador e Sensores Analógicos

Tabela A.1. Desvios de tempo Modelo InRob Sensores Analógicos

| Classe do desvio | Desvio |
|---|---|
| Maior (majD) | Atrasos que disparam uma falha da dinâmica do sensor (abrupta) |
| Menor (minD) | Atrasos que com acúmulo disparam uma falha da dinâmica do satélite (suave) |

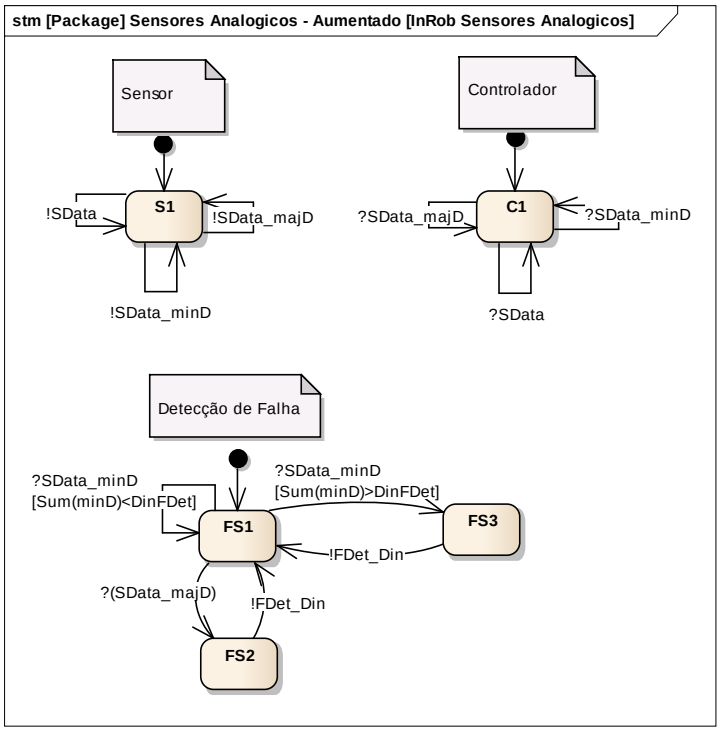


Figura A.2. Modelo InRob Aumentado CA_CSA_1: Controlador e Sensores Analógicos

O modelo apresenta a comunicação entre sensores analógicos e o computador de bordo. Em um sensor analógico não existe a falta de dado, pois qualquer valor representa um dado válido. Porém um mau funcionamento do sensor pode atrasar a propagação das medidas gerando falhas da dinâmica do sensor, se abrupta, ou da planta se suave.

- **Modelo Sensores Digitais Automáticos**

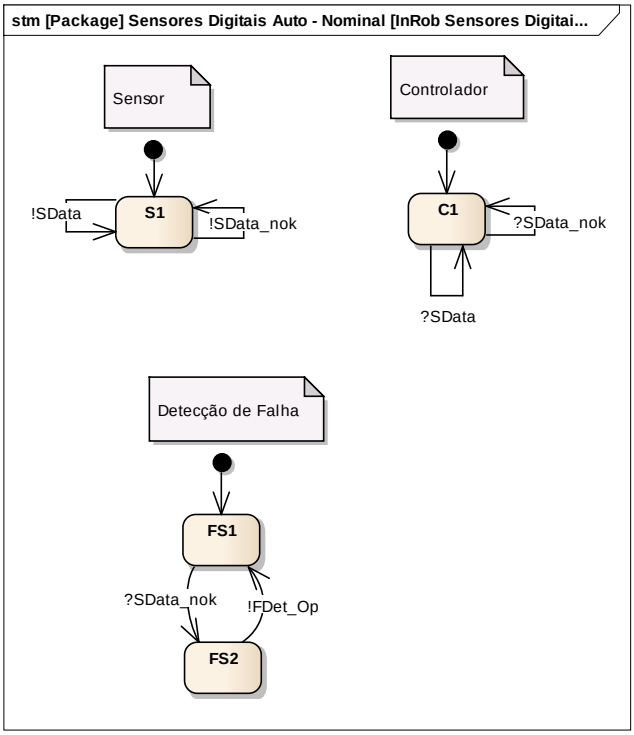


Figura A.3. Modelo InRob Nominal CN_CSDA_1: Controlador e Sensores Digitais Automáticos

Tabela A.2. Desvios de tempo Modelo InRob Sensores Digitais Automáticos

| Classe do desvio | Desvio |
|---|---|
| Maior (majD) | Atrasos que extrapolam a janela de aceite do dado do sensor dentro de um ciclo do controle |
| Menor (minD) | Atrasos que não extrapolam a janela de aceite do dado do sensor dentro de um ciclo do controle |
| Perda (TO) | Perda da mensagem, considerada um atraso infinito |

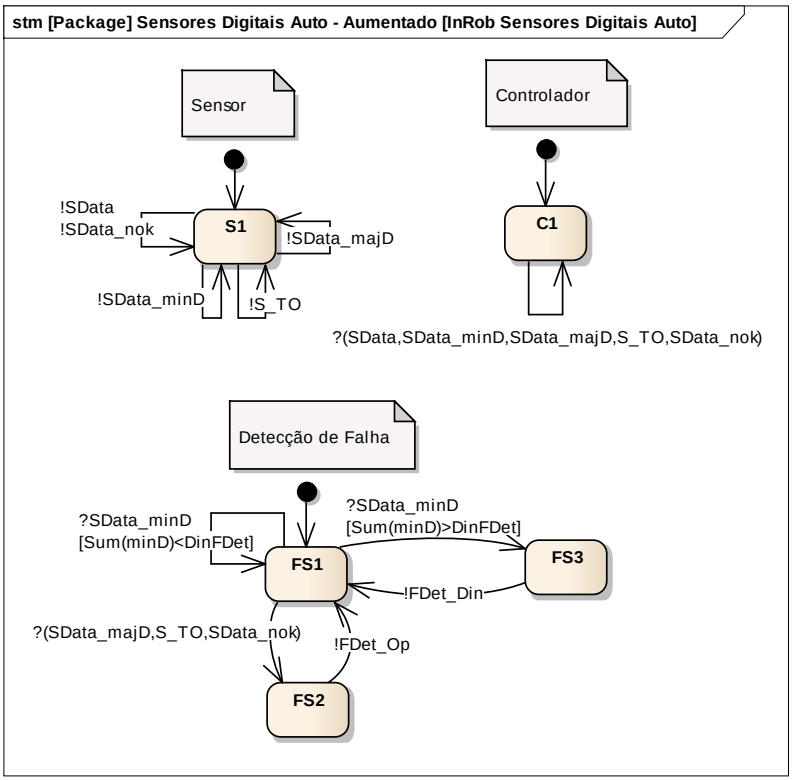


Figura A.4. Modelo InRob Aumentado CA_CSDA_1: Controlador e Sensores Digitais Automáticos

O modelo apresenta a comunicação entre um sensor digital de transmissão de dados automaticamente e o computador de bordo, isto quer dizer que o sensor operando automaticamente envia as medidas em um tempo definido. Neste caso espera-se uma nova medida dentro de um período após a medida anterior, e a perda deste tempo gera uma falha operacional. Pequenos atrasos nas medidas dos sensores podem ultimamente disparar uma falha na dinâmica do satélite, pois a medida divergirá da esperada.

- **Modelo Sensores Digitais por Requisição**

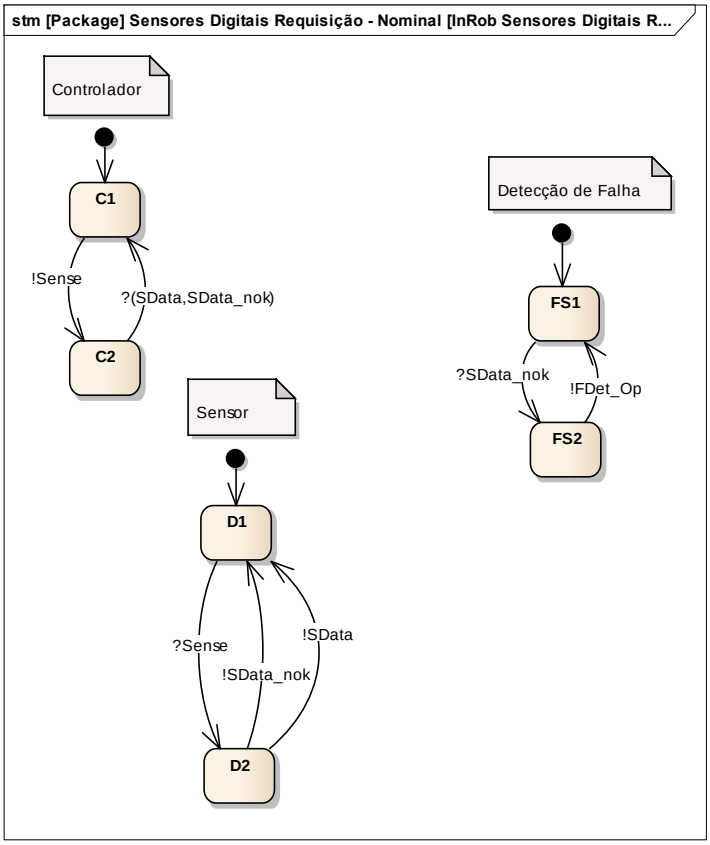


Figura A.5. Modelo InRob Nominal CN_CSDR_1: Controlador e Sensores Digitais por Requisição

Tabela A.3. Desvios de tempo Modelo InRob Sensores Digitais por Requisição

| **Classe do desvio** | **Desvio** |
|---|---|
| Maior (majD) | Atrasos que extrapolam a janela de resposta do sensor |
| Menor (minD) | Atrasos que não extrapolam a janela de resposta do sensor |
| Perda (TO) | Perda da mensagem, considerada um atraso infinito |

