MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21c/2019/01.07.11.32-TDI

METODOLOGIA DE INTEGRAÇÃO, INTERCONEXÃO E REUSO DE SOFTWARE EM UM PROCESSO COLABORATIVO NO AMBIENTE RCE E SUA APLICAÇÃO À CONCEPÇÃO DE SATÉLITES

Leonardo Leite Oliva

Tese de Doutorado do Curso de Pós-Graduação em Engenharia e Tecnologia Espaciais/Mecânica Espacial e Controle, orientada pelo Dr. Marcelo Lopes de Oliveira e Souza, aprovada em 21 de dezembro de 2018.

URL do documento original:
<http://urlib.net/8JMKD3MGP3W34R/3SGMREP>

INPE
São José dos Campos
2018

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GBDIR)
Serviço de Informação e Documentação (SESID)
CEP 12.227-010
São José dos Campos - SP - Brasil
Tel.:(012) 3208-6923/7348
E-mail: pubtc@inpe.br

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21c/2019/01.07.11.32-TDI

METODOLOGIA DE INTEGRAÇÃO, INTERCONEXÃO E REUSO DE SOFTWARE EM UM PROCESSO COLABORATIVO NO AMBIENTE RCE E SUA APLICAÇÃO À CONCEPÇÃO DE SATÉLITES

Leonardo Leite Oliva

Tese de Doutorado do Curso de Pós-Graduação em Engenharia e Tecnologia Espaciais/Mecânica Espacial e Controle, orientada pelo Dr. Marcelo Lopes de Oliveira e Souza, aprovada em 21 de dezembro de 2018.

URL do documento original:
<http://urlib.net/8JMKD3MGP3W34R/3SGMREP>

INPE
São José dos Campos
2018

Dados Internacionais de Catalogação na Publicação (CIP)

Oliva, Leonardo Leite.

Ol4m Metodologia de integração, interconexão e reuso de software em um processo colaborativo no ambiente RCE e sua aplicação à concepção de satélites / Leonardo Leite Oliva. – São José dos Campos : INPE, 2018.

xxviii + 317 p. ; (sid.inpe.br/mtc-m21c/2019/01.07.11.32-TDI)

Tese (Doutorado em Engenharia e Tecnologia Espaciais/Mecânica Espacial e Controle) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2018.

Orientador : Dr. Marcelo Lopes de Oliveira e Souza.

1. Engenharia de Sistemas. 2. Modelagem multi-paradigma. 3. Processos colaborativos. 4. Diagramas de interconexão. I.Título.

CDU 629.78:629.7.06

Aluno (a): ***Leonardo Leite Oliva***

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

Engenharia e Tecnologia Espaciais/Mecânica Espacial e Controle

Dr. Mario Cesar Ricci

Presidente / INPE / SJCampos - SP

( ) Participação por Video - Conferência

(X) Aprovado ***( ) Reprovado***

Dr. Marcelo Lopes de Oliveira e Souza

Orientador(a) / INPE / SJCampos - SP

( ) Participação por Video - Conferência

(X) Aprovado ***( ) Reprovado***

Dr. Leonel Fernando Perondi

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

( ) Participação por Video - Conferência

(X) Aprovado ***( ) Reprovado***

Dr. Fernando José de Oliveira Moreira

Convidado(a) / EMBRAER / SJCampos - SP

( ) Participação por Video - Conferência

(X) Aprovado ***( ) Reprovado***

Dr. Guilherme Conceição Rocha

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

( ) Participação por Video - Conferência

(X) Aprovado ***( ) Reprovado***

Este trabalho foi aprovado por:

( ) maioria simples

(X) unanimidade

São José dos Campos, 21 de dezembro de 2018

"No futuro, Deus será concebido não como hoje, antropomorficamente, mas cientificamente, como um organismo conceitual de princípios e de leis sempre em ação, produtos executivos de uma vontade sempre presente em todos os campos, positivo e universal como os já descobertos pela ciência."

Pietro Ubaldi

AGRADECIMENTOS

Esse trabalho, em tudo que possui de original, geral e útil, não seria possível sem a presença contínua e amorosa de forças desconhecidas, situadas no imponderável, que no entanto são a causa dos maiores feitos. Essas forças emanam de Deus.

Igualmente, seria impossível adentrar em território desconhecido sem a compreensão e apoio de minha querida esposa, Vanessa, que sempre esteve ao meu lado nas dificuldades e sempre me apoiou no que eu estava fazendo.

Agradeço a meu pai, que deu conselhos valiosos em certos assuntos, e me criou da melhor forma possível – materialmente, culturalmente e espiritualmente. E à minha mãe, Suely, que de regiões muito mais tranquilas me acompanha e trabalha para o meu desenvolvimento.

Ao Rodrigo Britto Maria, que me ofereceu auxílio de grande valor – que me possibilitou desvendar novos horizontes.

Sou grato pela ajuda do meu orientador, Dr. Marcelo Lopes, que sempre soube receber minhas ideias e delinear o desenvolvimento geral do que estava a ser feito, dando significado a cada parte.

Por último, mas não menos importante, sou muito grato ao INPE, por sempre ter me passado conhecimento e companheirismo, dando-me a oportunidade de me tornar cada vez mais competente e ousado em termos profissionais. À Banca examinadora e a todas pessoas envolvidas direta ou indiretamente nesse processo, meu muito obrigado.

RESUMO

Sistemas como satélites, aviões, automóveis, hospitais,cidades, entre outros, vêm se tornando cada vez mais complexos e/ou altamente integrados. Todos integram diversas tecnologias e lidam com fenômenos díspares (físicos, químicos, biológicos, sociais, ecológicos, etc.) em subsistemas que trabalham em condições severas. Por essas razões são projetados através de diversos processos, usando muitos modelos e grande quantidade de dados. Estes devem ser combinados, avaliados e melhorados progressivamente, através da experiência e das modificações de contexto nos quais operam. Para suprir essa evolução, as tecnologias de computação e comunicação vêm oferecendo um suporte cada vez mais amplo, originando diversas cadeias de processos. No entanto, pouco tem sido feito no sentido de integrar e interconectar de forma eficiente ferramentas tão díspares, além de não haver uma simbologia capaz de mapear possibilidades de arranjos de cadeias de processos a partir de uma dada disponibilidade de *software*, ferramentas e modelos.Por isso, nesta tese desenvolveu-se uma metodologia de integração, interconexão e reuso de *software* em um processo colaborativo no ambiente RCE e sua aplicação à concepção de satélites. Para realizar isso, partiu-se dos princípios da Engenharia de Sistemas Baseada em Modelos, fazendo um levantamento geral de requisitos e conceitos, e do uso de diferentes abordagens, domínios físicos e plataformas de simulação – que devem interagir coerentemente entre si. A seguir, desenvolveram-se e integraram-se no ambiente colaborativo ferramentas computacionais de diferentes naturezas. Através de um diagrama de relações inter-ferramentas aqui desenvolvido, é possível mapear subsistemas de diferentes categorias de satélites e adotar políticas de reuso de ferramentas eficazes, utilizando a seleção e fusão das mesmas. Os modelos desenvolvidos usam diferentes abordagens (física, informacional, híbrida, etc.) e englobam domínios físicos diversos (mecânico, hidráulico, termodinâmico, etc.). No caso de estudo, construíram-se duas cadeias de processo referentes às fases iniciais (0 e A) de um subsistema de controle de atitude de um microssatélite. As conclusões sintetizam os progressos feitos em termos de sistematizar o desenvolvimento de sistemas na fase inicial, destacando as iniciações e avanços em temas novos. Dentre os aspectos originais destacam-se: (a) um diagrama de classificação de ferramentas; (b) a sistematização do processo de desenvolvimento inicial; (c) a construção de um diagrama dinâmico inter-ferramentas, com transformações lógicas; (d) a elaboração e execução de cadeias de processo multidisciplinares, multidomínio e multiplataforma, de modo automático para análises paramétricas.

Palavras-chave: Engenharia de Sistemas. Modelagem multi-paradigma.Processos colaborativos.Diagramas de interconexão.

A METHODOLOGY FOR SOFTWARE INTEGRATION, INTERCONNECTION AND REUSE IN A COLLABORATIVE PROCESS IN THE RCE ENVIRONMENT AND ITSAPPLICATION TO THE CONCEPTION OF SATELLITES

ABSTRACT

Systems such as satellites, airplanes, automobiles, hospitals, cities, among others, are becoming increasingly complex and/or highly integrated. All of them integrate different technologies and embrace several phenomena (physical, chemical, biological, social, ecological, etc.) in subsystems that work in severe conditions. For those reasons they are designed through different processes, using several models and a great amount of data. These models and processes must be combined, improved and evaluated through system life cycle, experience and context modification in which they operate. To support this evolution there are computer and communication technologies that offer a broad spectrum of work possibilities that translates into different process chains. However, little has been done in the sense of integrating and interconnecting so diverse tools with efficiency. Besides that, there is lack of a symbology capable of mapping linking possibilities in process chains from a certain software (tools and models) availability.Therefore, this thesis develops a methodology for software integration, interconnection and reuse in a collaborative process in the RCE environmentand its application to the conception of satellites. To accomplish this, several principles of Model Based Systems Engineering were adopted, thorough concepts and requirements analysis, use of different approaches, physical domains and simulation platforms – that must interact coherently between themselves. After this stage, different tools were developed and integrated in the collaborative environment – whose main differential consists in remote execution. Through a inter-tool relation diagram, it is possible to map subsystems of different satellites and adopt efficient reuse policies for tools, through fusion and selection. The models developed use different approaches (physical, informational, hybrid) and represent diverse physical phenomena (mechanical, hydraulic, thermodynamic, etc.). As a case study two process chainsrelated to early phases (0 and A) of a microsatellite attitude control subsystem were developed. Conclusions synthesize progress done in terms of systems development systematization in early phases, highlighting initiations and advances in each new topic proposed. Within those original topics we can quote: (a) a tools classification diagram; (b) the systematization of the initial development process; (c) construction of an inter-tool dynamic diagram, with logical transformations and; (d) construction and execution of automated multidisciplinary, multidomain and multiplatform process chains, for parametric analysis.

Keywords: Systems Engineering. Multi-paradigm modeling.Collaborative process.Interconnection diagrams.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE SIGLAS E ABREVIATURAS

ACS	*Attitude Control System*
CAD	*Computer Aided Design*
CAE	*Computer Aided Engineering*
DB	Diagrama de Blocos
DLR	*Deutschen Zentrums für Luft- und Raumfahrt*
EMS	Engenharia de Modelagem e Simulação
EP	Engenharia de Projeto
ERS	Engenharia de Requisitos e Sistemas
ESBM	Engenharia de Sistemas Baseada em Modelos
FCV	*Flow Control Valve*
GL	Grafos de Ligação
GUI	*Graphical User Interface*
INPE	Instituto Nacional de Pesquisas Espaciais
LQR	*Linear Quadratic Regulator*
M&S	Modelagem e Simulação
MMM	Multi-paradigma, Multi-disciplinar, Multi-domínio
PMM	Plataforma Multi-Missão
SimMoLib	Simulation Model Library
P2P	Peer-to-Peer
RCE	*Remote Component Environment*
TSM	*Total System Model*
SCAO	Subsistema de Controle de Atitude e Órbita

LISTA DE SÍMBOLOS

$\vec{A}$	Sistema LVLH com origem no centro de massa do satélite
A	Área frontal do veículo
A_s	Área de saída do bocal
$\vec{B}$	Sistema de referência em relação a $\vec{A}$
C_D	Coeficiente de arrasto
C_d	Coeficiente de descarga da válvula
e	Variável de esforço
F_m	Força magnética da válvula solenoide $[N]$
F_1	Empuxo na direção *roll*$[N]$
F_2	Empuxo na direção *pitch*$[N]$
F_3	Empuxo na direção *yaw*$[N]$
f	Variável de fluxo
I_1	Momento de inércia principal (*roll*) $[kg \cdot m^2]$
I_2	Momento de inércia principal (*pitch*) $[kg \cdot m^2]$
I_3	Momento de inércia principal (*yaw*) $[kg \cdot m^2]$
K_v	Constante de válvula
K	Ganhos de retroalimentação
K_i	Ganhos integrais LQR
K_{LQR}	Ganhos de retroalimentação LQR
m_c	Massa do carretel (válvula) $[g]$
m_p	Massa de propelente $[g]$
$\dot{m}$	Vazão mássica de propelente $[g/s]$
n	Frequência orbital $[1/s]$
R	Raio médio do tanque $[m^3]$
T_a	Torque aerodinâmico $[N \cdot m]$
T_m	Torque de campo magnético $[N \cdot m]$
T_g	Torque de campo gravitacional $[N \cdot m]$

$\vec{u}$	Vetor unitário de velocidade do veículo
$\{u\}$	Vetor de sinal para o satélite (ou válvula de controle)
V_e	Velocidade de ejeção dos gases $[m/s]$
$\{x\}$	Vetor de estados do satélite
ρ_p	Densidade do propelente
θ	Ângulo de *pitch*
φ	Ângulo de *roll*
Ψ	Ângulo de *yaw*
μ_E	Constante de permissividade magnética

SUMÁRIO

Pág.

1 INTRODUÇÃO

1.1. **Motivação**

Nas últimas décadas viu-se o desenvolvimento de diversas ferramentas computacionais para estudo de fenômenos distintos (CAD, Inventor, NASTRAN, Matlab/Simulink, Scilab, AMESim, 20-Sim, Dymola, OpenModelica, SysML, etc). No entanto, ainda foi feito pouco progresso no sentido de torná-las aptas a intercambiarem informações entre si.

Nesse sentido, diversas instituições mundo afora vêm se preocupando em desenvolver um ambiente colaborativo que torne as fases iniciais de desenvolvimento de sistemas mais eficaz. Na NASA, pode ser citada a inciativa de (CHEN at al., 2006), focado em otimizar uma rede para auxiliar dados geoespaciais. Na ESA (BANDECCHI, et al. 1999) trabalham com o uso de engenharia simultânea para apoiar projetos e missões espaciais. Na Europa, a Agência Espacial Alemã (DLR) vêm desenvolvendo um ambiente colaborativo que satisfaça as necessidades de desenvolvimento de qualquer sistema nas fases iniciais de projeto (0 e A). Esse ambiente em desenvolvimento se chama RCE (*Remote Component Environment*).

Segundo Booch e Brown(2002):

> Um ambiente colaborativo de desenvolvimento (ACD) é um espaço virtual no qual diversos interessados de um projeto – mesmo que distanciados no espaço e no tempo – podem negociar, trocar ideias, debater, trocar conhecimentos, e geralmente trabalhar conjuntamente para efeturar alguma tarefa, sendo a mais comum a construção de um aparato executável e seus aparatos de apoio artificiais.(BOOCH; BROWN, 2002, p. 2)

A gênese do presente trabalho partiu de uma série de questionamentos a respeito do papel dos diferentes tipos de paradigmas, abordagens, métodos e ferramentas de Modelagem e Simulação (M&S) para o desenvolvimento de sistemas multi-domínio e integrados – presentes em aeronaves, satélites, redes de transporte, provedores energéticos, entre outros. À medida que as tecnologias avançam e se inter-relacionam, surgem interfaces antes inexistentes, e com isso a emergência de novosfenômenos, trazendo resultados imprevistos – e não raramente, indesejados. A questão fundamental deste trabalho é desenvolver uma metodologia de integração, interconexãoe reuso de

ferramentas computacionais aplicada a processos colaborativos, usando como apoio os princípios da Engenharia de Sistemas Baseada em Modelos (ESBM), que faz uso do diálogo entre equipes multidisciplinares nas fases iniciais de desenvolvimento de sistemas. Nesse estudo, de um subsistema de satélite. Dentre essas equipes, podemos elencar o Engenheiro de Requisitos e Sistemas (ERS) e o Engenheiro de Modelagem e Simulação (EMS). Enquanto o primeiroestá interessado em levantar requisitos de um problema, propor as especificações de uma solução, e executar a verificação e validação do sistema apoiado nas mesmas; o segundoestá focado no processo de desenvolvimento dos modelos e simulações dos sistemas, desde o levantamento de hipóteses e equacionamentos analíticos, passando pela elaboração dos modelos simbólicos, até chegar à execução dos mesmos. Soma-se a isso o Engenheiro de Projeto (EP), cuja finalidade é – junto aos processos de manufatura – materializar o sistema em termos sistemáticos, a partir das diretrizes gerais (missão).

Para realizar essa construção podem ser usadas abordagens distintas (física, informacional, mista, etc.), co-modelagem (modelos de simulação, descritivos, se articulando com modelos de otimização, prescritivos, por exemplo) e co-simulação (modelos distintos se retroalimentando). Isso à medida que uma corrida de simulação é executada, para integrar parte das particularidades presentes no desenvolvimento de um sistema complexo e melhorá-lo simultaneamente ao seu desenvolvimento virtual. Neste trabalho, procurou-se aprofundar essas questões a partir da concepção de um subsistema de satélite nas fases iniciais de desenvolvimento – terreno em que ERS, EMS e EPcoexistem com intensidades equivalentes – e buscar possíveis modos de contornar problemas de reuso e comunicação entre profissionais e ferramentas computacionais.

Quando se percebe uma necessidade, o primeiro passo consiste em identificar os interessados (*stakeholders*). Estes podem verbalizar seus desejos numa linguagem técnica ou não técnica. No entanto, mesmo dentre os técnicos existem problemas: as áreas são diversas, seus membros possuem formações diferenciadas, existem jargões específicos – e dificilmente abandonáveis. Isso torna necessária a existência de uma base universal de: 1) conhecimentos (modelo mental) e; 2) comunicação (modelo linguístico) sobre a qual todos os interessados possam se compreender e comunicar – e a partir da qual seja possível, sem equívocos, um engenheiro de requisitos e sistemas

elicitar uma série de requisitose propor especificações. A partir dos requisitos é possível construir modelos de simulação e otimização. Estes últimos para melhorar os parâmetros do sistema, de tal forma a respeitar os requisitos (restrições) e simultaneamente melhorar o rendimento geral do sistema (desempenho); aqueles para observar o comportamento das variáveis do sistema e suas relações à medida que o projeto amadurece – e o sistema opera.

Requisitos são a base de qualquer sistema. Eles definem detalhadamente o que os interessados – usuários, fornecedores, desenvolvedores, etc. – precisam desse sistema e informações (para os desenvolvedores) do que este precisa fazer para satisfazer essas necessidades. O desafio consiste em expressar os requisitos numa linguagem natural, compreensível a qualquer pessoa, e ao mesmo tempo evitar cair em ambiguidades. Isso é uma tarefa não trivial, pois o uso de jargões específicos é tanto um auxiliar (objetividade) quanto um empecilho (restrição de público apto a assimilar).

Para isso, aqui se desenvolveum novo processo multidisciplinar colaborativo apoiado na Engenharia de Sistemas Baseada em Modelos e aplicado à concepção de subsistemas de satélites. Isso inclui um conjunto de modelos a partir da identificação dos potenciais interessados, fazendo uso de um ambiente virtual colaborativo que integre todas as especialidades envolvidas em torno da melhoria de um subsistema em particular. Para isso será feito uso extensivo da Modelagem Multi-Paradigma (MMP) e Multi-Domínio (MMD), de tal forma a obter um modelo que reflita a melhor configuração possível dentro dos limites impostos inicialmente. Como ideia geral, é essencial que o estudo desenvolvido apresente Generalidade, Originalidade e Utilidade.

A **Originalidade**deste trabalho reside em:

a) **aplicar diferentes abordagens** (simbologia dos grafos de ligação e diagramas de blocos) para representar fenômenos distintos **efundir ferramentas diversas** (cálculos estruturais, simulação de planta, controle, convergência), que podem usar diferentes plataformas de simulação (ex: *AMESim, 20-Sim, OpenModelica, MatLab/Simulink, Scialb, Octave, Excel,* etc), **num mesmo ambiente de desenvolvimento**, que pode ser operado por qualquer nó da rede;

b) **criar duas cadeias de processoesistematizar todos os passos envolvendo as fases 0 e A de desenvolvimento de sistemas**, envolvendo múltiplas disciplinas e plataformas de simulação, e adaptá-las para executarem análises paramétricas de forma automatizada;

c) **criar um diagrama dinâmico de relaçõesinter-ferramentas** – com a possibilidade de realizar alterações na rede de relações através de operações – que permitam adotar políticas de reuso eficazes;

d) **desenvolver ferramentas auxiliaresde manuseio de dados**, que visibilizem uma maior integração entre ferramentas, com compartilhamento de parâmetros de projeto e de simulação.

A título de sustentar os desenvolvimentos, foram tangenciados outros aspectos – não originais mas de implementação relevante – como: (e) **consideraçãoda variação de parâmetros**tanto em nível de projeto quanto de operação nos modelos; (f) **representação simbólica e implementaçãocomputacional de fenômenos mecânicos** (dinâmica de atitude), **hidráulicose termo-pneumáticos** (micropropulsão) a partir do paradigma físico; (g) **representação e implementação de atuadores aeroespaciais** no espaço de estados.

A **Generalidade** deste trabalho reside em:

a) **possibilidade de aplicar um diagrama de relações entre quaisquer modelos** – de qualquer nicho da indústria ou da pesquisa – para uma política de reuso e mapeamento eficaz;

b) **análises de diferentes configurações possíveis de uma rede de desenvolvimento**, que são objeto de estudo para o desenvolvimento de qualquer sistema, sendo um ponto central;

c) **possibilidade de realizar melhoramentos de forma remota**, com **compartilhamento de informações** e resultados de maior fidelidade. Isto é de interesse primordial nas fases iniciais de qualquer processo de desenvolvimento, especialmente para sistemas de alto índice de complexidade – e,

consequentemente, custo – como os que o binômio sociedade-instituições vem demandando.

A **Utilidade** reside em:

a) **potencial reduçãode tempo e custo no desenvolvimento de sistemas complexos**;

b) **maior compartilhamento de informações de interesse** (ex: variáveis, parâmetros, relatórios) entre sujeitos que podem se encontrar em locais geograficamente distantes;

c) **a redução da demanda por licenças**(dependência e custo) pois estas se concentram onde são mais úteis;

d) **a execução rápida**(cada nó possui apenas sua ferramenta específica) sem sobrecargas inúteis;

e) **o controle de configuração**, uma vez que todos os nós da rede estão interligados e as atualizações de parâmetros e modelos se dão simultaneamente para todos – garantindo assim resultados atualizados e fiéis;

f) **a execução automatizadaem ambiente colaborativo** permite gerar conhecimentos compartilhados de forma rápida, otimizando a eficiência dos nós da rede ao lhes fornecer dados importantes envolvendo todo o sistema – revelando como a sua parte afeta o todo e como este influencia (limita ou potencia) sua ferramenta.

1.2. **Objetivos**

O objetivo desta tese é desenvolver uma nova metodologia de integração, interconexão e reuso de software em um processo colaborativo no ambiente RCE e sua aplicação à concepção de satélites.

Além disso, podemos elencar os seguintes objetivos específicos:

a) Sistematizar o processo de construção de uma rede colaborativa, através da elaboração e amadurecimento de um diagrama de processos;

b) Construir uma rede colaborativa com ferramentas de diferentes abordagens, domínios e plataformas de simulação, capaz de intercambiar informações entre si de forma integrada;

c) Construir um diagrama dinâmico de relações entre ferramentas computacionais que viabilize um mapeamento e política de reuso eficaz, através de operações de transformação;

d) Criar uma (ou mais) cadeia(s) de desenvolvimento, com integração de ferramentas e execuções a partir de qualquer nó da rede;

e) Aplicar a simbologia de Grafos de Ligação (GL) à dinâmica de veículos espaciais e seus subsistemas para sua implementação em ferramentas computacionais que operam baseada nesse paradigma;

A proposta inicial foi cumprir os itens **a**, **b**e **c**em sua totalidade. O item **d**é objeto central, e deve ser cumprido, se não com toda variedade prevista (abordagens, domínios e plataforma), ao menos com uma ou duas delas. O item**e**não constitui aspecto inovador em si, porém serve de embasamaneto para o caso de estudo, sendo portanto um exercício de conhecimento.Para problemas de execução de uma ferramenta, pode ser construída outra equivalente, em outra plataforma, para ao menos conseguir cumprir o objetivo de executar uma rede. Foi possível amadureceras redes colaborativas a título de constatar diferenças e pontos a serem desenvolvidos na cadeia de processo proposta – que é o desenvolvimento preliminar (fases 0 e A) de um microssatélite e um subsistema de controle de atitude.

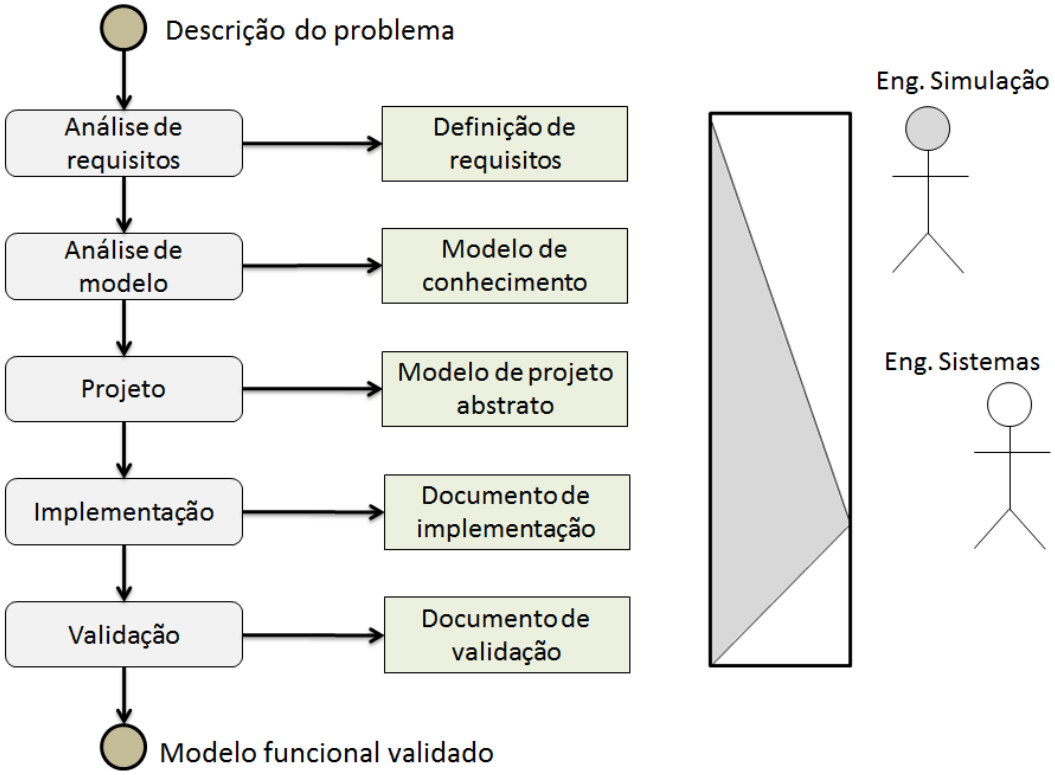
Ao final da Revisão de Literatura (Capítulo 2) será aprsentada uma síntese dos tópicos desenvolvidos e dos respectivos autores (Tabela 2.5), destacando o que já foi iniciado e está em processo de desenvolvimento nas diversas vertentes relacionadas à questão colaborativa em Engenharia de Sistemas, além das contribuições do autor dadas neste trabalho, relacionando os aspectos originais, gerais e úteis de cada tópico.

1.3. **Metodologia**

Conforme dito anteriormente, este trabalho se apoia no ambiente colaborativo RCE concebido pela DLR, que consiste em cinco passos (SCHAUS et al.,2013). Cada passo depende da atuação de dois profissionais: o Engenheiro de Requisitos e Sistemas e o Engenheiro de Modelagem e Simulação, conforme a Figura 1.1. O primeiro está interessado em: identificar interessados, definir métricas, elaborar requisitos, listar hipóteses, delinear o modelo do sistema a nível conceitual, e validar o modelo; o segundo está focado na elaboração do modelo analítico e computacional, além de cuidar da implementação, com preocupações referentes ao método numérico, abordagem, interfaces, escala de tempo, fenômenos físicos, lógicas, entre outros. Percebe-se que o peso do engenheiro de modelagem e simulação aumenta à medida que o modelo atinge maior grau de maturação, sendo máximo na fase de implementação; ao passo que o engenheiro de requisitos e sistemas tem maior importância na fase conceitual, em que deve identificar o problema (modelo do sistema a ser desenvolvido), os interessados e suas vontades (requisitos), inicialmente em linguagem comum, e posteriormente em termos técnicos; e propor especificações, opções, métricas e arquiteturas das soluções.

Nessa etapa o EP ainda não está presente. Podemos afirmar que o mesmo começa a adquirir protagonismo a partir do ponto em que existe um modelo funcional validado – que servirá para o projeto detalhado dos subsistemas e componentes, posteriormente. A interação EP e EMS é muito forte a partir dessa etapa.

Figura 1.1 – Grau de atividades do engenheiro de sistemas e de simulação.



Fonte: Adaptado de Schaus et al. (2013).

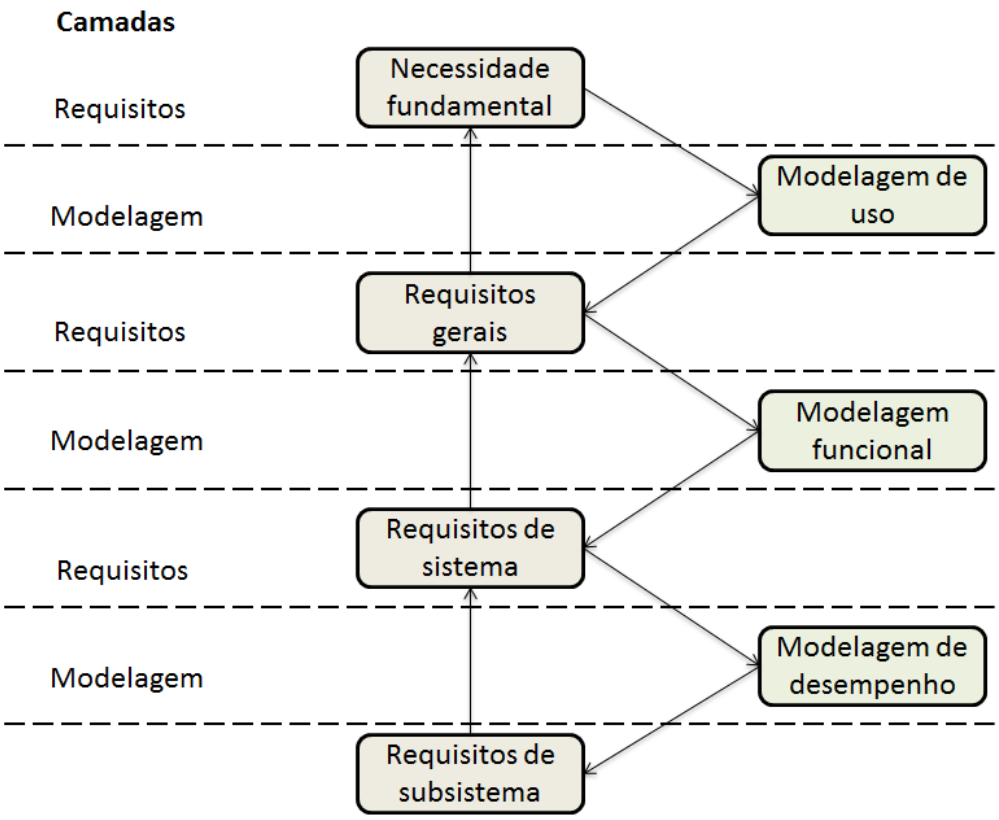
O processo de desenvolvimento de um modelo nas fases iniciais está na cadeia esquerda(etapas em cinza), ao passo que seus respectivos produtos estão ao lado (em verde). O diagrama da Figura 1.1 se refere à primeira parte da curva em “V” da Engenharia de Sistemas, que se refere à fase de elaboração de requisitos e delineamentos gerais da arquitetura do sistema. A entrada do processo são os interesses dos clientes (linguagem informal). A saída é o modelo funcional do projeto, isto é, em linhas gerais, sem especificidades de componentes ou fenômenos. Ou seja, o sistema a ser desenvolvido ainda não foi sequer materializado.

A etapa subsequente à saída do diagrama da Figura 1.1 seriam os projetospreliminar e detalhado. Nessa etapa a engenharia de modelagem e simulação volta a ter um papel relevante, passando a ser eixo central de desenvolvimento. Essa etapa – e outras posteriores – não são escopo deste trabalho.

O início do ciclo consiste em descrever um dado problema a partir de um conjunto de opiniões de grupos interessados em resolvê-lo. Essa descrição é encaminhada a uma equipe especializada, composta por engenheiros requisitos e sistemas, cuja função é analisar o problema e transformá-lo em requisitos, através de um laborioso processo que visa traduzir necessidades/desejos dos interessados em uma linguagem capaz de delimitar fronteiras claras (geométricas, cinemáticas, dinâmicas, financeiras, de tempo) e objetivos concretos (ex: velocidade mínima, eficiência, emissão de poluentes, sobressinal, etc).