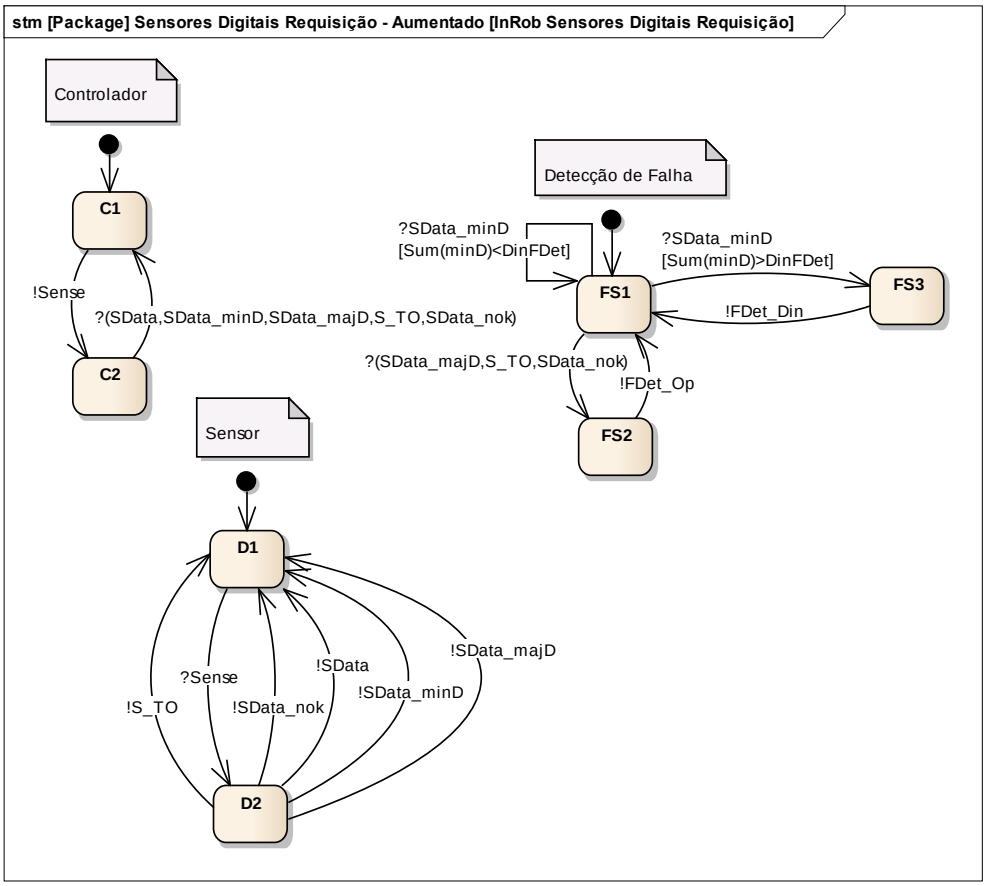


Figura A.6. Modelo InRob Aumentado CA_CSDR_1: Controlador e Sensores Digitais por Requisição

O modelo apresenta a comunicação entre um sensor digital por requisição e o computador de bordo. Neste caso, como há uma requisição de medida e uma janela de tempo esperada para resposta, a não satisfação ou um desvio de tempo grande causam uma falha operacional. Pequenos atrasos nas medidas dos sensores podem ultimamente disparar uma falha na dinâmica do satélite, pois a medida divergirá da esperada.

- **Atuadores Analógicos e Bilevel**

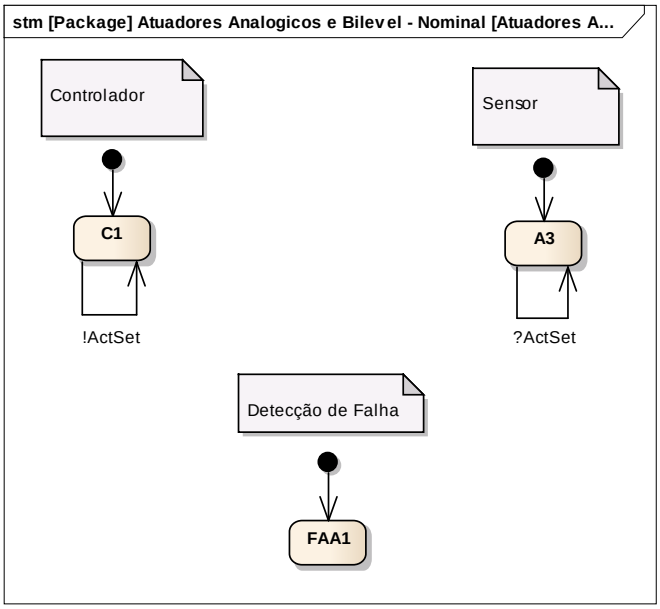


Figura A.7. Modelo InRob Nominal CN_CAAB_1: Controlador e Atuadores Analógicos e Bilevel

Tabela A.4. Desvios de tempo Modelo InRob Atuadores Analógicos e Bilevel

| Classe do desvio | Desvio |
|---|---|
| Maior (majD) | Atrasos que disparam uma falha da dinâmica do satélite (abrupta) |
| Menor (minD) | Atrasos que com acúmulo disparam uma falha da dinâmica do satélite (suave) |

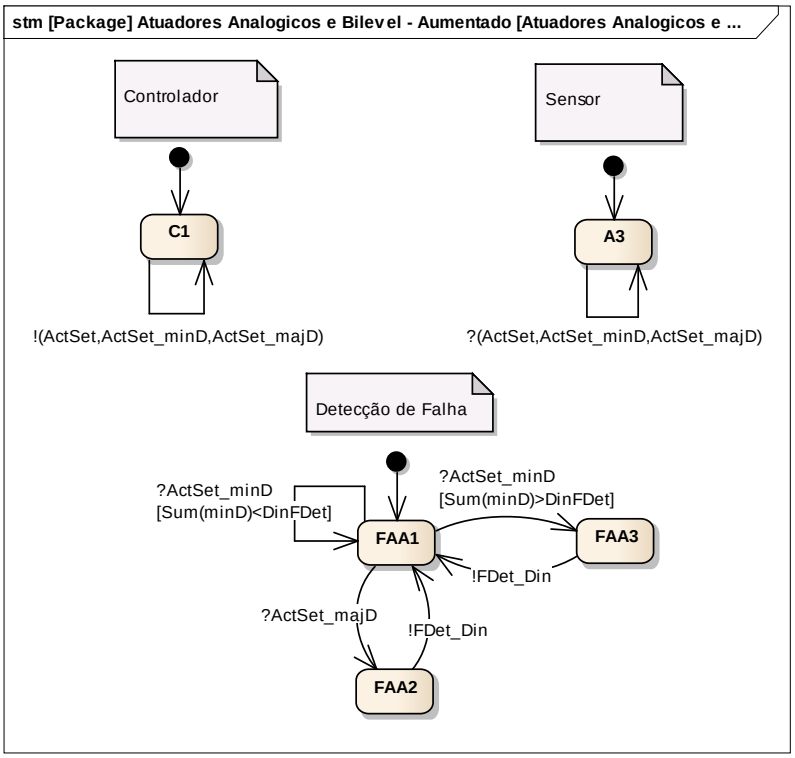


Figura A.8. Modelo InRob Aumentado CA_CAAB_1: Controlador e Atuadores Analógicos e Bilevel

O modelo apresenta o comando de atuação em atuadores analógicos ou bilevels, onde o comando não tem retorno, portanto não tem *time-out*. Um atraso na recepção destes comandos causam falhas da dinâmica esperada do satélite.

- **Atuadores Digitais**

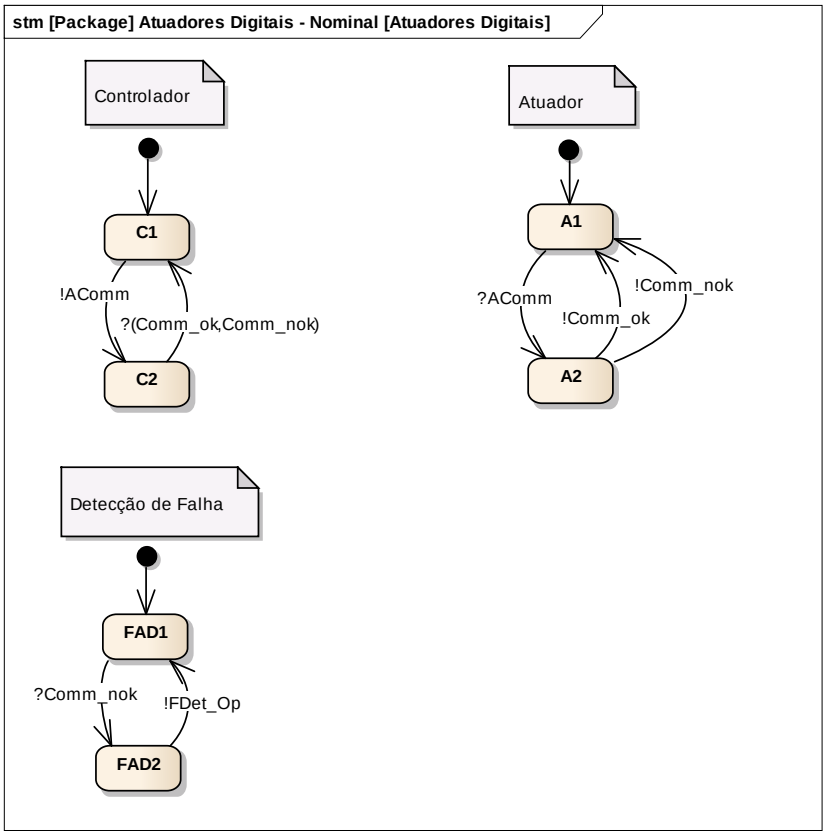


Figura A.9. Modelo InRob Nominal CN_CAD_1: Controlador e Atuadores Digitais

Tabela A.5. Desvios de tempo Modelo InRob Atuadores Digitais

| **Classe do desvio** | **Desvio** |
|---|---|
| Maior (majD) | Atrasos que a extrapolam a janela de aceite do comando de atuação |
| Menor (minD) | Atrasos que não extrapolam a janela de aceite do comando de atuação |
| Perda (TO) | Perda da mensagem, considerada um atraso infinito |