É importante diferenciar modelagem do sistema e elicitação dos seus requisitos. Estes são extraídos/elicitados de grupos e/ou pessoas interessadas no projeto. Nesta proposta, a representação (modelos) do sistema é feita concomitantemente ao processo de amadurecimento dos requisitos, que vão se desdobrando desde a declaração das necessidades, que se tornam requisitos dos interessados, de sistema e de subsistemas, conforme a Figura 1.2.

Figura 1.2 – Requisitos e modelagem ao longo do desenvolvimento.



Fonte: Adaptado de Hull et. al. (2005).

A relação entre dois processos de desenvolvimento (requisito e de modelo) se dá através de uma interação paralela entre as atividades de modelagem e de elaboração de requisitos, sendo esta o objetivo e aquela um apoio – que visa facilitar o processo de elicitação de requisitos. Parte-se da necessidade fundamental, que é a plataforma para o modelo de uso, que por sua vez serve para elaborar os requisitos gerais (dos interessados), e assim sucessivamente.

1.4. Organização da Tese

A Tese está organizada em sete capítulos:

a) O primeiro Capítulo apresenta o que motivou o desenvolvimento deste trabalho e quais os objetivos a serem atingidos ao longo e na conclusão do mesmo. Além

disso, é delineada uma metodologia a ser seguida durante o desenvolvimento das ferramentas, destacando a importância de cada etapa;

b) O segundo Capítulo faz uma revisão de literatura, varrendo de modo orientado os principais assuntos a serem tangenciados e/ou aprofundados durante o desenvolvimento deste trabalho. Dentre eles podem ser elencados: (a) O escopo, métodos e desafios da Engenharia de Sistemas Baseada em Modelos (ESBM); (b)A ESBM estendida, com seus principais desafios e tendências; (c) O conceito de conexão de modelos para viabilizar o reuso e a visibilidade dos mesmos durante o desenvolvimento do ciclo de vida de um sistema, destacando ferramentas desenvolvidas por institutos de renome; (d) A descrição dos processos multidisciplinares colaborativos usando uma plataforma adequada para intercâmbio de informações e execução de modelos interconectados num processo de desenvolvimento; (e) a questão de interconexão de ferramentas;

c) O terceiro Capítulo faz uma esquematização do novo processo à luz da ESBM, inserindo o processo de desenvolvimento, com suas atividades, em diagramas. Nessa etapa é apresentado o diagrama de processo e a conexão existente entre engenharia de projeto, de sistemas e de simulação nas fases de desenvolvimento conceitual (0 e A). Traça-se também, a nível de esclarecimento, uma série de sujeitos e seus interesses no sistema proposto, elencando as métricas e medidas de efetividade de cada qual. Tendo definido as diretrizes gerais, são elaborados os requisitos dos subsistemas em questão. Descreve-se o processo de integração e interconexão das ferramentas na rede, além de sua execução, de forma manual e automática. As ferramentas são descritas usando uma árvore de classificação. Adicionalmente é apresentado um esboço incial de um diagrama de relações, com modelos e suas interconexões, requisitos de processo colaborativo e um diagrama de relações dinâmico inter-modelos. Esses conceitos são implementados para o caso de estudo, descrito no Capítulo 4;

d) O quarto Capítulo formula um caso de estudo. Inicia-se definindo, contextualizando e caracterizando os microssatélites, destacando o porquê de sua importância crescente, tangenciando aspectos como custo, potência, volume e massa. Traça-se um breve histórico, com tendências para os próximos anos. Em

seguida fala-se sobre as particularidades dos subsistemas de controle desses veículos, com foco na atitude, e em conceitos de micropropulsão. Dessa forma inicia-se um mapeamento de fatores críticos para o desenvolvimento de tal subsistema. Com essas definições é possível iniciar a identificação dos potenciais interessados no desenvolvimento do sistema proposto, baseado nas características do mesmo, passando para a definição dos atributos (interesses específicos de cada interessado) e medidas de efetividade (métricas para quantificar os atributos). Com isso feito foi possível iniciar a elaboração dos modelos simbólicos – que por sua vez foram implementados computacionalmente usando abordagens, domínios e plataformas computacionais distintas;

e) O quinto Capítulo apresenta os resultados da implementação da nova metodologia (Cap. 3) para o caso de estudo (Cap. 4), com observações dos diferenciais. As cadeias de processo fundamentais (1 e 2) são usadas para realizar análises paramétricas remotas automatizadas, com o escopo de explicitar a potencialidade da nova metodologia dentro de processos remotos;

f) O sexto e sétimo Capítulos tiram conclusões do presente estudo, destacando a importância dos processos colaborativos para o desenvolvimento de sistemas (Cap.6), além de apontar caminhos para pesquisas futuras (Cap.7). Esse último trata de aspectos de verificação e validação em nível sugestivo. Primeiramente recorre-se à validação conceitual, que independe da existência do sistema (relação modelo analítico – modelo computacional), o que permite estabelecer uma primeira aproximação de compatibilidade. Se as condições forem favoráveis, pretende-se aplicar conceitos de análise dimensional e semelhança para “construir resultados de teste” do sistema proposto a partir de um sistema real com resultados de teste (PMM). Dessa forma seria possível estabelecer uma validação apoiada na semelhança entre sistemas e geração de resultados de teste esperados.

2 REVISÃO DA LITERATURA

Por se tratar de um trabalho cujo foco está mais na metodologia e conceito, podemos inseri-lo parcialmente no **pensamento sintético/sistêmico**. Apesar de não estar diretamente relacionado, existe um fio condutor que permeia o desenvolvimento de cada etapa do presente estudo, caracterizado por um foco que está muito mais centrado na questão da interconexão entre elementos do que nos mesmos. Dessa forma, pode ser interessante fazer uma retrospectiva histórica, definindo formalmente o que é um sistema e definindo os três pilares (pressupostos) do pensamento sistêmico, que adicionam – não negam – uma nova perspectiva ao (já consolidado) **pensamento analítico/elemental**. Esse resgate histórico e conceituação se encontra no Apêndice A deste trabalho.

Um segundo ponto relevante são os **modelos**. Por ser um vocábulo de grande amplitude, é preciso defini-lo de modo formal. Além disso, existem diversas classes de modelos, que servem tanto às ciências físicas e naturais (*hard*) e humanas e sociais (*soft*), quanto às engenharias. Consequentemente, para situar o leitor, é imprescindível fazer um resgate do tema, definindo, classificando e exemplificando esse vasto universo, que serve para aplicações de ponta tanto nas indústrias quanto nas ciências.

Além disso, existe um aspecto particular no universo dos modelos de simulação computacionais, que é o **tipo de abordagem**. De modo geral, ela pode ser tanto **física** quanto **informacional**. Aí entram os grafos de ligação (GL) e os diagramas de blocos (DB), respectivamente. Essas definições, juntamente com a de modelos, podem ser vistas no Apêndice B.

2.1. Engenharia de Sistemas Baseada em Modelos

Antes de tratar da Engenharia de Sistemas Baseada em Modelos (*Model Based Systems Engineering* – MBSE) convém falar sobre a Engenharia de Sistemas (ES). A ES tem como escopo abordar de forma diferenciada os Sistemas Complexos (SC). Estes são "*caracterizados pelas inter-relações, interações e interconectividade de elementos dentro de um sistema e entre este e o ambiente circunjacente"* (CHAN, 2001).

Quanto à definição de ES, podemos elencar algumas:

- *"...uma abordagem robusta no desenvolvimento, criação e operação de sistemas."* (*Systems Engineering Handbook*. NASA, 1995);
- *"A Arte e Ciência de criar sistemas eficientes..."* (HITCHINS, ex-presidente da INCOSE);

Apesar da importância das fontes, percebe-se a carência na definição da ES. Considerando três dos atributos de um **sistema** (componentes, interconexão e propósito), pode-se definir ES como a sistematização dos processos de desenvolvimento, de manufatura, de integração e testes de forma geral, usando simbologias e métodos adequados, podendo estes ser novos ou fazer uso de conhecimento já existente – adaptado para novas finalidades.

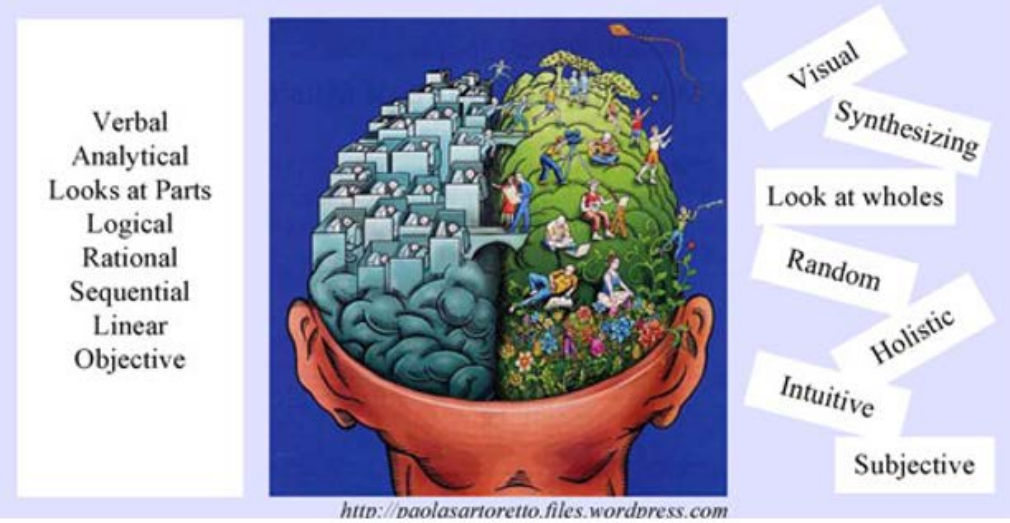
A interdisciplinaridade cresce à medida que os sistemas se tornam mais complexos e/ou altamente integrados, levando a uma maior interconexão entre suas partes. O INCOSE define o nascente campo como arte e ciência pelo seu aspecto dual: inexiste uma terminologia, métodos, processos, ferramentas, etcconsagrados dessa nova abordagem sistêmica, devendo-se trabalhar com ambos os hemisférios cerebrais, esquerdo e direito, conforme ilustra a Figura 2.1. Isso implica o uso de ferramental matemático-lógico, de conceitos científicos, de métodos e ferramentas já consolidados, que podem servir de apoio a uma nova abordagem. Mas, ao mesmo tempo, significa criar novas formas de conceber relações (interpretar), de representar (modelar), de relacionar (incorporar disciplinas nos modelos). Deve-se, portanto, unir a razão sistematizadora ao sentimento criativo, gerando uma intuição orientada.

A causa primeira que levou ao surgimento da ES foi o início de uma nova era no conceito de produção.

Observando a evolução do conceito de produção, Figura 2.2, podemos destacar três modos de relacionar custo (variável dependente) de um produto / sistema com a grandeza mais influente (variável independente).

Nos processos de manufatura artesanal, pré-industrial (4.000 a.C. – Séc. XVIII), o custo (total) de produzir algo era proporcional à quantidade de itens manufaturados (i.e., dominado pelo custo variável). A eficiência do processo só poderia ser incrementada a nível individual, seja pelo aumento da força daquele executando a tarefa, da velocidade com que esta era realizada ou da habilidade do indivíduo.

Figura 2.1 – Características dos hemisférios do cérebro.



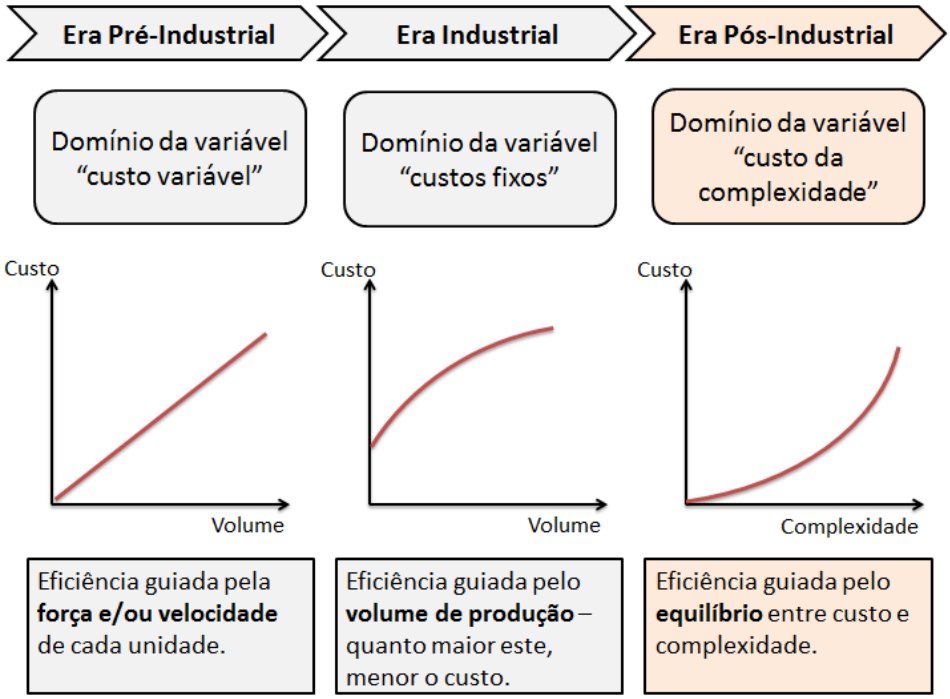
O hemisfério esquerdo é analítico, lógico, sequencial, linear e objetivo. O hemisfério direito é sintético, holístico, intuitivo, não-linear e subjetivo.
Fonte: Adaptado de Ramos et al. (2012).

A partir do Século XVIII, surge uma série de tecnologias e métodos, (máquina a vapor, petróleo, eletricidade, linha de produção em série, especialização de tarefas, entre outras) apoiada pelas recentes descobertas científicas, que possibilita multiplicar o volume de produção sem que o custo (total) acompanhe este aumento (i.e., dominado pelo custo fixo), tornando a produção em massa mais barata relativamente, com um lucro global maior. Essa lógica imperou por 2 séculos, caracterizando a Era Industrial (1750 – 1950).

Após a 2ª Guerra Mundial, em 1945,a economia começa a diversificar seus produtos, gerando enorme variedade de veículos, eletrodomésticos, aeronaves, imóveis, alimentos, medicamentos, serviços, entre outros, e inovando produtos com maior frequência, em larga medida devido ao início de um era de estabilidade econômica,

política e social nos países industrializados (EUA, Europa Ocidental e Japão). Isso começou a acarretar uma mudança profunda no modo de conceber produção, cujo custo (total) cresce com a sofisticação e as relações entre funções (dominado pela complexidade e/ou pela integração) que continua em processo nos dias atuais. Isso caracteriza a Era Pós-Industrial.

Figura 2.2 – Evolução do conceito de produção.

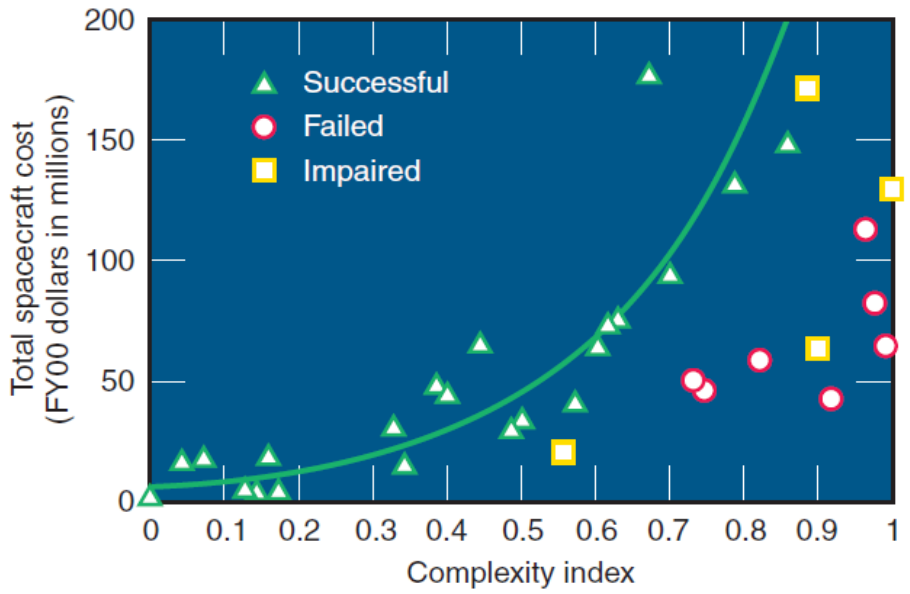


Fonte: Adaptado de Seifert (2014).

O que gera a inserção do índice de complexidade no balanço geral, como variável independente, reside em duas questões: uma ecológica e outra tecnológica. A primeira aponta que não é mais possível pensar em redução de custos de produção em função da quantidade de itens produzidos. O Sistema-Terra e o Sistema-Vida têm limites biofísicos que não podem satisfazer indefinidamente às idealizações da espécie humana, com sua indústria e serviços vinculados aos fenômenos naturais, espécies e à matéria prima. A segunda pode ser compreendida através de dados de missões diversas coletados recentemente, explicitados logo adiante.

A Figura 2.3 mostra resultados de diversas missões espaciais, com seu custo total (eixo das ordenadas) e índice de complexidade (eixo das abscissas). Esse índice foi elaborado baseado nos seguintes fatores: (a) identificação de parâmetros que guiam ou influenciam significativamente o projeto de espaçonaves; (b) quantificação de parâmetros; (c) combinação de parâmetros num índice médio de complexidade – normalizado em uma faixa de valores de zero a um.

Figura 2.3 – Custo e sucesso/falha de missões representados em função de um índice de complexidade – normalizado de zero a um – derivado de desempenho, massa, potência e escolhas tecnológicas.



Fonte: Adaptado de Bearden (2001).

As estatísticas revelam uma forte correlação entre complexidade e custo em missões bem-sucedidas. De acordo com BEARDEN (2001):

"Um limiar, ou 'zona de não voo', era aparente quando recursos de projeto (tempo, fundos) eram insuficientes frente à complexidade da empreitada."

As conclusões do estudo destacam implicitamente a incompatibilidade entre aumento de lucro (obtida em larga medida através de redução de custos) e desenvolvimento de sistemas complexos. Apesar de não ser possível afirmar que a alocação adicional de

recursos pode aumentar a probabilidade de sucesso de uma dada missão, algo é certo: quando uma missão falha ou se torna deficiente, ela provavelmente possui uma alocação de recursos incompatível àquela complexidade, acarretando, forçosamente, custos maiores.

Os dados da Figura 2.3 revelam que custos sobem exponencialmente à medida que a complexidade do sistema aumenta. Logo, se se deseja forjar um sistema de alta complexidade, é mister alocar a quantidade de recursos compatível – caso o objetivo seja sucesso.

A complexidade se traduz no aumento da interconectividade e integração dos sistemas, sejam eles, naturais ou artificiais (tecnológicos ou sociais).Enquanto os naturais são objetos exclusivamente de descobertas, os artificiais, sendo construções humanas, estão sujeitos a uma incessante melhoria em busca de uma perfeição sempre mais harmônica e eficiente.Pode-se afirmar que a complexidade é uma evolução mental do conceito que se tem de sistemas. Logo, é natural que a modelagem, com suas teorias, métodos e ferramentas, avance nesse “novo” terreno, buscando abordá-lo.

A Engenharia de Sistemas Baseada em Modelos (ESBM) faz uso da modelagem para apoiar a ES. Parte-se da formalização do desenvolvimento de sistemas – objeto original da ES – através do uso de modelos, – elemento novo – sejam estes mentais, linguísticos, físicos, matemáticos (equações, tabelas, gráficos, esquemas, etc.) ou computacionais. É uma área com grande envergadura, que abarca os múltiplos estágios do desenvolvimento do produto (concepção, identificação dos interessados, elicitação dos requisitos, desenvolvimento, integração, testes, operação, descarte) e lida com abordagens distintas (física, informacional, híbrida, etc.), domínios físicos múltiplos (mecânico, elétrico, térmico, hidráulico, etc.) e tipos (parâmetros concentrados ou distribuidos), a depender do escopo do sistema.

De acordo com o INCOSE:

> A Engenharia de Sistema Baseada em Modelos é a aplicação formal da modelagem para apoiar as atividades de requisitos de sistema, projeto, análise, verificação e validação, desde a fase de desenvolvimento conceptual até o descarte do sistema (SEIFERT et al., 2007).

Dentre algumas das aplicações da ESBM, podemos citar (BEHERE, 2015):

a) Requisitos: análise, rastreabilidade, geração de casos de teste, geração de documentos;
b) Arquitetura: descrição, estrutura, comportamento;
c) Teste: simulação em fase inicial de desenvolvimento (*Model In the Loop*– MIL), automação, otimização, parametrização;
d) Verificação e validação: cobertura, confiabilidade;
e) Outros: térmica, mecânica, montagem, fluidodinâmica.

Os benefícios decorrentes são muitos e importantes do ponto de vista econômico e social. Dentre eles podemos citar:

a) Permite desenvolver protótipos virtuais;
b) Facilita a comunicação entre áreas da instituição, e entre esta e os interessados;
c) Possibilita tanto abordagens formais quanto semiformais (aproximações inovadoras de casos formais);
d) Permite o gerenciamento da complexidade do sistema – não a sua eliminação.

Quanto ao aspecto concernente à satisfação, Seifert (2014) cita uma pesquisa em diversas empresas que lidam com sistemas complexos buscando descobrir quais são os resultados mais importantes (impactantes) que a ESBM pode gerar (Tabela 2.1). Constatou-se que a **facilidade de uso** e a **adequação aos padrões mais recentes de conformidade e execução** são os mais desejados (56%). Recomendações, apoio competente e ágil, capacidade de extensão, ambiente de colaboração dedicado e qualidade da documentação gerada, apesar de apontados, tem uma importância somada consideravelmente menor (40%). O preço tem importância marginal (4%).

Não é difícil encontrar uma explicação para isso. Um dos grandes problemas da indústria é capacitar seu corpo laboral para usar métodos e ferramentas. Quanto mais simples e intuitivos estes forem, menos erros (e custos, e possíveis processos judiciais) haverá no ciclo de vida do produto. Os padrões de conformidade e execução pesam muito devido à imposição da regulação, que deve ser cumprida para evitar perdas judiciais.

Tabela 2.1 – Fatores prioritários da ESBM das principais empresas que lidam com sistemas complexos.

#	Fator	Importância
1	Facilidade de uso	31%
2	Últimos padrões de conformidade e execução	25%
3	Recomendações de especialistas	9%
4	Apoio competente e rápido	8%
5	Ambiente de colaboração dedicado	8%
6	Capacidade de extensão	8%
7	Qualidade da documentação	7%
8	Preço competitivo	4%
Tot.	-	100%

Fonte: Adaptado de Seifert (2014).

Companhias e Instituições pesquisadas: NASA, Raykiel, Konecranes Plc, Bombardier Transportation, ABB Inc, Siemens AG, Bernafon AG, KB Medical SA, Mechatronic AG, Science and Technology Facilities Council, Sercel, etc [Jul 29, 2014].

Podem-se listar seis estágios do ciclo de vida de um sistema. Cada um deles possui um ou mais propósitos (Tabela 2.2). Cada estágio tem um propósito e deve se nortear por ele e pelos (estágios) subsequentes e precedentes – isso será melhor percebido no Capítulo 4.

Outro conceito importante a ser desenvolvido é o de verificação e validação. Segundo Vanghelue (2001), a **verificação** é o processo de comparação interna entre um código de simulação referente ao modelo de conhecimento e o próprio modelo, identificando incoerências na transformação entre um e outro tipo de modelo. Ou seja, é uma atividade que busca (mas não espera) encontrar problemas na transformação entre o modelo de conhecimento (conceitual) e o modelo de simulação (implementação), garantindo que o código / diagrama deste reflita fielmente o comportamento implícito do modelo abstrato (simbólico).

Tabela 2.2 – Estágios do ciclo de vida de um sistema com seus respectivos propósitos (TRADUZIR!).

LIFE CYCLE STAGES	PURPOSE	DECISION GATES
CONCEPT	*Identify stakeholders' needs* *Explore concepts* *Propose viable solutions*	*Decision Options* *– Execute next stage* *– Continue this stage* *– Go to a preceding stage* *– Hold project activity* *– Terminate project*
DEVELOPMENT	*Refine system requirements* *Create solution description* *Build system* *Verify and validate system*	
PRODUCTION	Produce systems Inspect and test [verify]	
UTILIZATION	*Operate system to satisfy users' needs*	
SUPPORT	*Provide sustained system capability*	
RETIREMENT	*Store, archive, or dispose of the system*	

Fonte: Hull et al. (2005).

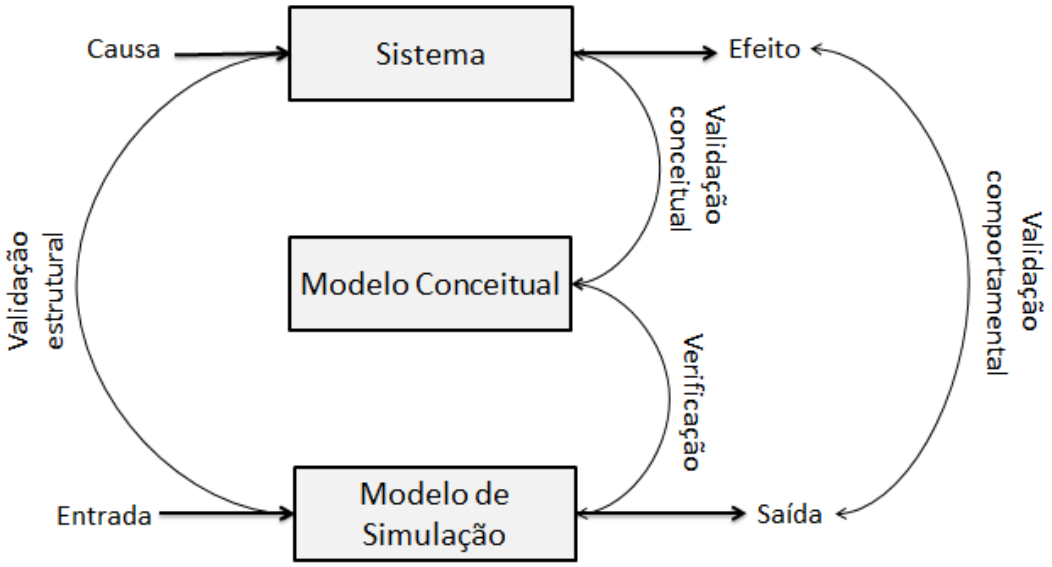
A **validação** realiza um processo de comparação externa entre a estrutura interna do modelo – seja ele conceitual ou de simulação – e seus resultados com a estrutura do sistema representado e seus comportamentos, considerando-as no mesmo contexto e condições iniciais. Podemos subdividir esse tipo de análise em três, conforme a Figura 2.4.

A **validação conceitual** avalia o modelo simbólico em relação ao sistema. O foco desse processo é avaliar o grau de realismo impresso no modelo conceitual em relação aos objetivos do estudo (ex: premissas, hipóteses, teorias, métodos), pautados pelos requisitos de sistema.

A **validação estrutural** avalia a arquitetura do modelo de simulação em relação à estrutura do sistema.

A **validação comportamental** avalia o comportamento do modelo computacional em relação ao comportamento do sistema, no mesmo contexto e com condições iniciais (dinâmico) e/ou de contorno (estática) idênticas.

Figura 2.4– Verificação e tipos de validação.



Fonte: Adaptado de Vanghelue (2001).

A Tabela 2.3 sintetiza os três tipos de validação com exemplos para uma indústria aeronáutica.

Tabela 2.3 – Tipos de validação exemplificados na indústria aeronáutica.

Tipo de Validação	**Exemplo**
Conceitual	Aeronaves voando a 10 km de altitude devem considerar em seus modelos de mecânica de vôo a variação da aceleração da gravidade com a altura, g=g(h)?
Estrutural	A tubulação hidráulica do sistema de controle das superfícies está com os parâmetros corretos? (comprimentos, diâmetros, material, forma de componentes, elementos de dobra, etc).
Comportamental	A variável temperatura dentro da cabine está acompanhando os resultados da temperatura desse ambiente na simulação? Qual o erro percentual?

Fonte: Produçãodo autor.

2.2. **Extensão da ESBM e seus Desafios**

O INCOSE indica que haverá uma expansão da ESBM para além da área técnica, englobando múltiplas esferas relacionadas direta ou indiretamente à vida humana:

> Na próxima década espera-se que a ESBM tenha um papel crescente na prática da Engenharia de Sistemas, o que irá estender seus domínios de modelagem para além de sistemas tecnológicos de hardware e software, passando por sistemas sociais, econômicos, ambientais e humanos. (INCOSE, 2007). (Tradução do autor).

De acordo com (INCOSE Vision for 2020), "o futuro da Engenharia de Sistemas será baseada em modelos, englobando modelos estáticos e dinâmicos de alta fidelidade pertencentes a diferentes níveis de abstração."(Tradução do autor).

A tendência atual da ES é passar de uma base de documentação (Engenharia de Sistemas Baseada em Documentos - ESBD) para uma base de modelos (Engenharia de Sistema Baseada em Modelos - ESBM). Inicialmente, toda a arquitetura dos modelos e os requisitos de sistema estavam arquivados e desconectados (ESBD), caracterizando um sistema estático; ao passo que nos últimos anos a computação vêm possibilitando que esses requisitos e arquitetura evoluam à medida que o projeto sofra alterações (ESBM), caracterizando um sistema dinâmico.

A linguagem de modelagem *SysML* (*System Modelling Language*) vêm se tornando a mais usual para apoiar esse tipo de abordagem.

Apesar dos avanços, implementar a ESBM vêm se deparando com desafios. O desenvolvimento de sistemas complexos faz uso de vários modelos: CAD (*Computer Aided Design*) para delimitação geométrica de projetos, Simulação 1D para comportamento da dinâmica (parte lógica e física), CAE (*Computer Aided Engineering*) para análise 3D de projetos, código de software para lógicas de funcionamento, entre outros. Dessa forma, existem diversos modos de visualizar o sistema, sendo que nenhum deles é absoluto em suas afirmações. Há a necessidade de conectá-los de forma coesa para que a especificação do sistema não sofra contradições (BAJAJ et al., 2011).

Dentre os desafios atuais da ESBM estendida ao longo de todo ciclo de vida do produto, podemos citar os seguintes pontos a serem desenvolvidos (BAJAJ, 2016):

a) Trabalhar com dados heterogêneos e descentralizados;
b) Capturar e manter uma arquitetura de sistema de alto nível;
c) Um espectro de conexões baseadas em modelos;
d) Uma plataforma unificada de conexões baseadas em modelos;
e) Da rastreabilidade ao impacto das conexões;
f) Muitos pontos de vista para uma diversidade de usuários.

Os aspectos que se pretende serem aprofundados neste trabalho se relacionam com os pontos **c** e **d**. Eis algumas observações a respeito deles:

c) Um espectro de conexões baseadas em modelos

Quando se fala em conexão entre modelos, nenhuma delas – atualmente – satisfaz todas as necessidades das equipes de engenharia de sistemas. Os trabalhos anteriores tendem a focar em dois polos: (a) conexões que apontam de um elemento de um sistema A para outro elemento de um sistema B; (b) transformações de domínio de um modelo para outro. Muitos casos de uso lidados por alguns pesquisadores são mais bem tangenciados através de uma abordagem híbrida. Nela, é criada uma conexão entre elementos equivalentes ou relacionados entre os modelos. Pode-se usar um elemento de um modelo (A) para gerar uma estrutura de elementos no outro modelo (B). Esses tipos de conexões são conhecidas como **conexões inter-modelos**.

Esse tipo de conexão permite que cada elemento participe plenamente nas fronteiras de seu domínio / modelo usando **conexões intra-modelos** (ex: dependência e relações de restrição com outros elementos no mesmo modelo), ao passo que as conexões inter-modelos podem ser usadas para comparar e sincronizar modelos à medida que o **Modelo Total do Sistema** (*Total System Model* – TSM) evolui (adaptado de BAJAJ, 2016).