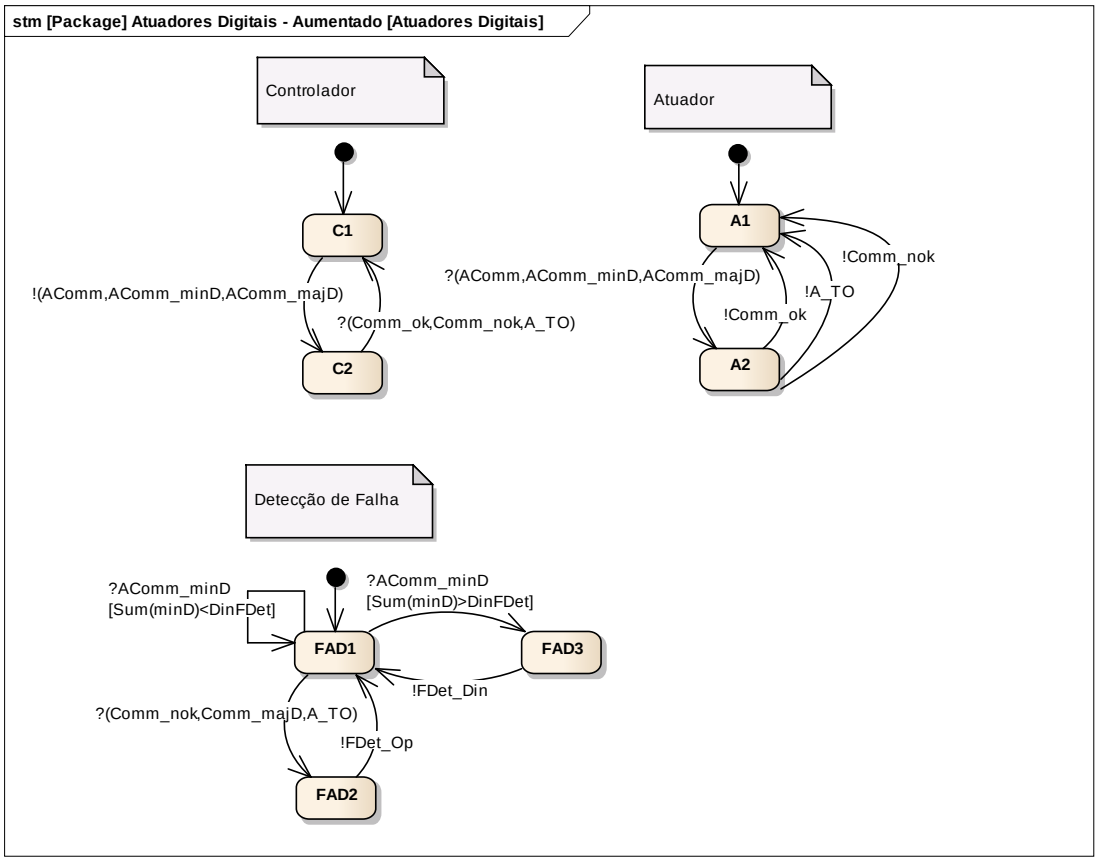


Figura A.10. Modelo InRob Aumentado CA_CAD_1: Controlador e Atuadores Digitais

O modelo apresenta o comando de atuação em atuadores digitais, onde há o comando e a confirmação do comando recebido pelo atuador. No caso de desvios que não extrapolam a janela nada ocorre inicialmente, porém o acúmulo de pequenos desvios de tempo na atuação pode influenciar ultimamente a dinâmica do atuador comparada com a esperada pelo controlador, disparando uma falha da dinâmica do equipamento.

- **Controlador**

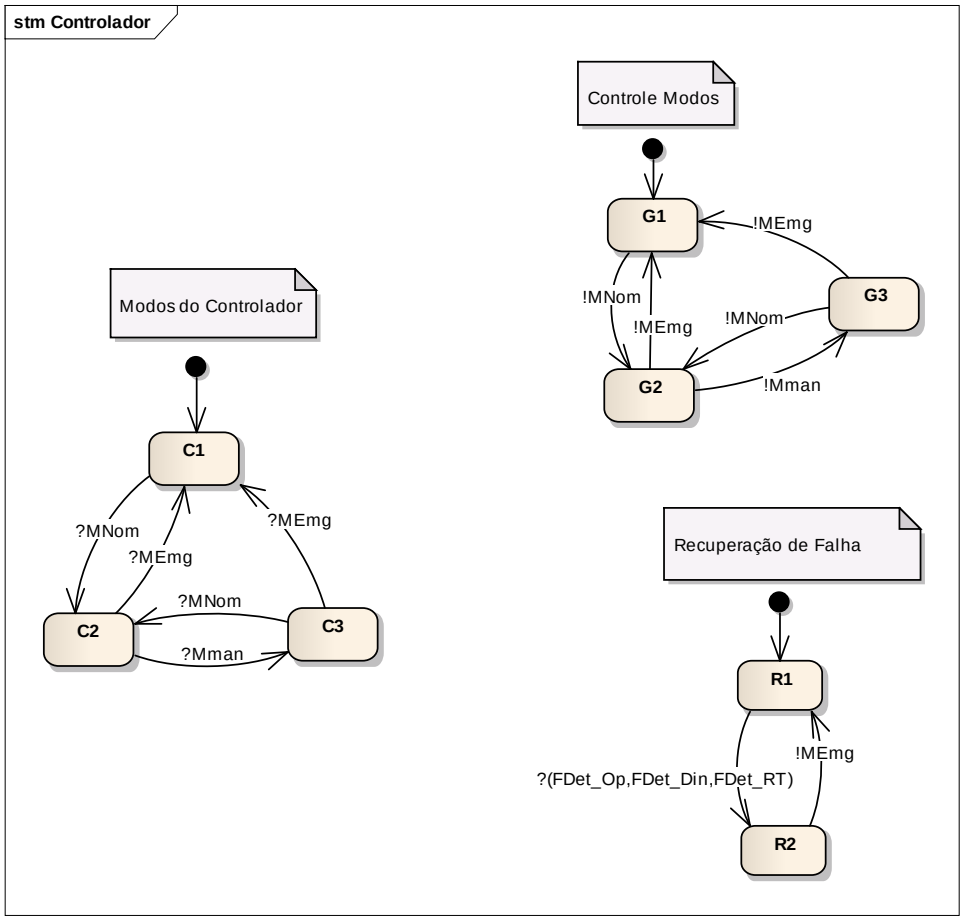


Figura A.11. Modelo InRob Nominal CN_CNTR_1: Controlador

Este modelo não apresenta desvios de tempo, pois engloba funções do controlador apenas, o modelo representa as máquinas TIOA complementares às apresentadas, com a porção de recuperação da falha do Controlador, centralizada como proposto no Protótipo de AOCS desenvolvido. Estas máquinas são disparadas por eventos nos demais modelos e representam o restante das funções de FDIR e operação do controlador.

- **Modo Nominal**

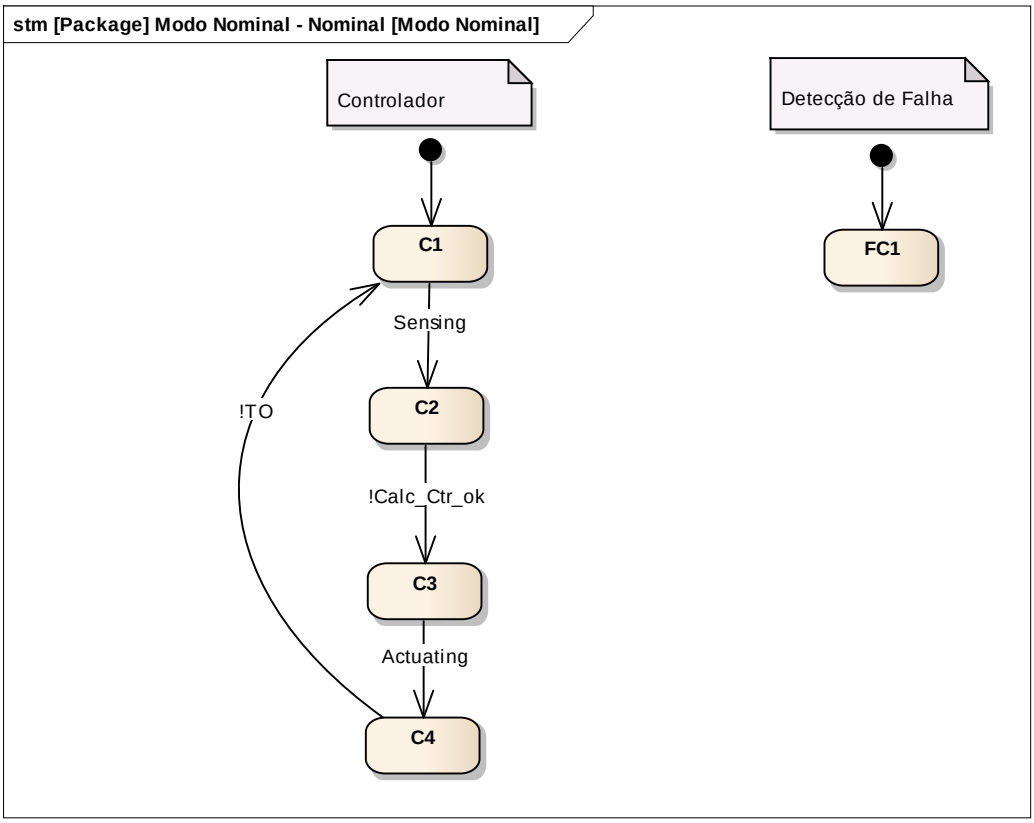


Figura A.12. Modelo InRob Nominal CN_MNOM_1: Modo Nominal

Tabela A.6. Desvios de tempo Modelo InRob Modo Nominal

| Classe do desvio | Desvio |
|---|---|
| Maior (majD) | Atrasos que extrapolam a janela de tempo de computação da tarefa de controle |
| Menor (minD) | Atrasos que com acúmulo disparam uma falha de tempo de computação da tarefa de controle |