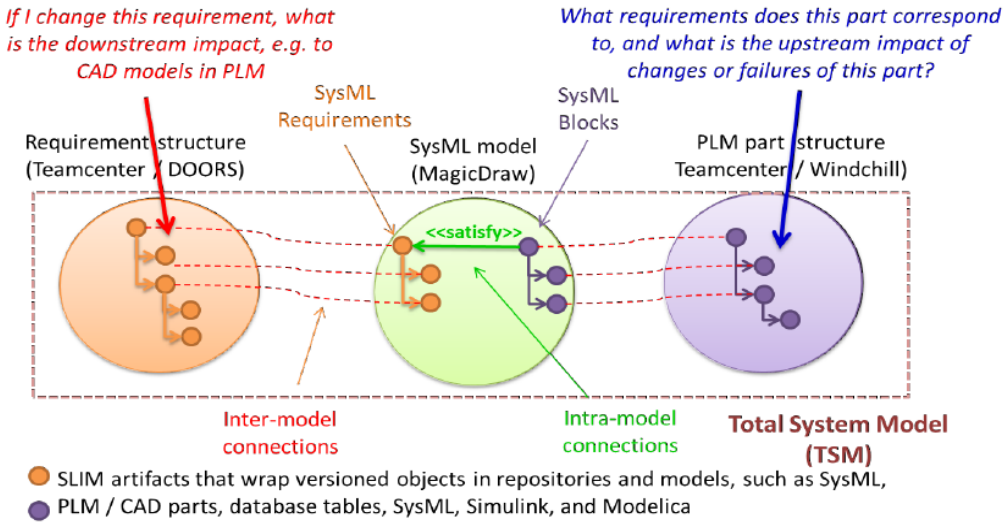
d) Uma plataforma unificada de conexões baseadas em modelos

A extensão da ESBM permite o uso do TSM para responder questões a respeito do sistema. Isso exige o uso de intra-conexões e inter-conexões. A diferença é que as

primeiras se dão entre elementos de um mesmo modelo/repositório, ao passo que as outras são entre elementos de modelos/repositórios distintos.

Um engenheiro de desenvolvimento trabalhando no ambiente de simulação física precisa acessar os requisitos a título de levantar hipóteses e especificar parâmetros de seus componentes. Dessa forma, ele deve: (1º) navegar do seu modelo de simulação 1D (ex: *AMESim*) para o modelo *SysML* (conexão inter-modelos); (2º) ir de um bloco em SysML para um requisito em *SysML* através da sua relação específica, via conexão intra-modelo e; (3º) do requisito em SysML para o requisito central com suas informações adicionais numa ferramenta de gerenciamento de requisitos (ex: DOORS), via conexão inter-modelos.

Figura 2.5 – Conexões inter-modelos e intra-modelos.



Fonte: Bajaj et al. (2016).

2.3. **Conceito de Conexão entre Modelos**

A partir do conceito de conexão entre modelos, desenvolvido pela DLR (DESHMUKH et al., 2014), pode-se melhorar a busca dos mesmos. É normal que diferentes especialidades necessitem de modelos auxiliares que satisfaçam um dado objetivo específico de seu estudo. Por exemplo: analisar o sistema de controle térmico de um satélite necessita de um modelo orbital e de atitude, de ambiente espacial, de geometria

do satélite, entre outros, que fornecem informações essenciais ao sistema de controle térmico. Com base nessas questões, a criação de bibliotecas de modelos está sendo implementada com a finalidade de melhorar a busca dos mesmos e reutilizá-los. Trata-se de uma forma de organização que deve facilitar o trabalho entre equipes de diferentes especialidades que trabalham em locais diversos. Através de uma classificação e ordenação dos modelos num repositório comum a todos, qualquer pessoa ou equipe pode realizar uma busca para importar modelos relacionados ao seu. Pode ser feita uma filtragem por fabricante, por domínio, por sistemas relacionados, pela fase de desenvolvimento ou pela plataforma de simulação. Nesse aspecto já existem agências desenvolvendo ambientes virtuais com essa finalidade.

O SimMoLib (*Simulation Model Library*) é um software desenvolvido pela DLR que provê um ambiente de teste automatizado, cujos resultados e modelos utilizados podem ser submetidos numa biblioteca, de forma catalogada. O usuário pode adicionar arquivos, documentação e dados de teste junto com os metadados do modelo, além de compartilhar esse projeto com outros desenvolvedores com o intuito de melhorar o modelo de forma colaborativa. Uma vez que ele tenha atingido certo grau de maturidade, é possível que o modelo seja lançado, isto é, passe a ser visível para todos os usuários – que tenham direitos de acesso. No entanto, percebeu-se que apenas catalogar os modelos e inseri-los numa biblioteca não é suficiente. Para melhorar o reuso é imprescindível uma busca eficiente. A partir desse desafio surge o conceito de **conexão entre modelos**.

O princípio que rege a conexão de modelos pode ser exemplificado por mecanismos de busca conhecidos, como aqueles presentes em portais de busca como o da *Amazon*, da *eBay* ou *Booking.com*. O usuário entre com algumas palavras-chave que caracterizam o produto desejado, e com base nisso são listados uma série de produtos relacionados a essas palavras. Por exemplo:“o usuário que compraram um produto também compraram este e aquele”; “clientes que buscaram esse produto também buscaram...”; entre outras opções. Esse tipo de informação provê uma maior assistência ao usuário, permitindo que seja encontrado um produto ainda mais adequado às suas necessidades específicas – se houver. Esse conceito foi importado para o *SimMoLib*. Logo, além de fazer buscas baseada em palavras-chave, é possível fazer buscas relacionadas por domínios,

aplicações ou outros tipos de relações. Isso aumenta a probabilidade de serem encontrados modelos adequados a uma aplicação específica.

Durante o desenvolvimento do conceito foram estabelecidos dois tipos de ligação: direta e indireta (DESHMUKH et al., 2014).

Ligação direta

São ligações nas quais as referências entre modelos são explícitas. Esse tipo de ligação se dá na forma 1-1 (1 para 1). Isso significa que as conexões são feitas de um único modelo (denominado atômico) para outro – e não entre 1 e vários, 1-n, ou entre dois conjuntos de modelos, n-n. Considerando esse tipo de ligação podemos dividir o estudo em duas partes: (a) na análise do tipo de ligação e; (b) na análise de como essa ligação se dá.

Os tipos de ligação que podemos ter entre os modelos são:

a) *Ligação direcionada* (*unidirecional*);
b) *Ligação não direcionada* (*bidirecional*).

No primeiro caso deve-se considerar de que ponto de vista está sendo feita a relação (ex: se o modelo A depende do modelo B, então o modelo B é pré-requisito para a execução do modelo A, mas o inverso não se aplica necessariamente). No segundo caso a relação é igual em ambos sentidos (ex: o modelo A é uma alternativa ao modelo B, da mesma forma que o modelo B é uma alternativa ao modelo A). Ou seja, a relação A→B se dá com a mesma lógica do que B→A, não importando o sentido.

Quanto à forma de se fazer essa ligação podemos fazer outra subdivisão, que leva em consideração o sentido da fusão de modelos:

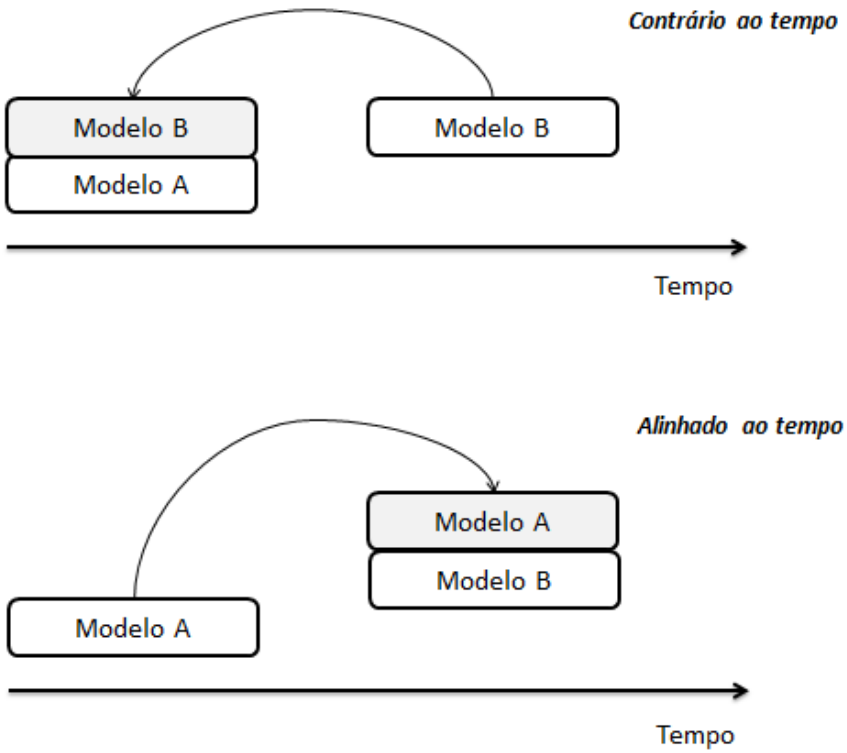
a) Direção contrária ao tempo (*retrograde direction*);
b) Direção alinhada ao tempo (*prograde direction*).

No primeiro caso (*retrograde*) o modelo novo é adicionado a um já existente. No segundo caso um modelo mais antigo é adicionado a outro concebido recentemente, como explicitado de forma visual na Figura 2.6.

Para modelos com ligações unidirecionais, o SimMoLib só realiza ligações na direção contrária ao tempo (*retrograde*). Isso se dá por motivos de consistência e de propriedade.

Para exemplificar essas duas situações, podemos esboçar a seguinte situação: alguém submete o modelo A ao repositório do SimMoLib. Após isso, alguém submete um modelo B que depende do modelo A no repositório. O desenvolvedor do modelo A não irá saber se – nem quando – o seu modelo foi incluído em outro modelo (B) a partir do ponto em que o modelo A é requisito para esse novo modelo. Apenas o desenvolvedor do modelo B sabe que ele precisa do modelo A para a execução do seu próprio.

Figura 2.6 – Formas de se adicionar modelos no tempo.



Fonte: Adaptado de Deshmukh et al. (2014).

Para exemplificar os tipos de relações citadas anteriormente, DESHMUKH et al. (2014) elaborou uma tabela (Tabela 2.4) que mostra oito tipos de relações possíveis entre modelos (A e B) e um exemplo prático. Essas relações podem ser colocadas em um diagrama que estabeleça as relações entre os modelos que serão trabalhados no

estudo proposto. Dessa forma, a biblioteca a ser criada começa a se tornar inteligente, revelando relações especiais entre modelos.

Ligação indireta

À medida que os metadados dos modelos são inseridos junto ao modelo na biblioteca, é feita uma classificação e exibição automática para os usuários através de filtros de busca. Essa ligação indireta dá uma ideia dos modelos relacionados – de alguma forma – àqueles já sendo usados. Por exemplo, todos os modelos do domínio do SCAO; ou todos que fazem transformações de unidades, ou coordenadas; ou os modelos de um fabricante particular.

As relações entre modelos foi classificada por (DESHMUKH et al., 2014). Foram identificadas oito tipos de relações fundamentais entre modelos – sejam eles estáticos, para de cálculo de parâmetros, conversão de unidade ou transformação de coordenadas; ou dinâmicos, para simulação 3D ou 1D, por abordagem física (simbologia GL) ou informacional (simbologia DB). A Tabela 2.4 elenca e exemplifica as relações entre modelos.

Na Tabela 2.4 é possível ver as relações unidirecionais (I a V) e bidirecionais (VI e VII).

Tabela 2.4 – Relações entre tipos de modelos no *SimMoLib*.

Nº	Tipo de relação	Explicação e Exemplo
I	"depende de" relação inversa: "é requisito para"	Modelo B é requisito para o modelo A; modelo A depende do modelo B. **Exemplo**: Power.mdl - Battery.mdl
II	"foi derivado de" relação inversa: "é um *template* para"	Modelo A é baseado no modelo B, mas há mudanças de interface ou o comportamento geral do modelo foi alterado. **Exemplo**: convertMJD2Date.mdl - convertJD2Date.mdl
III	"é uma extensão de" relação inversa: "foi estendido por"	Modelo A é uma extensão do modelo B. **Exemplo**: LinkBudget.xls - DataRate.xls
IV	"é uma reimplementação de" relação inversa: "foi reimplementado com"	Modelo A foi reimplementado para outra plataforma de simulação a partir do modelo B; sua função é idêntica à de B. **Exemplo**: Orbit.mdl - Orbit.xls
V	"é mais complexo que" relação inversa: "é mais simples que"	Modelo A é uma versão refinada do modelo B para fornecer um maior grau de fidelidade. **Exemplo**: MultiBodyDynamics.mdl SimpleDynamics.mdl
VI	"é uma alternativa a"	Modelo A é uma opção ao modelo B, fornecendo uma função semelhante (ex: com diferente algoritmo). **Exemplo**: MagneticFieldIGRF.mdl MagneticFieldTsyganenko05.mdl
VII	"é compatível a"	Modelo A pode ser usado conjuntamente com modelo B (ex: fazem parte do mesmo cenário e possuem interfaces que dialogam entre si). **Exemplo**: Power.xls - System Simulator.mdl

Fonte: Adaptado de Deshmukh et al. (2014).

Baseado nessas relações – e considerando o escopo deste trabalho de pesquisa – foi estabelecido um esboço do diagrama de relações entre os modelos, que será apresentado no Capítulo 3, Figura 3.7. Esse diagrama possibilita um mapeamento geral para orientar a construção da biblioteca de modelos. Esta pode ser acessada por qualquer pessoa envolvida no projeto colaborativo através da rede (cadeia de desenvolvimento) virtual. Dessa forma, um profissional que deseja saber como foi construído um modelo que possua interface(s) com o seu pode realizar buscas através não apenas de palavras-chave, mas colocando informações específicas como: autor; departamento (ao qual este pertence); sistema (satélite, aeronave, automóvel, hospital,...); subsistema ou ambiente (SCAO, SCA, Atuador,...); domínio(s) do modelo (eletrônico, hidráulico, mecânico, termodinâmico, elétrico,...); tipo de modelo (simulação, cálculos, conversão de unidade, transferência de coordenadas,...); e plataforma de simulação/execução (*MatLab*/*Simulink*, *SciLab*, MATRIXX, Excel, *AMESim*, 20-SIM, *Modelica*, Inventor, CATIA,...).

Podemos separar o diagrama de relações em três níveis: sistema, subsistema e componente. O primeiro é responsável por estabelecer o ambiente de co-simulação do conjunto (caso seja utilizado), integrando as ferramentas, fornecendo os métodos numéricos, tempo de simulação, formato dos arquivos de saída. O segundo trata dos modelos principais de cada domínio (associado ao subsistema que mais depende dele); o terceiro contém particularidades (componentes importantes) essenciais ao funcionamento dos modelos dos subsistemas. Cada modelo possui uma função e interconexões. Os modelos de projeto (cinza) alimentam o modelo de simulação de dinâmica de atitude; este, junto ao modelo do atuador e o subsistema que o alimenta, é construído a partir da abordagem por fluxos de potência (verde). Os modelos de simulação baseado na abordagem de fluxos de informação (azul) interagem diretamente com os modelos físicos e entre si.

Com a definição dos tipos de conexão entre modelos e dados, pode-se partir para os processos colaborativos multidisciplinares, que se apoiam no conceito de conexão.

2.4. **Ambiente Colaborativo: Rede e Comunicação**

A colaboração é um conceito central na ESBM. Com o progresso na ciência da computação, é possível iniciar a fazer uso de tecnologias da informação para estabelecer colaboração *ad hoc* durante o desenvolvimento de sistemas multidisciplinares e aperfeiçoá-los. A DLR vem desenvolvendo uma plataforma (*framework*) que considera esse tipo de colaboração: o Ambiente Remoto de Componentes (*Remote Component Environment* - RCE). Este é um ambiente virtual que permite a integração de ferramentas, a inserção dessas numa cadeia de produção e a execução distribuída delas a partir de um único ponto. Trata-se de um sistema distribuído. Cada instância de execução é chamado de nó na rede RCE (SEIDER et al., 2013).

Serão apresentados brevemente os conceitos de rede e comunicação do ambiente RCE, que serão implementados nos modelos computacionais em questão.

O modelo cliente-servidor, no qual uma instância central (servidor) controla todas as interações na rede, é fácil de ser implementado. No entanto, os *layouts* de rede são

estáticos – o que não é ideal para uma colaboração *ad hoc*. Dessa forma, para obter mais flexibilidade, o ambiente RCE se baseia na abordagem ponto-a-ponto (*peer-to-peer*, P2P) (SEIDER et al., 2013). Dentro desse conceito, qualquer nó do RCE pode aceitar conexões de outros nós. Uma rede P2P (do Inglês, par-a-par) se refere a uma arquitetura de redes de computadores em que cada nó da mesma funciona tanto como cliente quanto como servidor, possibilitando o compartilhamento de serviços e dados sem a mediação de um servidor central.

O ponto central é que "todos os nós que são direta ou indiretamente conectados são combinados numa rede virtual que é independente da rede física" (Adaptado de SEIDER et al., 2013).

É comum que projetos complexos tenham seus grupos de trabalho espalhados. Geralmente há uma rede de proteção (*firewall*) que inviabiliza a comunicação entre membros de localidades diferentes. O RCE contorna esse problema estabelecendo um nó (retransmissor) numa rede pública que está ao alcance das redes particulares. Dessa forma, as equipes apenas precisam estabelecer conexão com esse nó para trocar informações e executar modelos entre si. Isso acaba reduzindo problemas relacionados ao gerenciamento da rede.

As instâncias do RCE podem ser inicializadas com uma interface gráfica para usuário (*Graphical User Interface* – GUI) que provê um editor gráfico para definir cadeias de processo. Todas as ferramentas são listadas no editor e podem ser inseridas na cadeia de processo. Uma ferramenta é considerada disponível se ela está integrada a (pelo menos) um nó. Quando a cadeia de processos é executada, o RCE executa cada ferramenta no nó onde ela está integrada (SEIDER et al., 2013).

Os modelos computacionais (Simulação 1D física, Simulação 1D informacional, CAD, Otimização) podem ser desenvolvidos em ferramentas diferentes e integradas ao ambiente RCE. O objetivo é permitir que os resultados de interesse comum gerados por uma ferramenta sejam compartilhados. A finalidade do caso de estudo foi realizar uma análise paramétrica automatizada para otimizar o processo de desenvolvimento de um conjunto de subsistemas de um satélite (controle de aitiude, propulsivo e estrutura) baseado nos requisitos do veículo e as especificidades fenomênicas dos domínios

envolvidos, fazendo uso de ferramentas diversas, que representam domínios distintos, e que estão integradas de modo a diminuir distâncias (espaciais e de compreensão) entre áreas envolvidas num projeto colaborativo.

2.4.1. **Distribuição de Ferramentas *Ad-hoc***

Segundo Seider et al.(2013), para que as ferramentas sejam integradas ao RCE é necessário que alguns requisitos sejam cumpridos:

a) A ferramenta deve ser executável a partir de uma única linha de comando, e sem a interação de qualquer usuário durante a execução;
b) A entrada da ferramenta deve ser apenas arquivos e parâmetros por linha de comando;
c) Todos os arquivos de entrada devem estar localizados num diretório específico e;
d) Todos os arquivos de saída gerados pela ferramenta devem ser escritos numa pasta específica.

Toda ferramenta integrada ao RCE é tratada como uma caixa preta. Isto é, apenas suas saídas e entradas são visualizadas.

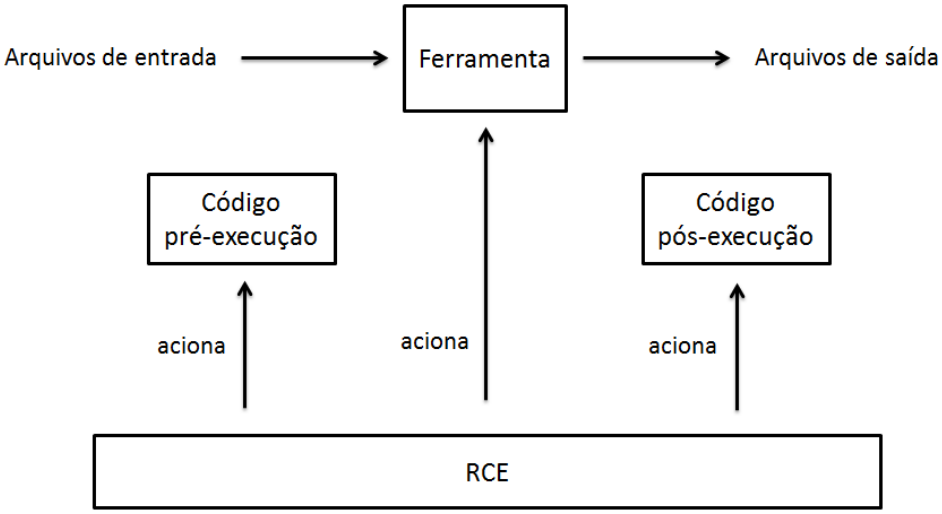
O RCE executa uma ferramenta da mesma forma que esta seria executada por conta própria. Um comando pré-definido é invocado. Após iniciar, o RCE aguarda de forma assíncrona pelo término da execução.

As ferramentas são projetadas para serem executadas autonomamente. Caso elas sejam inseridas num processo automático elas devem passar por um processo que as adeque a essa cadeia. Isso se traduz na conversão do formato dos arquivos de entrada ou escrever as saídas da ferramenta num diretório comum – compartilhado pelas outras ferramentas. Essas tarefas não precisam fazer parte da lógica da ferramenta – são externas a ela. Somente devem ser implementadas nesse contexto colaborativo. Consequentemente, o RCE pode executar processos pré e pós-execução, visto na Figura 2.7.

Para cada ferramenta integrada ao RCE, um arquivo descritivo da mesma deve ser anexado (reponsabilidade do usuário da ferramenta). Essas informações incluem:

diretórios para execução da mesma; códigos de pré e pós-processamento; o ícone a ser exibido na interface gráfica, entre outros.

Figura 2.7 – Conceito de integração de ferramenta no RCE.



Fonte: Adaptado de Seider et al. (2013).

2.4.2. **Distribuição de Métodos de Otimização *Ad-hoc***

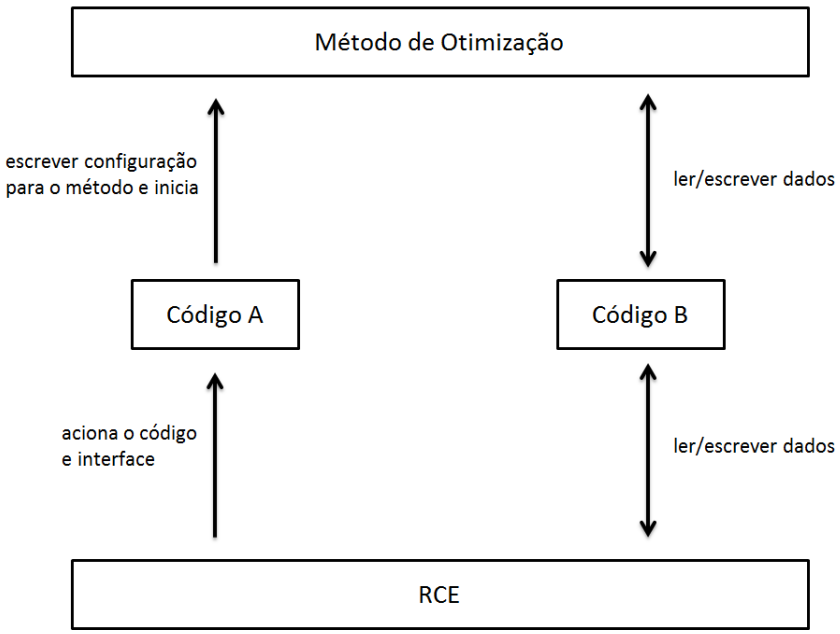
Para se melhorar o método de otimização é interessante projetar e implementar um método que se adeque ao problema. Se aquele que desenvolve o método puder implementá-lo em todas as ferramentas na qual ele será usado, maiores serão as chances de criar um método eficaz. Além disso, se diversos usuários do método puderem dar sua opinião ao criador do mesmo, menos erros tendem a surgir posteriormente na execução. Isso tende a tornar o processo de desenvolvimento mais eficiente.

O componente de integração define uma interface de código que possibilita a qualquer usuário integrar seu método de otimização ao RCE e disponibilizá-lo aos outros usuários da rede. Para isso, existe uma pasta no *template* com dois roteiros predefinidos. O primeiro deles irá ser acionado pelo RCE para iniciar o método; o segundo é responsável pelo trabalho de comunicação, como o intercâmbio de novos projetos ou de variáveis entre o RCE e o método. Os roteiros são escritos na linguagem *Python*, mas o usuário pode escrever seu método em qualquer linguagem (C++, MATLAB, Pascal,

etc.) – desde que esta possa ser acionada via *Python*. Uma ilustração desse processo, na Figura 2.8, dá uma ideia do que ocorre.

A execução do método é realizada na máquina (nó) que provê o mesmo. Isso evita o conflito de versões – algo muito comum nos projetos multidisciplinares.

Figura 2.8 – Conceito de integração de método de otimização no RCE.



Fonte: Adaptado de Seider et al. (2013).

Tendo considerado os eixos centrais desse estudo (ESBM e sua Extensão, Ambientes Colaborativos, Modelos, Paradigmas de Modelagem e Conceitos Sistêmicos), pode-se realizar um mapeamento dos eixos centrais de estudo e das contribuições pessoais dadas pelo autor.

A Tabela 2.5 ilustra qual a contribuição dada pelo autor, sendo que outros temas foram implementados, mas sem aspectos que gerem algo inédito. Na tabela são destacados os temas centrais que foram estudados e os autores principais que atuam ativamente nos respectivos campos.

Tabela 2.5– Relação dos autores principais (referências fundamentais) e temas de interesse relacionados ao trabalho (áreas centrais), com temas pesquisados (X) e contribuição dada pelo autor (LLO).

		TEMAS DE INTERESSE									
		Relações inter-modelos	Diagrama de relações inter-ferramentas dinâmico	Requisitos, sistematização e implementação de processos colaborativos	Desenvolvimento incremental de ambientes colaborativos	Paradigma orientado a serviços	Conceitos de micropropulsão	Gramática comum no desenvolvimento (Fases 0 e A)	Análise paramétrica automatizada	Otimização colaborativa multidisciplinar	Criação de ambiente colaborativo
AUTORES PRINCIPAIS	DESHMUKH et al. (2014)									X	
	ZUR, S.; TROLTZSCH, A. (2015)									X	
	SEIDER, D. et al. (2013)	X									
	SCHAUS, Volker et al. (2013)								X	X	
	NAGEL, B. et al. (2012)							X			
	WANG, H.; ZHANG. H. (2012)					X					
	HITT, Darrenn et al. (2001)						X				
	FISCHER, G. et al. (2001)				X						
	BRITTO, R. M. (2017)										X
	OLIVA, L.L. (2018)		LLO	LLO							
ITENS OGU	ORIGINALIDADE --->		c, d	a, b, d							
	GENERALIDADE --->	a		a, b	a, b, c	a		a	b	b	c
	UTILIDADE --->	a	a, b, c	a, b	a, d, e, f			a	a, f		b, f

Fonte: Produção do autor.

Cada autor – ou conjunto de autores – deu uma contribuição específica para o desenvolvimento colaborativo. No topo das colunas estão elencados os diversos temas de interesse relacionados com desenvolvimento colaborativo e com o desenvolvimento deste trabalho. A coluna à esquerda lista os autores principais que trabalham no tema de interesse. O intercepto entre cada autor / grupo e um tema constitui uma iniciação e desenvolvimento (X). Sendo assim, o autor do presente trabalho decidiu dar suas contribuições estabelecendo dois tópicos importantes a serem iniciados e desenvolvidos até certo grau.

As três linhas inferiores, abaixo dos temas de interesse e autores principais, destacam quais os aspectos de originalidade, generalidade e utilidade, da presente Tese,

emboraos outros temas também estejam presentes. Assim, tem-se um mapa geral dos eixos centrais, seus expoentes, e das contribuições dadas.

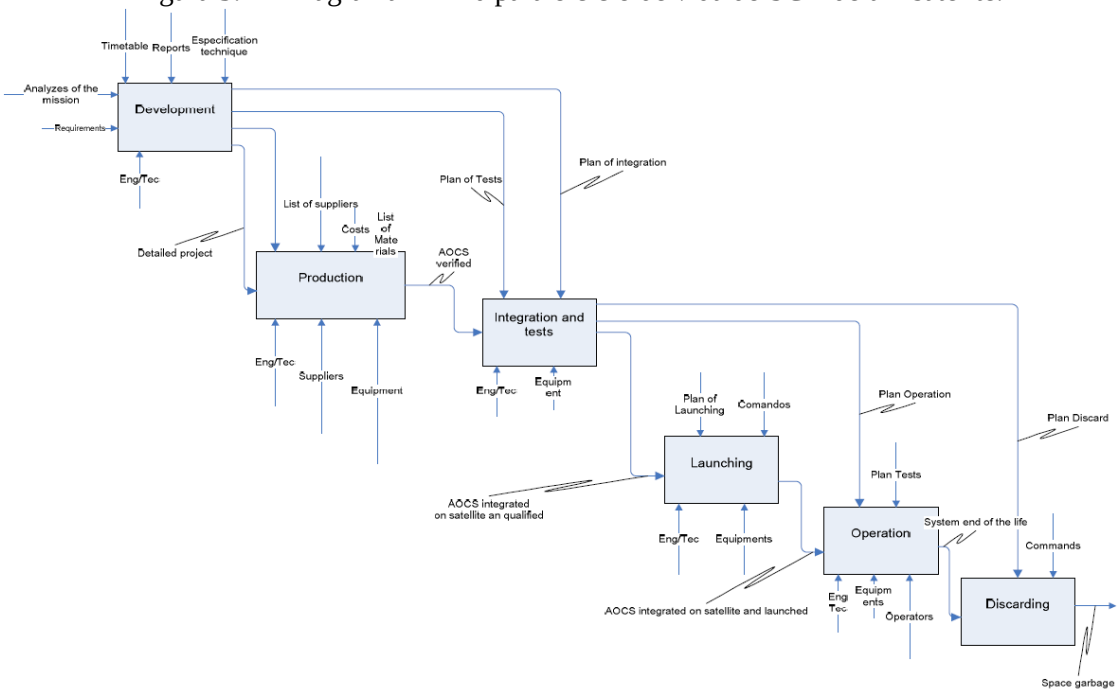
Com a revisão da literatura feita e as propostas de criações definidas, pode-se partir para a esquematização do novo processo. A partir dessa etapa, serão delineados os métodos do processo colaborativo que conduziram todo o desenvolvimento do trabalho.

3 ESQUEMATIZAÇÃO DA NOVA METODOLOGIA

3.1. **Diagrama de Processo e Atividades**

O diagrama de processo visa dar uma visão da cadeia de desenvolvimento de um subsistema. Foi utilizado o diagrama IDEF0 da ES para o ciclo de vida do SCA do satélite modelado e simulado num trabalho anterior do autor (OLIVA et al., 2010), ilustrado na Figura 3.1.

Figura 3.1 – Diagrama IDEF0 para o ciclo de vida do SCA de um satélite.



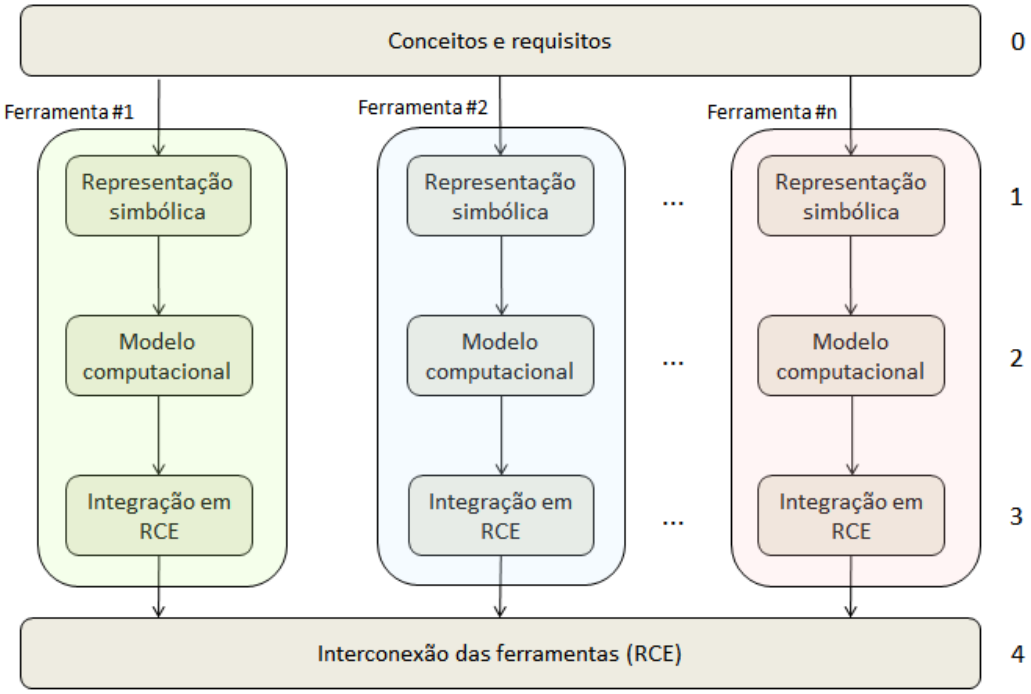
Fonte: Oliva et al. (2010).