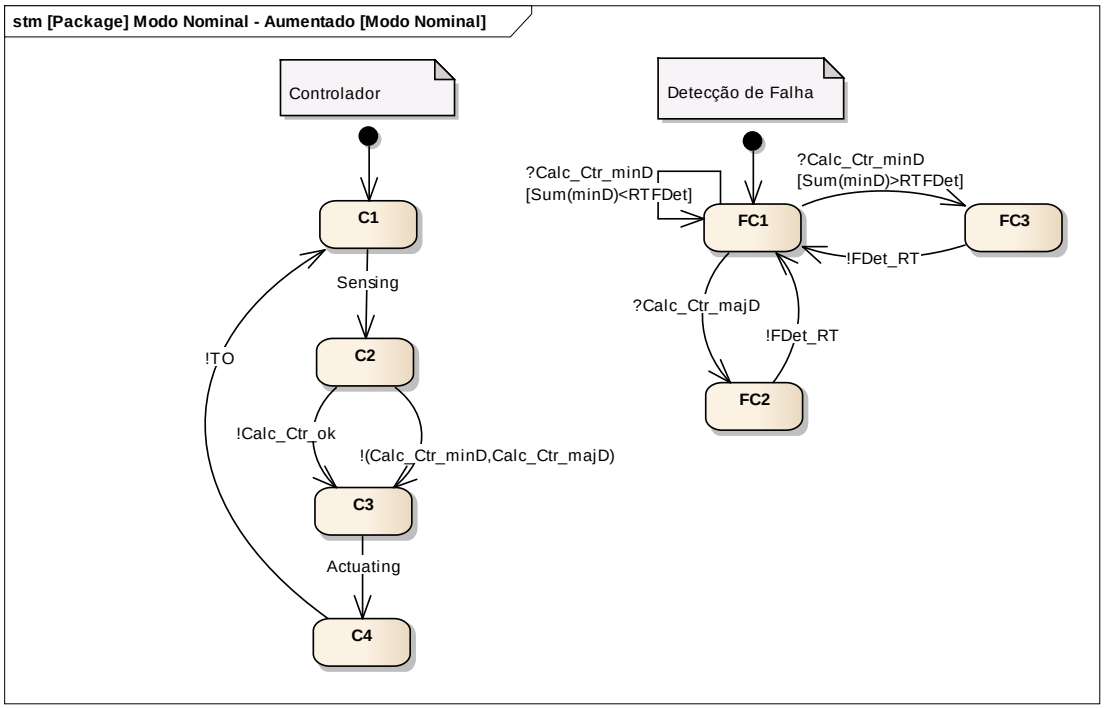


Figura A.13. Modelo InRob Aumentado CA_MNOM_1: Modo Nominal

O modelo apresenta o modo nominal de operação do controle, contendo a parte central do serviço de controle de atitude. Neste modelo podemos ver que atrasos na execução de computações podem gerar falhas de software relacionadas ao tempo real da execução. Neste modelo também vemos o encapsulamento e hierarquização da execução das máquinas dos sensores e dos atuadores, facilitando o trabalho e entendimento via modularização. As transições “Sensing” e “Actuating” representam as execuções das máquinas de Sensores e Atuadores respectivamente. O modelo apresenta uma característica de tempo real interna ao software do controlador, que precisaria de um mecanismo de falha como discutido na seção 5.5., porém é apresentado neste contexto para ilustrar a capacidade de modelagem das características em uma integração software-software.

- **Modo Manobra Orbita**

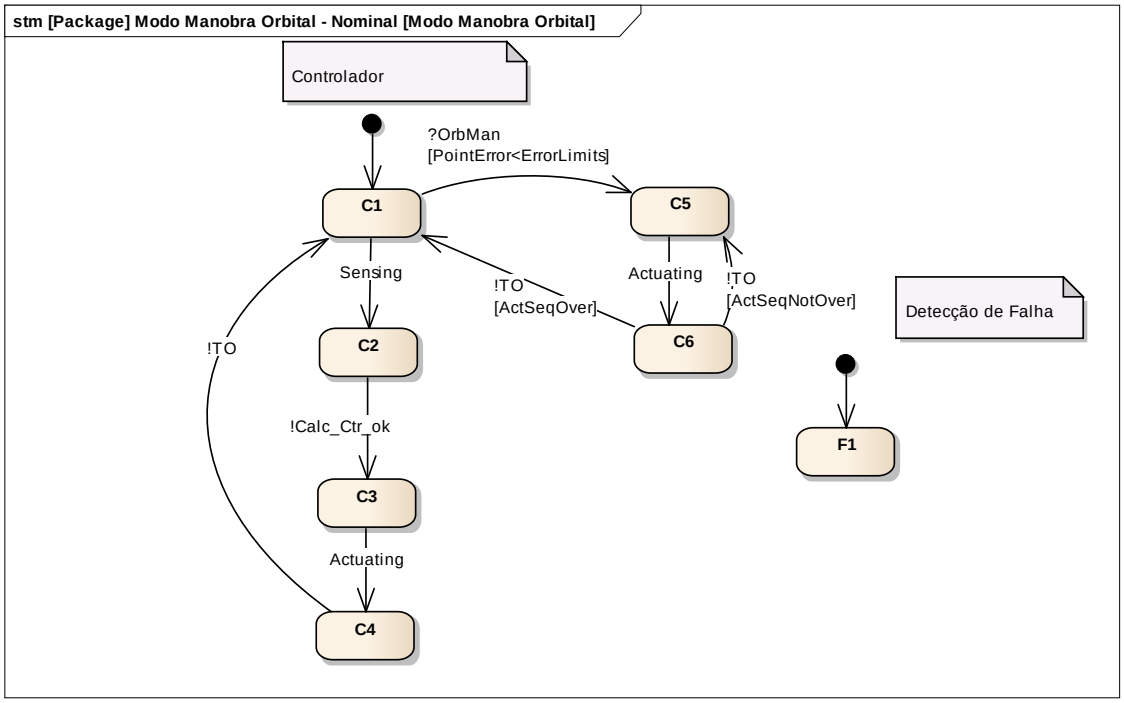


Figura A.14. Modelo InRob Nominal CN_MMANO_1: Modo Manobra Orbital

Tabela A.7. Desvios de tempo Modelo InRob Modo Manobra Orbital

| Classe do desvio | Desvio |
|---|---|
| Maior (majD) | Atrasos que extrapolam a janela de tempo de computação da tarefa de controle |
| Menor (minD) | Atrasos que com acúmulo disparam uma falha de tempo de computação da tarefa de controle |

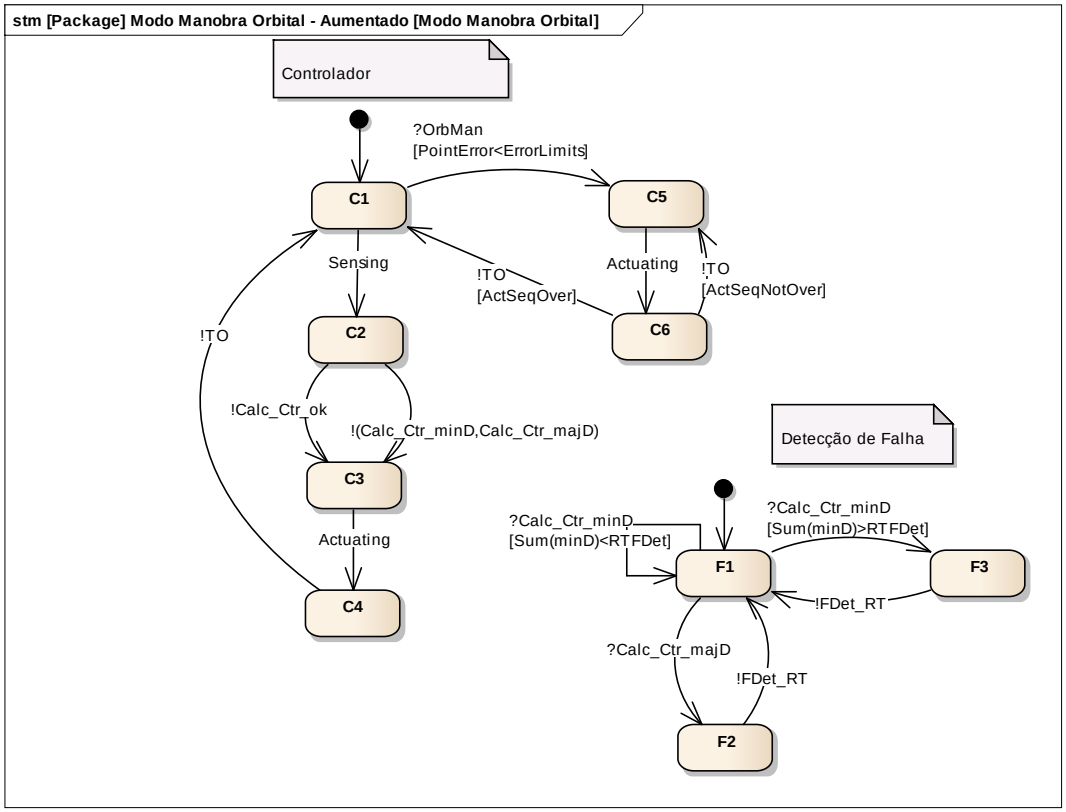


Figura A.15. Modelo InRob Aumentado CA_MMANO_1: Modo Manobra Orbital

O modelo apresenta o modo de manobra orbital, contendo a parte central do serviço de mesmo nome. Podemos ver a reutilização de modelagem possível com a modularização das máquinas de sensores e atuadores. Neste ponto como a atuação de uma manobra orbital ocorre em malha aberta não existem mecanismos de monitoramento do estado dependentes de tempo durante a sequência de ações da manobra. O modelo apresenta uma característica de tempo real interna ao software do controlador, que precisaria de um mecanismo de falha como discutido na seção 5.5., porém é apresentado neste contexto para ilustrar a capacidade de modelagem das características em uma integração software-software.

## APÊNDICE B – ARTEFATOS DA METODOLOGIA COFI

Este apêndice apresenta os modelos da metodologia CoFI.

Inicialmente são apresentadas as tabelas completas de Entradas, Saídas, Falhas de Hardware e Exceções Especificadas. E em seguida são apresentados os serviços.