O foco deste trabalho é o processo de desenvolvimento (o 1º bloco da cadeia). As entradas desse processo são os requisitos do subsistema e a análise da missão. Esta é constituída de um delineamento, em linhas gerais, do tipo de missão (ex: órbita, condições ambientais, tempo de operação, etc). A informação necessária (acima do bloco) consiste em: (a) um cronograma de desenvolvimento; (b) relatórios de missões relacionadas e; (c) especificações para o uso de técnicas. Técnicos e engenheiros (abaixo do bloco) serão os profissionais responsáveis pela execução dessa etapa. As saídas do processo de desenvolvimento (à direita do bloco) são: (a) o projeto detalhado

do subsistema e; (b) planos de teste e integração do mesmo no ssatélite. Enquanto o 1º irá ser a entrada para a manufatura (2º processo), o 2º será informação para realização da integração e testes (3º processo). A ERS entra na definição dos requisitos de sistema, ao passo que os métodos e ferramentas de M&S irão apoiar em larga medida o desenvolvimento do subsistema. Vê-se aí o trabalho paralelo entre ERS e EMS.

O ambiente colaborativo se insere como um coordenador geral, que irá possibilitar a **adoção, adaptação e integração** (construção de uma "orquestra") **de ferramentas computacionais** de acordo com as necessidades do sistema a ser desenvolvido. Nesse caso, o interesse é não apenas em tal construção mas também na elaboraçãoe no uso adequado de modelos, planilhas e documentos relativamente simples, que gerem resultados preliminares e rápidos, de forma a delinear em linhas gerais o desenvolvimento do produto. Esses passos podem ser sumarizados na Figura 3.2, enquanto o detalhamento do **processo proposto** neste trabalho pode ser visto na Figura 3.3.

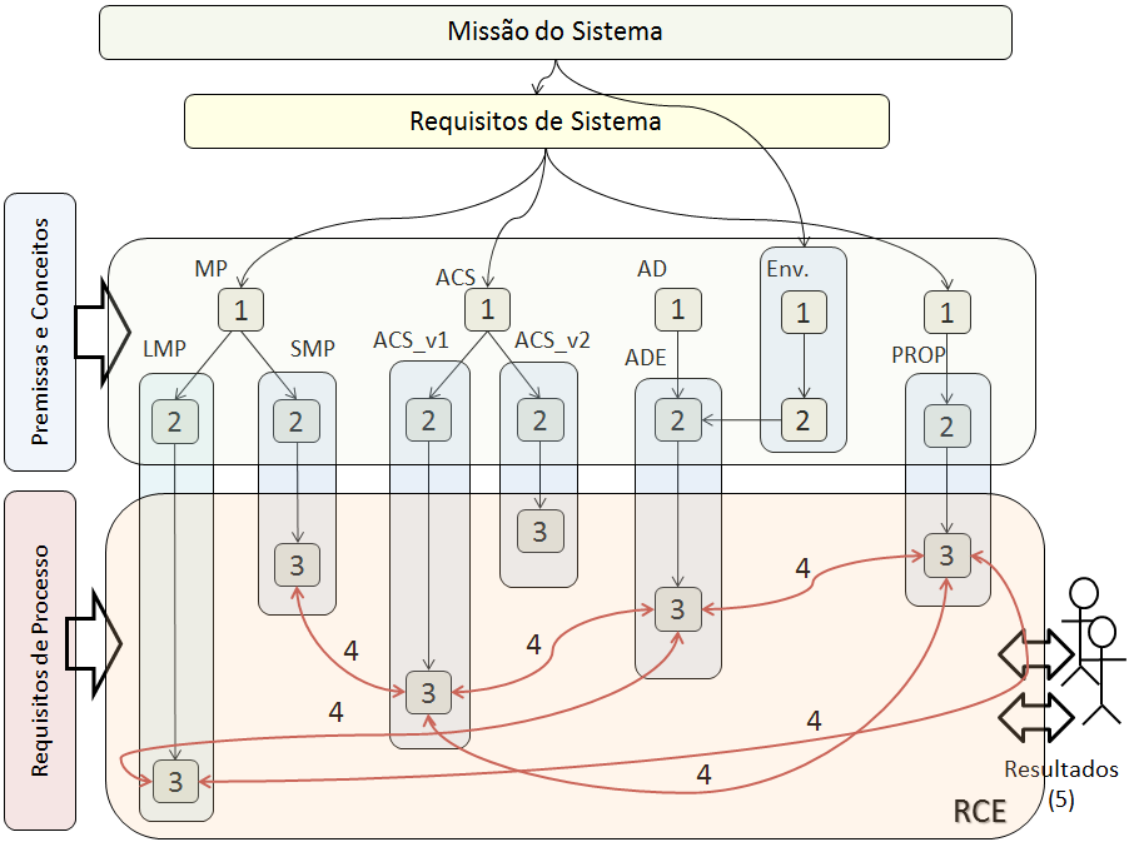
Figura 3.2 – Adoção, adaptação e integração de ferramentas computacionais a serem usadas num ambiente colaborativo geral.



Fonte: Produção do autor.

Na Figura 3.3 é ilustrado o processo proposto, a ser aplicado para o caso de estudo a ser desenvolvido. Nele observamos as etapas de desenvolvimento, partindo das premissas e conceitos, e da missão e requisitos de sistemas (e suas condições ambientais), passando pela formulação de hipóteses e resgate de teorias para a elaboração dos modelos conceituais (simbólicos) que, por sua vez, são implementados como modelos computacionais (executáveis)em diferentes ferramentas computacionais. Estas deverão ser integradas no ambiente colaborativo, se tornando parte do *menu* de opções do mesmo para qualquer equipe envolvida no projeto – que podem recorrer a elas para criar novas cadeias de processo. A partir desse ponto, com todas as ferramentas inseridas, é possível criar a **cadeia de processo**, fazendo a interconexão das ferramentas, e executando a cadeia como um todo, gerando resultados.

Figura 3.3 – Processo proposto para o desenvolvimento colaborativo.



Fonte: Produção do autor.

Na Figura 3.3, os números representam as seguintes etapas:

(1): Representação simbólica.

(2): Implementação computacional.

(3): Integração.

(4): Interconexão e interoperação.

(5): Manuseio de resultados.

Temos na Figura 3.3 os requisitos ditando as faixas e o escopo de cada modelo, e o ambiente influenciando um ou mais deles. A partir daí, a ferramenta passa por um amadurecimento, trilhando o caminho evolutivo: hipóteses → representação → implementação → verificação. Até esse ponto existe um processo de interação entre o operador humano e a ferramenta computacional. Quando a ferramenta está suficientemente maturada, ela pode ser integrada ao ambiente RCE (retângulo rosa transparente), que é uma característica de interfaceamento entre ambientes computacionais; e, após isso, interconectada a outras (setas bidirecionais vermelhas), encarnando uma cadeia de processos. A partir daí pode-se executar a rede e obter resultados.

A ideia fundamental é delinear superficialmente os requisitos de sistema, e elaborar os modelos e os requisitos do ambiente colaborativo de forma mais específica, com forte enfoque na questão de conexão, sequenciamento e temporização. O modelo irá representar o sistema num dado nível. Os requisitos irão delimitar restrições e objetivos nesse dado nível. Trata-se, portanto de uma diferenciação de funções: **representação**, **restrições**, e **finalidade**. O **modelo de otimização**, cuja natureza é **prescritiva**, é uma exceção ao caso, pois trabalha com as três funções em seu domínio. Por outro lado, o **modelo de simulação**, de natureza **descritiva**, se preocupa apenas com a representação – e consequente reprodução desta no tempo.

O escopo dos modelos irá variar à medida que se aprofundam as camadas. No topo, **modelos de uso** como “cenários dos interessados” são usados para derivar as primeiras declarações dos requisitos gerais (dos interessados). Seguindo a linha, podem ser utilizados vários **modelos funcionais** para derivar requisitos de sistema a partir dos requisitos gerais. No caso de sistemas de engenharia (controle, propulsão, energéticos,

estruturais, entre outros) podem-se usar diagramas ou *layouts*. Dos requisitos de sistema para a arquitetura (subsistemas) começa-se a preocupação com os **modelos de desempenho**.

Elaborados os requisitos gerais, inicia-se a etapa de análise do modelo a ser construído. Esse modelo visa descrever o sistema e apoiar o desenvolvimento do projeto. Sua primeira forma é a de **modelo de conhecimento (**físico-matemático-lógico**)**. Nele estão contidos os fenômenos físicos e lógicos relevantes, descritos através de equações diferenciais (modelo dinâmico) ou algébricas (modelo estático), além da enumeração das diversas hipóteses assumidas. É nesse terreno que os engenheiros de sistemas e de simulação mais interagem – é uma zona comum a ambos. Nela é possível perceber a influência dos requisitos na elaboração do modelo, e como os modelos simbólicos, físicos ou mentais, e as hipóteses assumidas, auxiliam o sistema a ser desenvolvido de acordo com requisitos.

O projeto se inicia paralelamente ao desenvolvimento do **modelo abstrato** (simbólico). No que consiste este modelo? Trata-se de uma representação simbólica (ex: em *SysML, MatLab/Simulink, AMESim, Modelica*), com delimitação de interfaces, entradas e saídas, do modelo geral (de conhecimento) elaborado na fase anterior. Nessa etapa também são definidos os parâmetros (ex: massa, dimensões geométricas, fluidos, etc) e quantidades, além dos possíveis estados do modelo, se houver.

Após isso, faz-se a elaboração do **modelo de implementação** (computacional) usando uma linguagem em um código fonte (C++, FORTRAN) ou em um ou mais ambientes de modelagem e simulação (*MatLab/Simulink, AMESim, Scilab, Modelica, Inventor*, etc.). Nessa etapa o engenheiro de modelagem e simulação está no auge de atividade, sendo responsável pela escolha dos métodos numéricos, dos intervalos de captura de dados, da escala de tempo, entre outros.

A implementação conduz à geração de resultados. Esses dados podem ser usados para a validação comportamental do modelo abstrato – ou parte dele – se o sistema já existir fisicamente – total ou parcialmente. Confrontam-se dados de teste do sistema com resultados obtidos via simulação, nas mesmas condições ambientais, iniciais e considerando os mesmos parâmetros e quantidades. Caso a faixa de erros seja tolerável

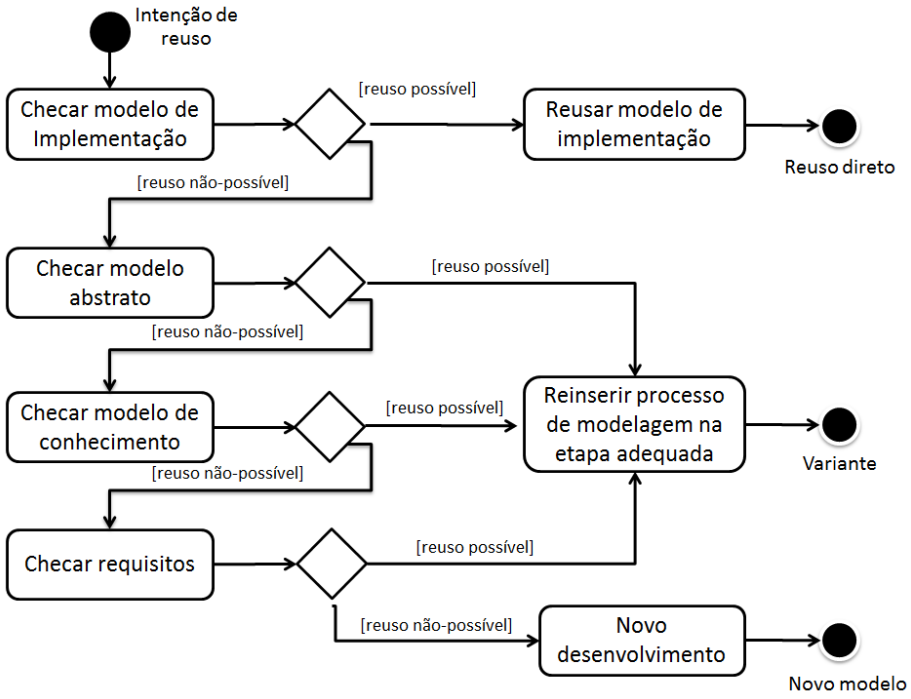
(definida a partir dos requisitos), pode-se validar o modelo, isto é, torná-lo confiável para estudar (compreender) e melhorar o sistema em desenvolvimento. E com isso gera-se um **documento de validação**.

É importante destacar a relação da Engenharia de Requisitos com todo esse processo: o modelo validado deve dirigir a construção efetiva e melhoria do sistema. Este deve satisfazer todos os requisitos iniciais. Além disso, com o passar do tempo, estes podem mudar, de forma que o modelo construído deve possuir certo grau de flexibilidade para permitir alterações em sua arquitetura interna, parâmetros e métodos numéricos de simulação, por exemplo. Dessa forma, torna-se importante ter uma **política de reuso**.

Um **Diagrama de Reuso**, na Figura 3.4, é proposto por Schaus et al.(2013) para redefinir o modelo de tal forma a mantê-lo a par com as mudanças de projeto. Ele nada mais é do que um fluxograma, que avalia o quanto é preciso retroceder nas etapas de desenvolvimento para que possamos adaptar o que foi construído.

Qual a primeira atitude a se tomar? Como se sabe, é desejável ter a menor quantidade de esforço possível para atingir um dado resultado para que o resto dos recursos seja direcionado para atividades de fins mais avançados. Logo, caso seja necessária fazer uma alteração, verifica-se o modelo implementado. Caso seja possível reutilizá-lo, basta fazer reuso direto, alterando apenas o método de simulação, o período e intervalo de tempo e as condições iniciais. Se o problema não puder ser resolvido dessa forma, parte-se para o modelo abstrato.

Figura 3.4 – Diagrama geral de reuso de um modelo.



Fonte: Adaptado de Schaus et al. (2013).

Se a geometria (dimensões, forma) de um dado componente se alterou ao longo do desenvolvimento do projeto, ou os componentes e a distribuição dos mesmos foram alterados (massa, momento de inércia), ou o material do qual é constituído/revestido foi trocado (condutividade térmica, condutividade elétrica, refletividade), ou houver um novo modo de operação do sistema, há necessidade de alterar o modelo abstrato. Nesse caso, fala-se em elaborar uma variante do modelo.

Da mesma forma, se além das mudanças citadas for necessário considerar (ou, menos usual, suprimir) outros fenômenos físicos, ou inserir uma nova lógica de controle, ou elaborar uma nova hipótese, deve-se recorrer ao modelo de conhecimento. O caso mais substancial na mudança do modelo seria a adoção de novas premissas. Mudanças mais profundas serão feitas em caso de redefinição dos requisitos, o que levará a equipe a reelaborá-los parcialmente. Todas essas situações são definidas como variantes.

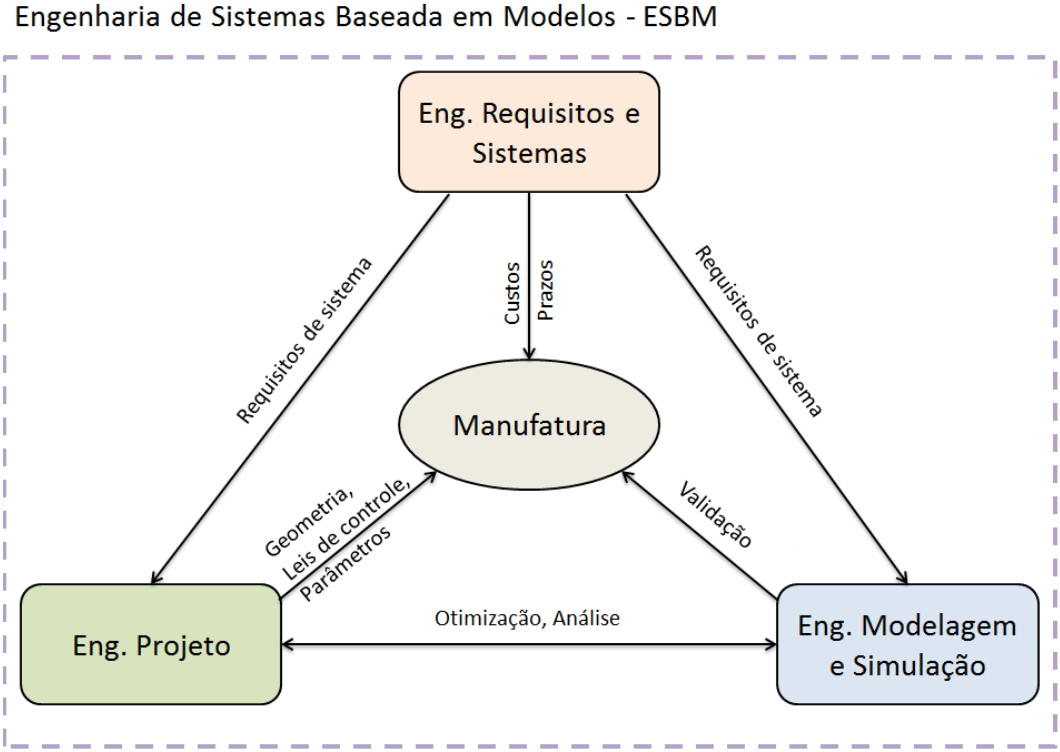
Caso não seja possível nem sequer alterar os requisitos para aproveitar uma parcela do que foi desenvolvido, deve-se partir para um novo desenvolvimento. Isso implica voltar

à definição do problema, coletando os novos interessados, suas novas necessidades, atributos, métricas, e suas novas descrições.