Tabela B.1. Tabela completa de Entradas

| PC | Evento de entrada | Descrição |
|---|---|---|
| PC-01 | Requer_Modo_Nominal | Requisição externa para mudança para modo de operação Nominal |
| | Requer_Modo_ManOrb | Requisição externa para mudança para modo de operação Manobra Orbital |
| | Requer_Modo_ Emergência | Requisição externa para mudança para modo de operação Emergência |
| | Requer_Execução_de_ Manobra | Requisição para executar sequência de atuação de Manobra Orbital |
| PC-02 | Injeta_Falha_Dinamica _Sensor_N | Insere falha de dinâmica em um sensor (Descontinuidade ou congelamento de dado) |
| | Injeta_Falha_Dados_ Sensor_N | Insere falha nos dados de um sensor por X ciclos |
| | Injeta_Falha_Perda_de _Dado_Sensor_N | Insere falha de perda de dado de um sensor no ciclo |
| | Injeta_Falha_Sensores _N_M | Insere falha em dois sensores em um mesmo ciclo |
| | Injeta_Falha_Dado_ Corrompido_Sensor_N | Insere falha de dado corrompido de sensor |
| | Injeta_Falha_Dinamica _Planta | Insere falha de dinâmica da planta |
| | Injeta_Falha_Energia_ Planta | Comanda sinalização de baixa energia do satélite |
| | Injeta_Falha_Atuador_ Verboso | Insere falha de atuador se comunicar em momento incorreto |
| | Injeta_Falha_Perda_ Comando_Atuador_N | Insere falha de perda de resposta a comando a um atuador |
| | Injeta_Falha_Comando _Corrompido_Atuador_ N | Insere falha de dado corrompido no comando a um atuador |
| | Injeta_Falha_Dinamica _Atuador_N | Insere falha da dinâmica de um atuador |
| | Início_de_ciclo | Início de ciclo de controle, gerado por temporizador interno do sistema |

Continua

Tabela B.1. - Conclusão

| | | |
|---|---|---|
| Interno | Fim_monitora_ sensores_ok | Finalização do monitoramento de sensores sem encontrar falhas |
| | Fim_monitora_ sensores_nok | Finalização do monitoramento de sensores com a descoberta de falha |
| | Fim_monitora_planta_ ok | Finalização do monitoramento da planta sem encontrar falhas |
| | Fim_monitora_planta_ nok | Finalização do monitoramento da planta com a descoberta de falha |
| | Fim_monitora_ atuadores_ok | Finalização do monitoramento de atuadores sem encontrar falhas |
| | Fim_monitora_ atuadores_nok | Finalização do monitoramento de atuadores com a descoberta de falha |
| | Fim_monitora_energia_ ok | Finalização do monitoramento da energia do satélite sem encontrar falhas |
| | Fim_monitora_energia_ nok | Finalização do monitoramento da energia do satélite com a descoberta de falha |
| | Fim_sensoriamento_ok | Finalização do sensoriamento sem falha |
| | Fim_sensoriamento_ nok | Finalização do sensoriamento com falha |
| | Fim_calccontrole_ok | Finalização da computação dos comandos de controle sem falha |
| | Fim_calccontrole_nok | Finalização da computação dos comandos de controle com falha |
| | Fim_atuação_ok | Finalização da atuação sem falha |
| | Fim_atuação_nok | Finalização da atuação com falha |
| | Executa_manobra_ok | Execução de comando de manobra orbital |
| | Executa_manobra_nok | Tentativa com falha de execução de comando de manobra orbital |
| | Executa_nenhuma_ manobra | Finalização do ciclo de controle do modo sem comandos de manobra orbital a executar |
| | Fim_atuação_manobra _ok | Finalização de atuação de manobra orbital sem falha |
| | Fim_atuação_manobra _nok | Finalização de atuação de manobra orbital com falha |

Tabela B.2. Tabela completa de Falhas de Hardware

| Falha | Evento | Descrição |
|---|---|---|
| Falha de Hardware OBC | Injeta_Falha_ HardwareOBC | Falhas no Hardware do Computador de Bordo sinalizadas, como bitflip em memória e erro de CPU. |
| Falha de Sensor | * | As falhas de sensores são tratadas como exceções especificadas pois a própria existência do sistema de FDIR é voltada a lidar as diferentes falhas dos equipamentos comunicantes |
| Falha de Atuador | * | As falhas de atuadores são tratadas como exceções especificadas pois a própria existência do sistema de FDIR é voltada a lidar as diferentes falhas dos equipamentos comunicantes |

Tabela B.3. Tabela completa de Saídas

| PO | Saídas | Descrição |
|---|---|---|
| | Muda_para_modo_ Nom | Indicação por TM que o modo alterou para Nominal |
| | Muda_para_modo_ ManOrb | Indicação por TM que o modo alterou para Manobra Orbital |
| | Muda_para_modo_ Emg | Indicação por TM que o modo alterou para Emergência |
| | Sinaliza_Mudança_não _permitida | Indicação por TM que tentativa de mudança de modo sem sucesso |
| | Requere_Modo_ Emergência | Indicação por TM de requisição interna para mudança para modo de operação emergência |
| | Sinaliza_Falha_Din_ Sensor | Sinalização por TM de falha na dinâmica de um sensor |
| | Sinaliza_Falha_Dado_ Defasado | Sinalização por TM de falha por dado de sensor muito defasado (mais de x ciclos) |
| | Sinaliza_Degradação_ Desempenho_Controle | Sinalização por TM de falha de parâmetro de desempenho de controle fora da faixa nominal |
| | Sinaliza_Falha_na_ Dinamica_do_Satélite | Sinalização por TM de falha na dinâmica do satélite |

Continua

Tabela B.3. - Conclusão

| | | |
|---|---|---|
| PO-01 | Sinaliza_Falha_ SoftwareRT | Sinalização por TM de falha interna ao software básico do AOCS |
| | Sinaliza_Falha_na_ Dinamica_do_Atuador | Sinalização por TM de falha na dinâmica de um atuador |
| | Sinaliza_Degradação_ Nível_Energia | Sinalização por TM de falha por nível de energia do satélite abaixo do nominal |
| | Sinaliza_Falha_ Simultanea_de_ Sensores | Sinalização por TM de falha simultânea em mais de um sensor |
| | Sinaliza_Falha_ HardwareOBC | Sinalização por TM de falha interna ao hardware do OBC |
| | Sinaliza_Dado_ Corrompido_Sensor | Sinalização por TM de falha por dado de sensor estar corrompido |
| | Sinaliza_Falha_ CalcControle | Sinalização por TM de falha na computação dos comandos de controle |
| | Sinaliza_RespExp_ Atuador | Sinalização por TM de falha por recepção de resposta espúria de um atuador |
| | Sinaliza_Nack_Atuador | Sinalização por TM de falha recusa de comando de controle por parte do atuador |
| | Sinaliza_Sensor_TO | Sinalização por TM de falha de recepção de dado de sensor |
| | Sinaliza_Atuador_TO | Sinalização por TM de falha de recepção de reposta de comando a atuador |
| | Manobra_Falha_ Finalizada | Sinalização por TM de finalização de manobra orbital com falha |
| | Sinaliza_Fora_Atitude | Sinalização por TM de falha na execução de comando de manobra orbital por atitude estar fora do esperado |
| | Sinaliza_Início_ Manobra | Sinalização por TM de início de Manobra Orbital |
| | Sinaliza_Fim_Manobra | Sinalização por TM de finalização de atuação de manobra orbital |
| | Sinaliza_Manobra_não _permitida | Sinalização por TM de Manobra não permitida no modo |
| | Nenhuma_Saída | Sucesso (nenhuma falha) na execução do controle |

Tabela B.4. Tabela completa de exceções especificadas

| **Requisito** | **Descrição** | **Máquina COFI** |
|---|---|---|
| 1 | O controlador não deve permitir uma ação de controle de manobra orbital ser executada fora do modo correto | S_CA_EE |

Continua

Tabela B.4. - Continuação

| | | |
|---|---|---|
| 2 | O controlador deve em um pior-cenário de requisições de uso de CPU da missão garantir o tempo de execução das tarefas de controle | NA |
| 3 | O controlador deve mesmo na ocorrência de falhas finalizar atuação sobre órbita antes de alterar o modo | S_MO_EE |
| 4 | A lógica e sequência de ações na operação de uma manobra orbital deve garantir uma finalização segura da manobra mesmo no caso de falhas relevantes a mesma | NA |
| 5 | Os atuadores de força lineares devem inibir a atuação mesmo na ocorrência de falha simples no atuador | NA |
| 6 | O controlador deve detectar desvios de comportamento dos atuadores | S_MS_EE |
| 7 | O controlador deve detectar erros no comando (comunicação) dos atuadores | S_CA_EE, S_MO_EE |
| 8 | O controlador deve detectar erros nos tempos de comandos (comunicação) | S_CA_EE, S_MO_EE |
| 9 | O controlador deve comandar um estado seguro do atuador caso este sinalize atuação em momento indevido | S_CA_EE, S_MO_EE |
| 10 | O controlador deve detectar atuação espúria visto desvios da dinâmica esperada para a planta | S_MS_EE |
| 11 | O controlador deve levar a um apontamento seguro caso detecte degradação de desempenho do controle | - |
| 12 | O controlador deve detectar erros de hardware e software durante sua execução e chavear para modo de operação seguro em caso de falha. Obs. Erros de hardware: erro de memória, erro de processador. Erros de software: Divisões por zero, Valores numéricos fora de faixa (NaN, +-inf), erro na obtenção de recursos | S_MS_FH |
| 13 | O Controlador deve levar a um apontamento seguro caso haja perda dos tempos de execução das tarefas | - |
| 14 | O controlador deve levar a um apontamento seguro caso haja degradação de energia no satélite | S_MS_EE |

Continua

Tabela B.4. - Conclusão

| | | |
|---|---|---|
| 15 | O controlador deve detectar leituras não coerentes de sensores. Obs. Descontinuidades e congelamento de valor | S_MS_EE |
| 16 | O controlador deve detectar erros nos tempos dos dados dos sensores (comunicação) | S_CA_EE, S_MO_EE |
| 17 | O controlador deve ser chavear para modo seguro após evoluir a dinâmica do sistema mesmo com sensor em falha por até X tempo | S_MS_EE |
| 18 | O controlador deve entrar em modo seguro de apontamento se houver simultaneamente mais de uma falha em sensores | S_MS_EE |
| 19 | O controlador deve entrar em modo seguro caso o comportamento da planta comece a divergir do esperado mesmo se não houver falhas detectadas de equipamentos | S_MS_EE |
| 20 | O controlador deve detectar comunicação com dados corrompidos nos sensores digitais | S_CA_EE, S_MO_EE |

- **Serviço Mudança de Modo**

É apresentado o modelo normal CoFI do serviço.

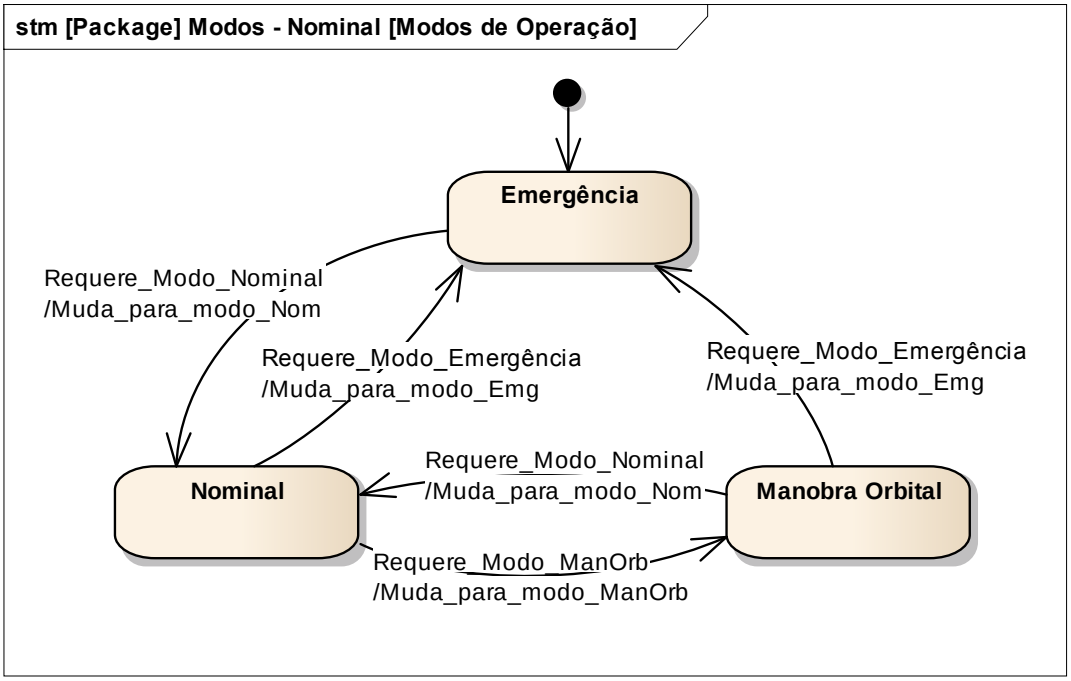


Figura B.1. S_MM_N: Modelo Normal – Mudança de Modo

Em seguida é apresentado a análise e o modelo de caminhos furtivos.

Tabela B.5. Tabela de transições de estados Serviço Mudança de Modo

| **Eventos/Estados** | **Modo Emergência** | **Modo Nominal** | **Modo Manobra Orbital** |
|---|---|---|---|
| **Requere_Modo_ Nominal** | Muda_para_ modo_nom | CF: Sinaliza_mudanca _nao_permitida | Muda_para_modo _nom |
| **Requere_Modo_ ManOrb** | CF: Sinaliza_mudanca _nao_permitida | Muda_para_modo _manorb | CF: Sinaliza_mudanca _nao_permitida |
| **Requere_Modo_ Emergência** | CF: Sinaliza_mudanca _nao_permitida | Muda_para_modo _emg | Muda_para_modo _emg |

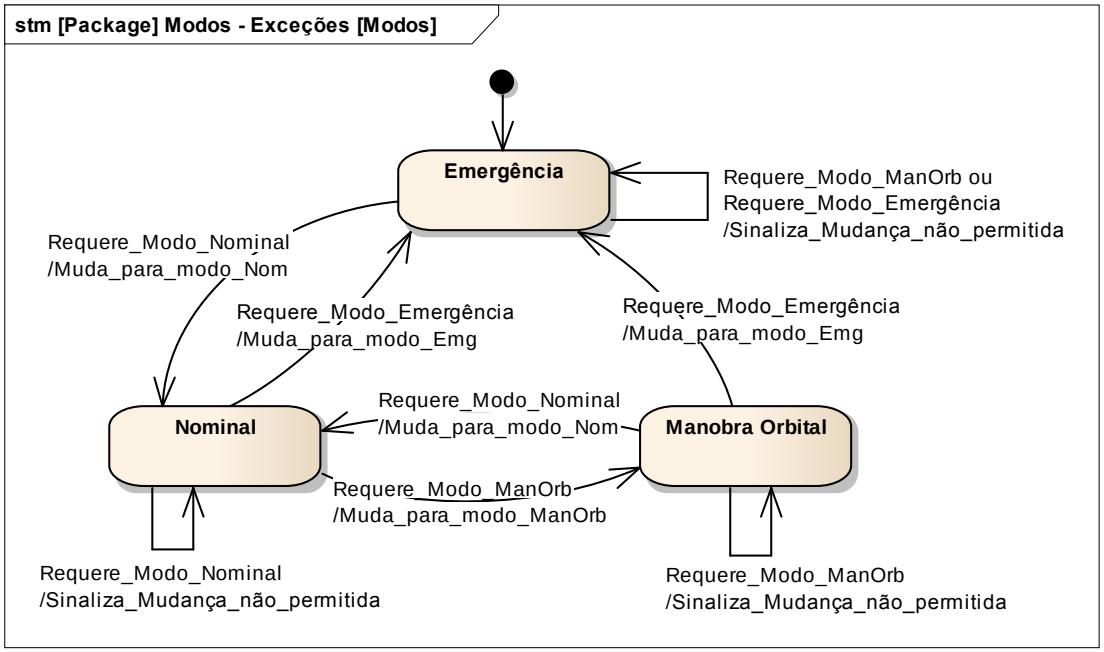


Figura B.2. S_MM_CF: Modelo de Caminhos Furtivos - Mudança de Modo

Este serviço do protótipo AOCS não contém modelo de tolerância a falhas de hardware pois nenhuma falha de hardware influencia diretamente nesta máquina. Este serviço também não teve nenhuma exceção especificada pelos requisitos encontrados. Em um caso real provavelmente os caminhos furtivos encontrados neste serviço estariam na especificação do software.

Este modelo representa o mais alto nível hierárquico das máquinas do protótipo de controle pela metodologia CoFI. Como no caso de um sistema de controle real, as funções são realizadas em um modo de operação.

- **Serviço Monitoramento de Saúde**

É apresentado o modelo normal CoFI do serviço.

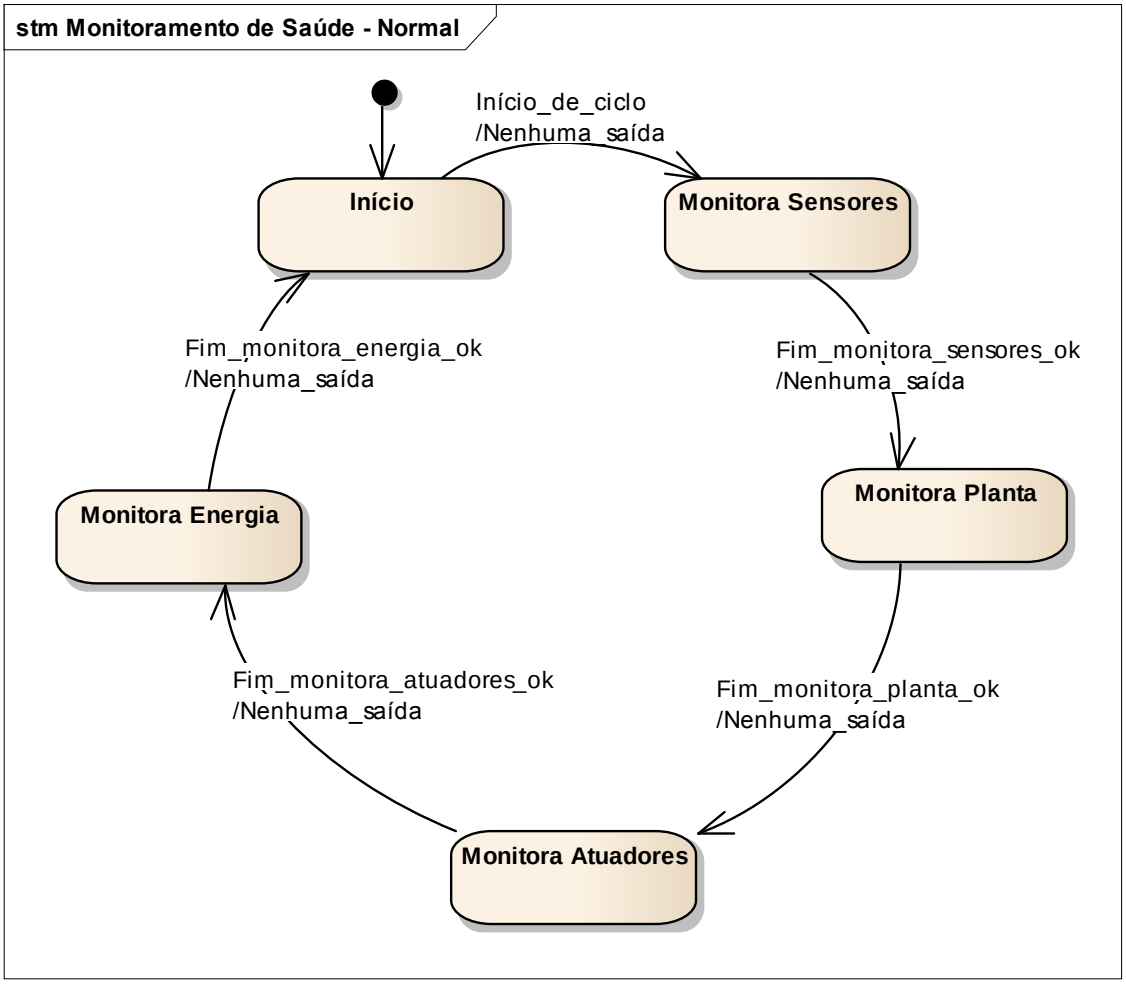


Figura B.3. S_MS_N: Modelo Normal - Monitoramento de Saúde

Em seguida é apresentado o modelo de exceções especificadas.

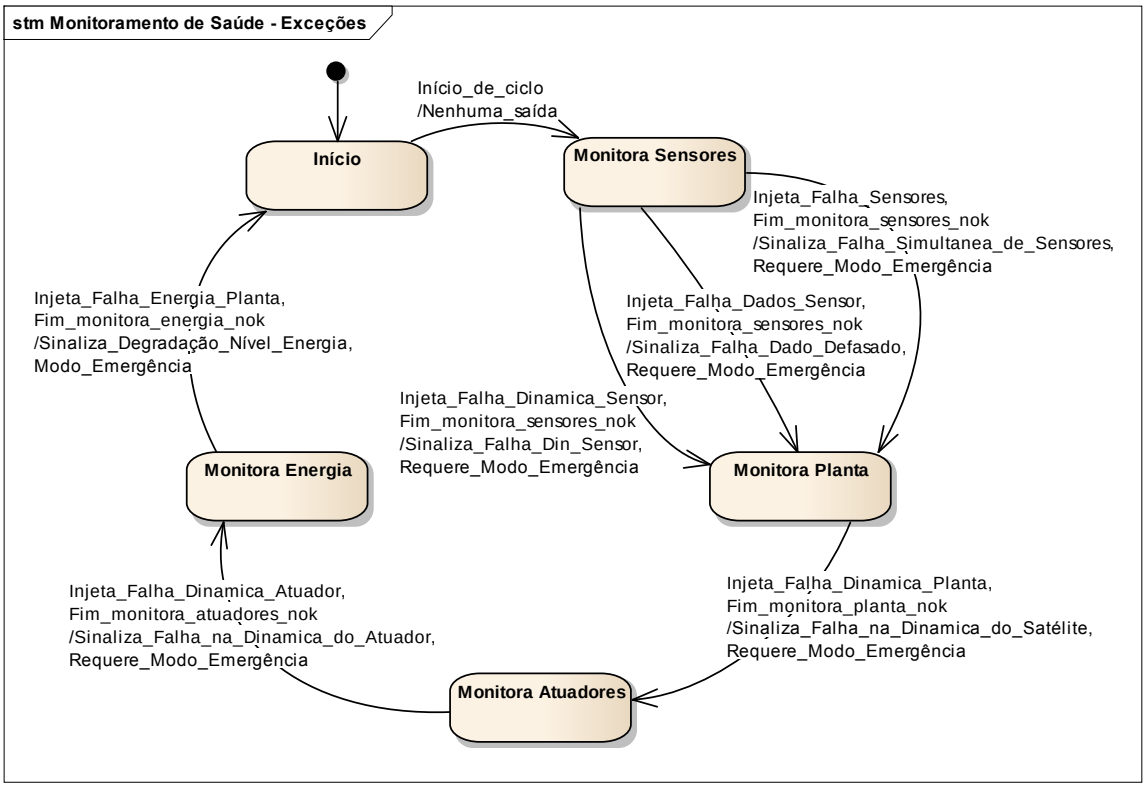


Figura B.4. S_MS_EE: Modelo Exceções Especificadas - Monitoramento de Saúde

Não é possível gerar o modelo de caminhos furtivos pois estas funções são realizadas de forma cíclica, com o fim de uma sendo o evento de inicialização da próxima, de modo que as ações são naturalmente sequenciadas, e não faz sentido a indicação de execução de eventos fora de ordem, uma vez que a implementação as amarra, isso por causa da característica esta aplicação. Também podemos ver que mesmo com a ocorrência de falhas as funções continuam a executar sequencialmente. A mudança para o modo de emergência somente ocorre no fim do ciclo.

Vale a pena ressaltar que a máquina apresenta uma ação de recuperação que é o chaveamento para o modo de emergência, condizente com a proposta de FDIR de meio-satélite do protótipo de AOCS, ver seção 5.2.

O modelo de tolerância a falhas de hardware é apresentado.

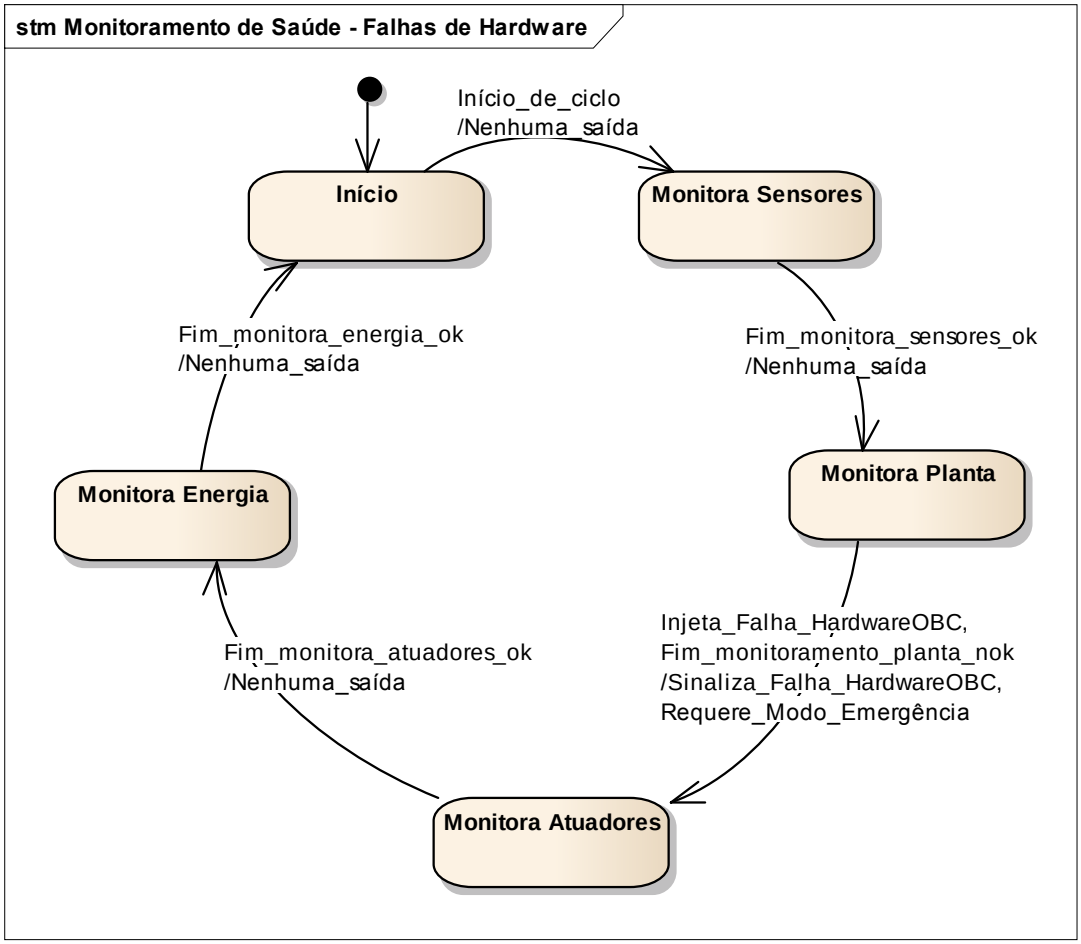


Figura B.5. S_MS_FH: Modelo Tolerância a Falhas de Hardware - Monitoramento de Saúde

A figura I. 20 ilustra um modelo de Tolerância Falha, com o evento Injeta_Falha_HardwareOBC. Porém, dado que o Sistema de teste não possibilita a ativação de uma falha de hardware, este modelo é apenas ilustrativo neste contexto.

- **Serviço Controle de Atitude (Modo Nominal)**

É apresentado o modelo normal do serviço.

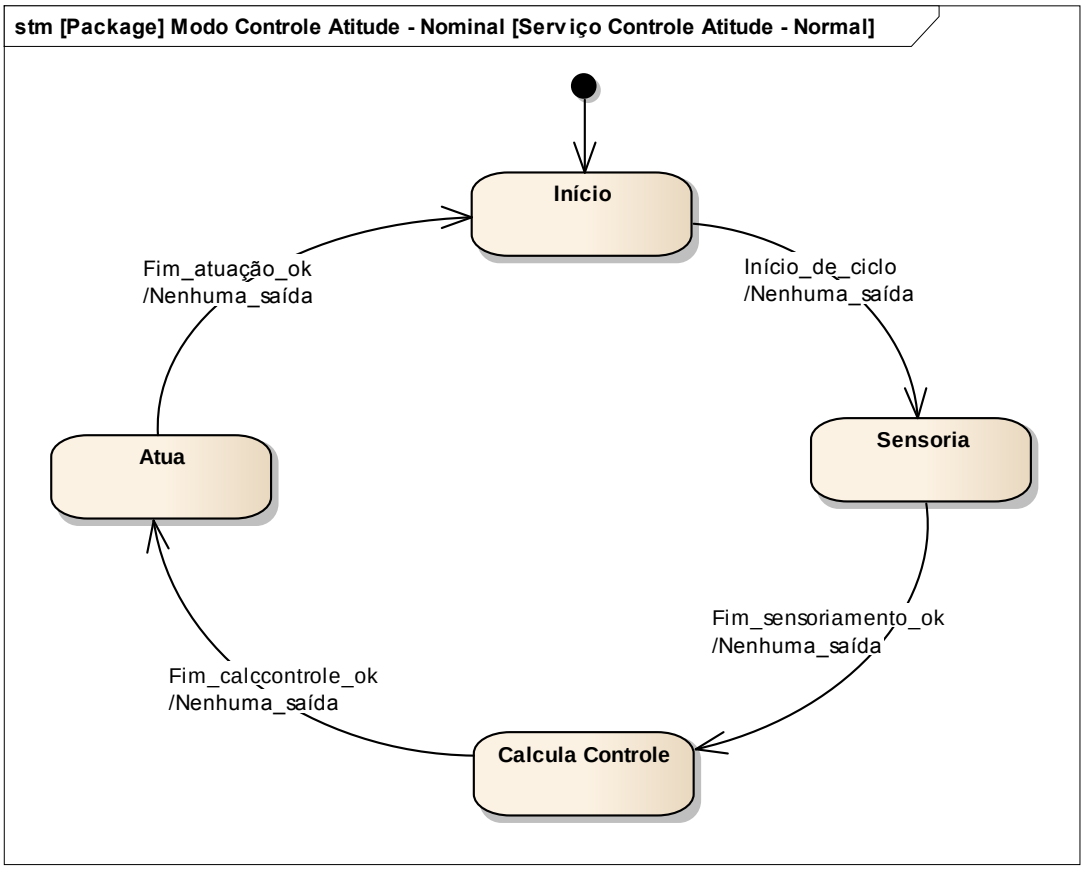


Figura B.6. S_CA_N: Modelo Normal - Controle de Atitude

Em seguida é apresentado o modelo de exceções especificadas.

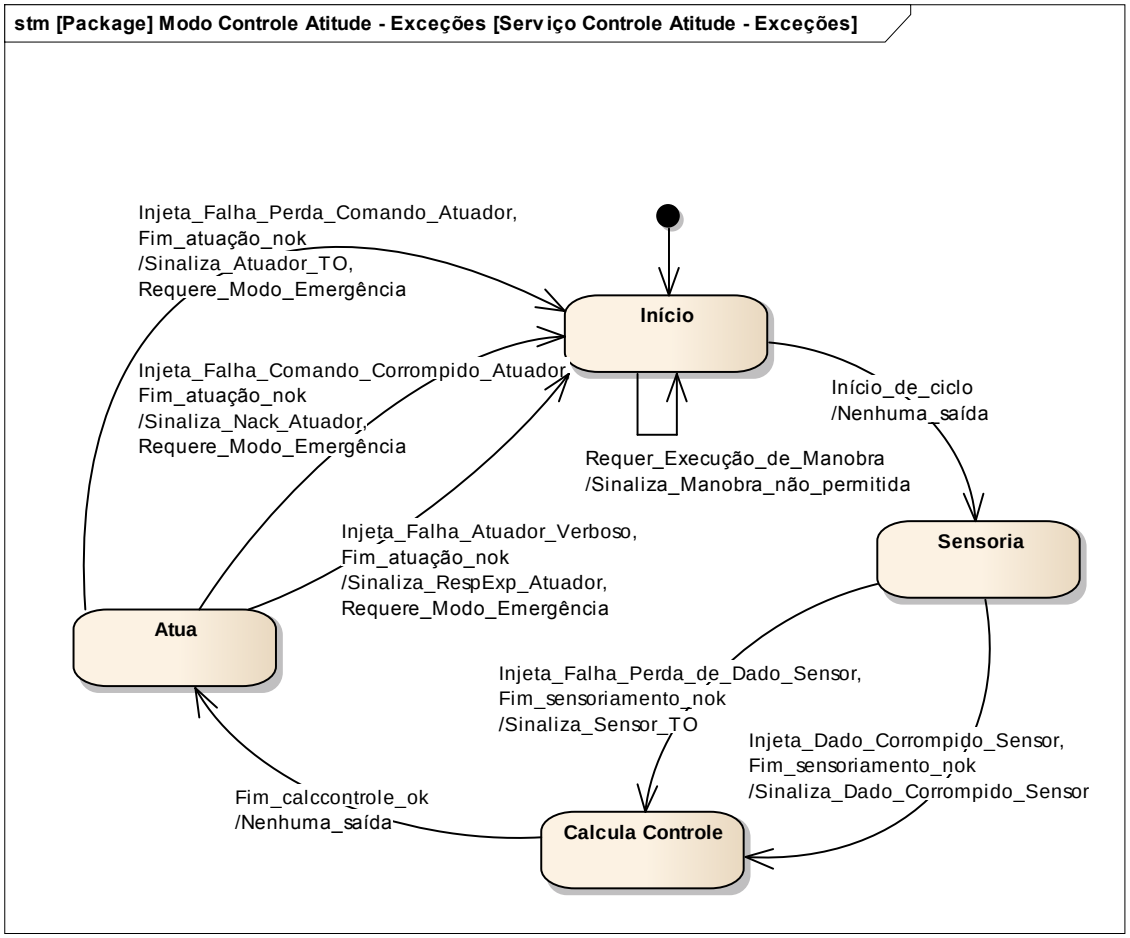


Figura B.7. S_CA_EE: Modelo Exceções Especificadas – Controle de Atitude

O modelo apresenta as funções que ocorrem ciclicamente durante o modo nominal de controle de atitude. Neste modo a cada ciclo o controle irá Realizar Sensoriamento, Calcular o controle, Atuar, e por fim Monitorar a Saúde do sistema.

Outras características da modelagem citadas anteriormente podem ser vistas, como o sequenciamento de funções mesmo com os diferentes resultados de sucesso das funções predecessoras.

- **Serviço Manobra Orbital (Modo Manobra Orbital)**

É apresentado o modelo normal do serviço.

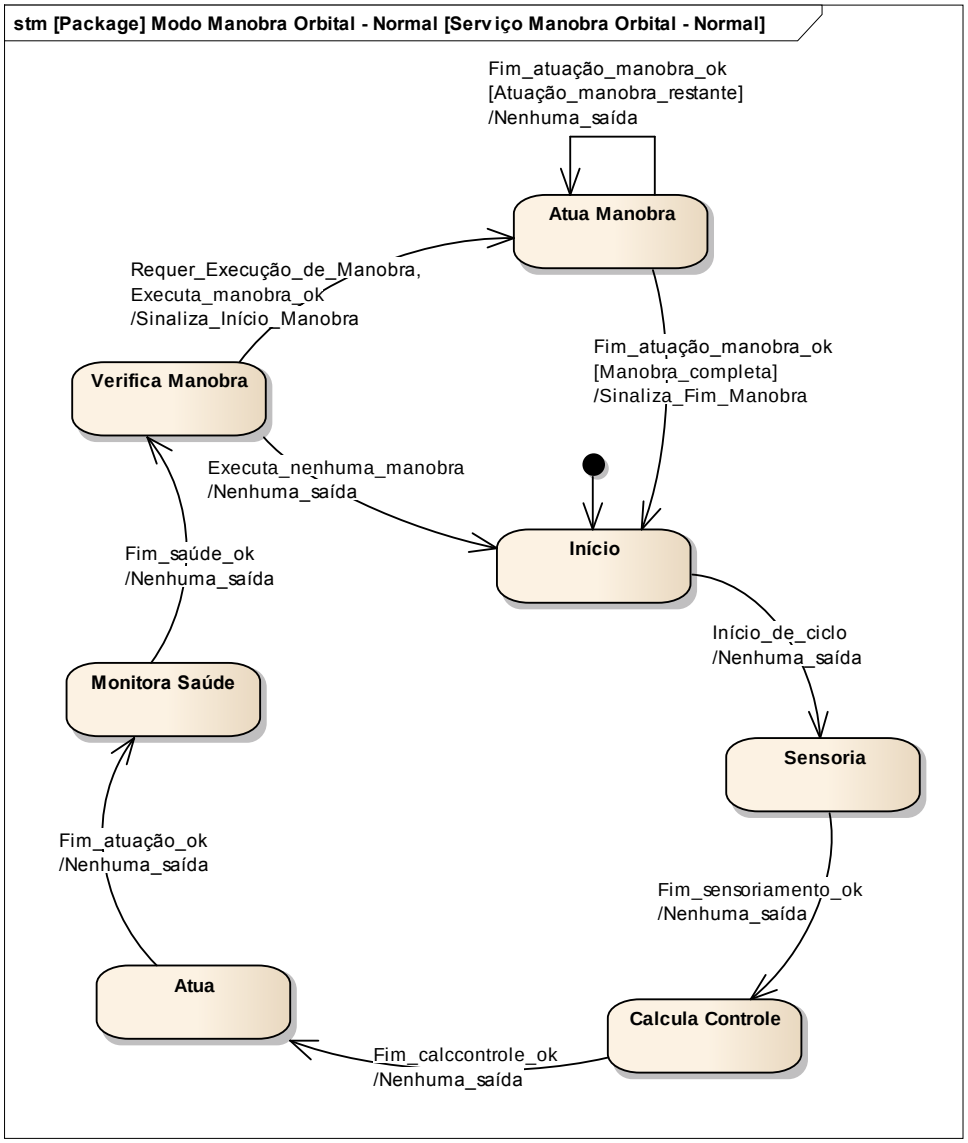


Figura B.8. S_MO_N: Modelo Normal - Manobra Orbital

Em seguida são apresentadas as exceções especificadas e o modelo de exceções especificadas.

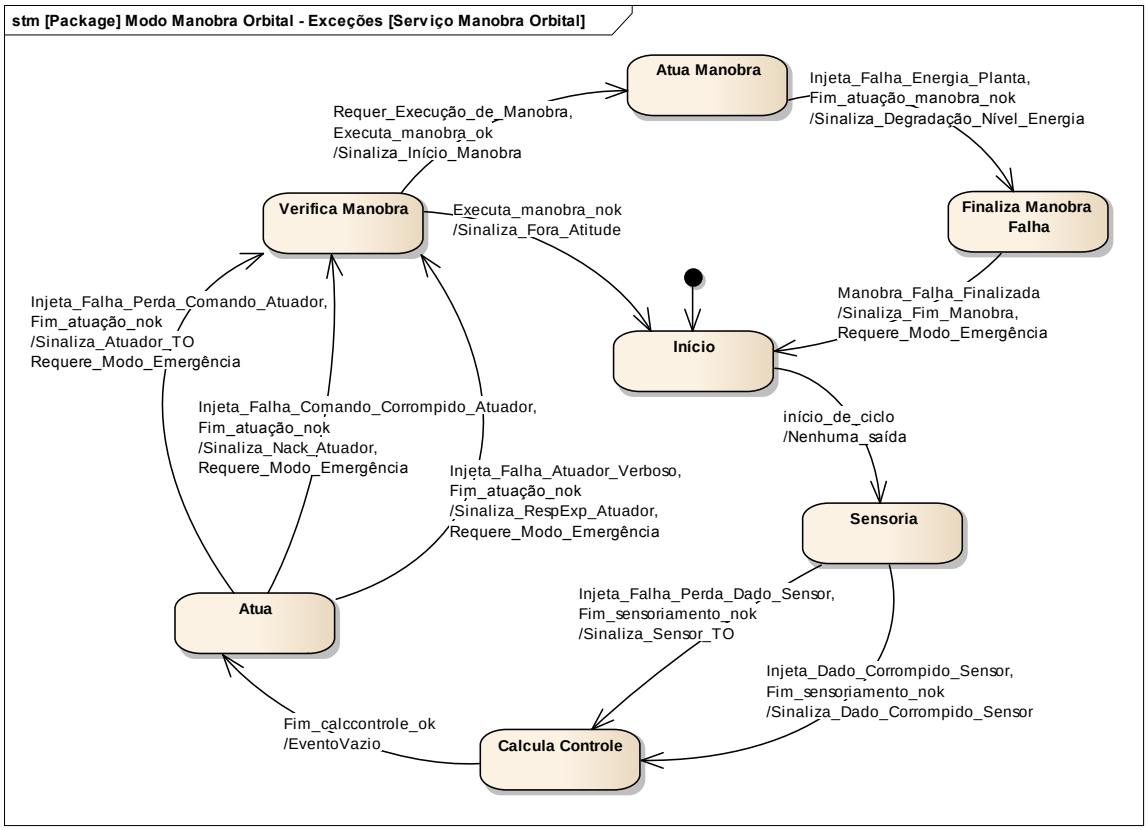


Figura B.9. S_MO_EE: Modelo Exceções Especificadas - Manobra Orbital

O modelo apresenta as funções do modo de manobra orbital, que englobam as mesmas funções do modo de controle de atitude e as funções de manobra orbital, que consistem em execuções sequenciadas de atuações por tempos definidos. A atuação em uma manobra orbital tem malha fechada com solo, ver seção 5.2., portanto depois de iniciada a manobra para por uma falha ou executa até o fim.