Para ilustrar a importância das áreas de conhecimento envolvidas no desenvolvimento deste trabalho, foi desenvolvido um diagrama que relaciona três grandes domínios: as Engenharias de Requisitos e Sistemas (ERS), a Engenharia de Projeto (EP) e as Engenharias de Modelagem e Simulação (EMS) – Figura 3.5. É importante destacar que essa interação é intensa nas fases iniciais do ciclo de vida do produto, especialmente na fase da ERS, na qual há interesse em construir modelos simples que deem resultados rápidos, gerando informações e relatórios que permitam delinear os projetos subsequentes (preliminar e detalhado), dos subsistemas e componentes.

Quando se conjuga de forma dinâmica a ERS com domínios da M&S – neste caso a EMS e EP – podemos denominar esse novo domínio, mais amplo, de ESBM, justamente por usar modelos, estáticos e/ou dinâmicos, simbólicos e/ou computacionais, para apoiar o desenvolvimento do produto. Atualmente esse apoio se encontra em fase de amadurecimento: muitos institutos, como o DLR, estão criando ferramentas computacionais e aprimorando metodologias com o intuito de criar uma biblioteca inteligente acessível a todas as áreas direta ou indiretamente envolvidos num projeto, seja qual for o distanciamento espacial entre as equipes (SCHAUS et al., 2013).

Figura 3.5 – Relações entre Engenharia de Sistemas, de Projeto e de Modelagem e Simulação.



Fonte: Produção do autor.

As definições de cada domínio podem ser sintetizadas da seguinte forma:

a) A ERS trata dos métodos de gerenciamento do ciclo de vida do produto, das necessidades (problemas) que conduzirão futuros desenvolvimentos (soluções funcionais), e da verificação e validação do sistema;

b) A EP está interessada em construir os subsistemas e componentes a serem manufaturados (soluções físicas), e que comumente usam a M&S para representar computacionalmente o produto sendo projetado, de forma a delimitar sua geometria e outros parâmetros. Isso dá uma visão da parte física do sistema para os usuários, e ao mesmo tempo permite que a EMS conduza o desenvolvimento de seus modelos, se orientando pela geometria para assumir hipóteses e escolher que modelo de conhecimento adotar, além das ferramentas e abordagens relevantes;

c) A EMS é uma área cujo interesse reside em reproduzir o comportamento do sistema no tempo, obtendo saídas que dão uma ideia geral de como o sistema irá

se comportar em uma série de situações, seja ela produção, operação, descarte, etc. Seu foco é o apoio às atividades da ERS e da EP.

As relações entre as três áreas e o produto podem ser sintetizadas em:

a) A ERS irá fornecer requisitos de sistema para a EP e EMS;

b) A EP define a geometria e parâmetros físicos do produto;

c) A EMS auxilia na EP e pode definir os tipos de subsistemas (S/S) e seus componentes, define parâmetros e adota lógicas adequadas ao sistema;

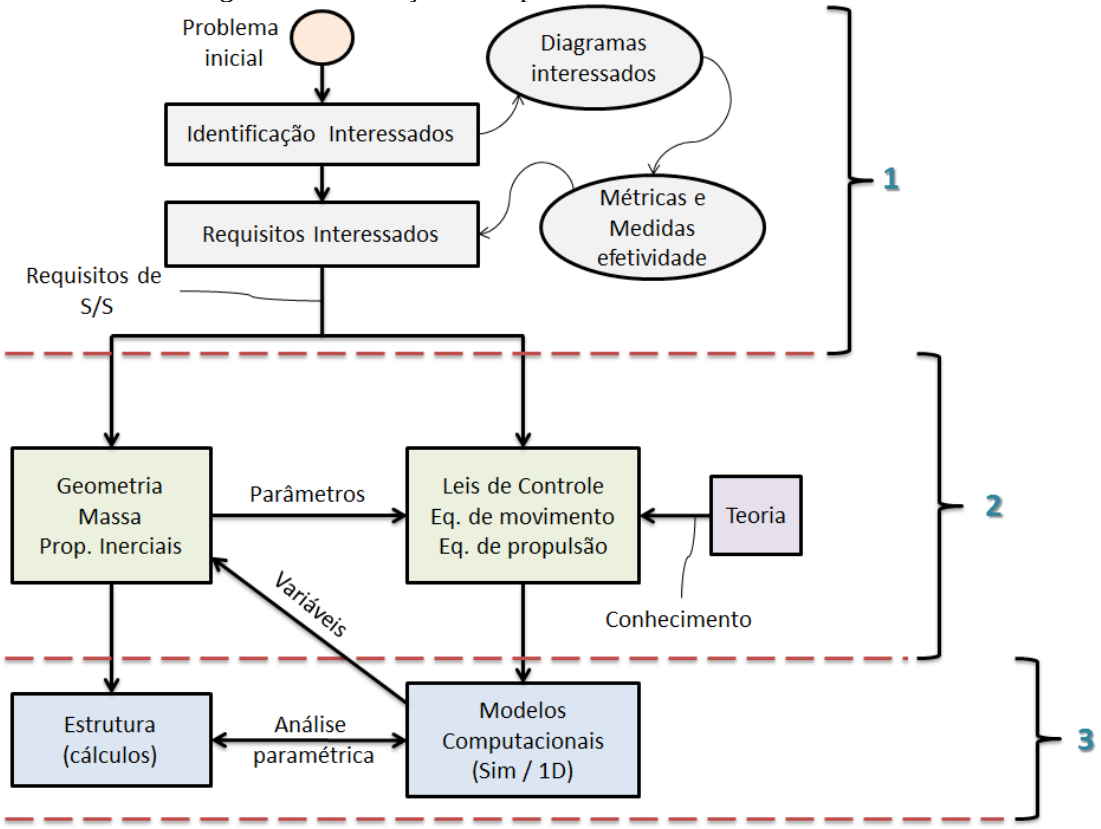
d) Entre a EMS e a EP existe, em geral, a necessidade de aperfeiçoar parâmetros – tanto dos modelos de projeto quanto daqueles de simulação;

e) A ERS fornece o custo total e tempo de desenvolvimento (esperado e desejado) do sistema;

f) A EMS pode fazer uso de diversos modelos, desenvolvidos em ferramentas diferentes (especialmente se as abordagens forem diferentes e considerando que os engenheiros geralmente estão habituados a trabalhar com uma ferramenta e mentalidade específica para realizar o seu trabalho) que geralmente são integrados entre si no sistema real. Dessa forma, a co-simulação é uma prática comum que enriquece a comunicação entre as áreas.

A Figura 3.6 mostra pormenorizadamente as relações e etapas do desenvolvimento do presente trabalho, concatenando as três grandes fases a serem atravessadas: métodos da ES (1); construções simbólicas (2) e; métodos e implementações colaborativas (3).

Figura 3.6 – Relações e etapas do desenvolvimento da Tese.



Fonte: Produção do autor.

Trata-se de um processo de desenvolvimento referente às fases iniciais (0 e A) de um modelo funcional para responder a um dado problema. No lado direito da Figura 3.6 pode-se ver três regiões – a próxima sempre englobando a(s) precedente(s). Na primeira, tem-se a presença da ER, que podemos definir como a primeira etapa de preparação (indireta) do processo colaborativo. Na segunda, tem-se o levantamento de hipóteses e formulação dos modelos simbólicos, apoiado pela teoria. Nessa etapa existe uma interação forte entre a ER e a ES. Na terceira, existe a implementação desses modelos funcionais em ferramentas específicas. Aí provavelmente há a presença da EMS, que interage com a ES, ao passo que a ER deixa de ter um papel relevante.

A metodologia proposta trabalha com a proposta de se fazer uma inteligente ordenação dos elementos da parte 3. É nessa etapa que tem-se a execução do processo, sendo que elementos novos são adicionados a essa parte, dentre os quais podem-se citar os requisitos de ambiente (BRITTO, 2017) e de processo (nesse estudo) e diagramas de relações dinâmicos inter-ferramentas.

3.2. **Diagrama de Relações**

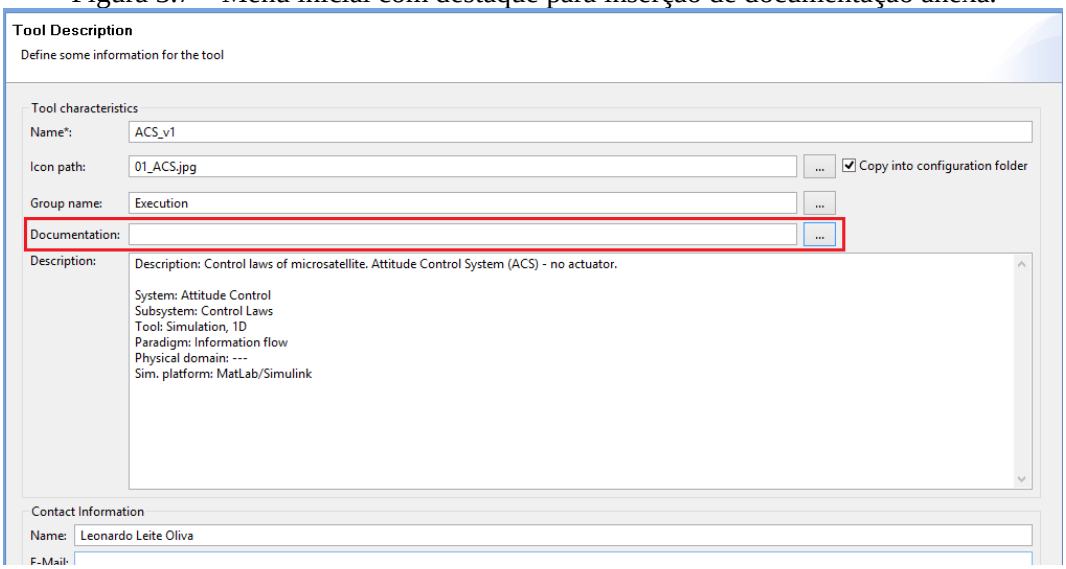
A atividade “mapear” serve para localizar algo de forma inteligente. Buscou-se construir, de forma embrional, uma classificação de modelos que facilite o mapeamento dos mesmos, estabelecendo relações baseada numa tabela de relações conforme apresentada anteriormente. Cada ferramenta receberá uma denominação que deve permitir que o sujeito insira ou não a ferramenta e sua documentação em seu projeto, como mostrado na Figura 3.7.

O **Mapa de Relações** pode (e deve) ser inserido na documentação de cada ferramenta. Trata-se de um diagrama que estabelece o(s) tipo(s) de conexão(ões) existente(s) entre as diversas ferramentas integradas ao ambiente colaborativo, mostrado na Figura 3.8.

Um requisito – esboçado pelo autor – pode ser sugerido aqui: *“Toda ferramenta integrada ao ambiente colaborativo deve conter em sua seção de documentação um diagrama comum (a todas as outras já integradas) chamado mapa de relações acompanhado de uma tabela explicativa. Nele, as partes constituintes da ferramenta são simbolizadas por um círculo numerado e as ferramentas são delimitadas por regiões. As relações são setas uni ou bidirecionais que devem conter uma sigla especificando o tipo de relação.”*

Toda equipe envolvida no projeto terá **acesso inicial** à mesma documentação (arquivo em formato .pdf). Qualquer alteração de modelos, de roteiros de cálculo, de conversão de unidades, ou de manuseio de arquivos deve ser informada a todos os nós da rede, que devem trocar o diagrama antigo pelo novo. Uma equipe coordenadora deve ser responsável pela **atualização** e **divulgação** do novo mapa de relações. Ter-se-á nesse caso adição ou remoção de elementos e uma subsequente reconfiguração de relações (setas) entre os novos elementos e aqueles relacionados diretamente a eles.

Figura 3.7 – Menu inicial com destaque para inserção de documentação anexa.



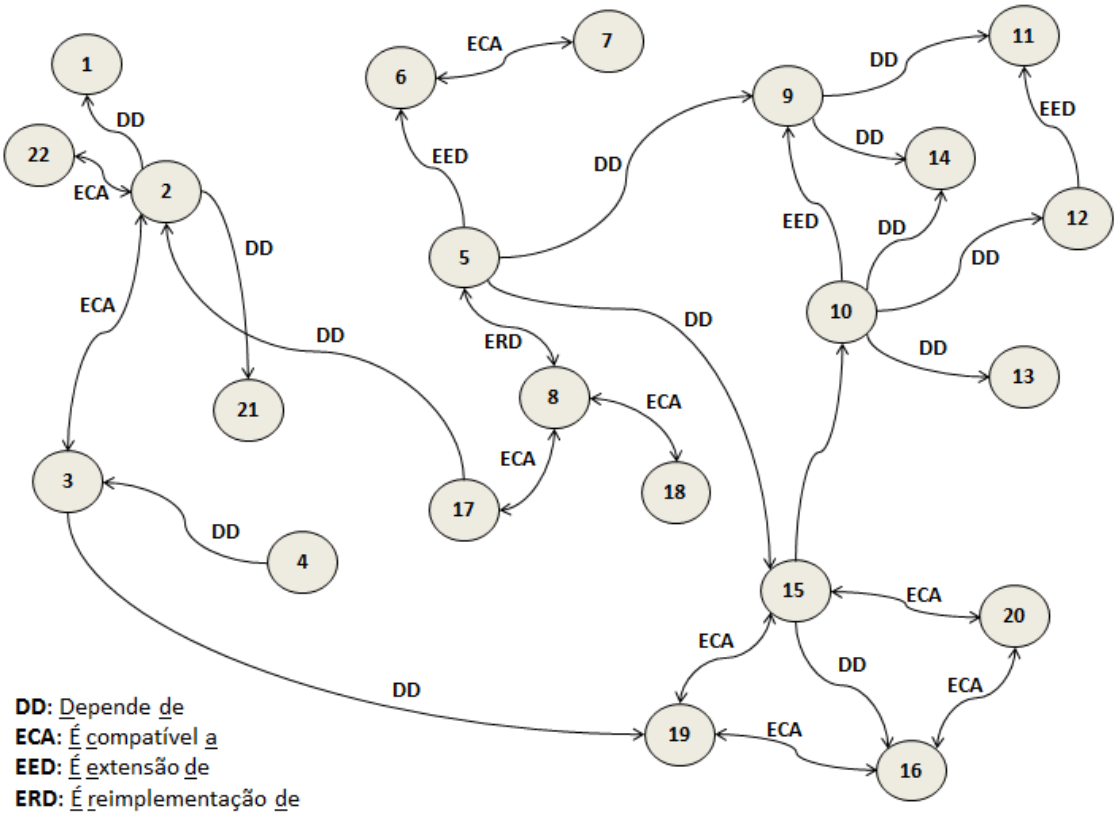
Fonte: Produção do autor.

O diagrama construído contemplou todos os modelos, códigos e arquivos envolvidos, além de outros que poderiam estar nas cadeias de processo desenvolvidas. A Figura 3.8 mostra o diagrama de interconexão inter-ferramentas, no qual as mesmas são elementos numerados, e as conexões (setas uni ou bidirecionais) podem ser de vários tipos. Esse diagrama não é estático: através de transformações pode-se mudar o esquema geral de relações.

É importante destacar que arquivos são tanto constituintes dos elementos quanto podem fluir através deles. Geralmente os elementos de entrada (*inputs*) que alimentam uma ou mais ferramentas contém os arquivos próprios da ferramenta (execução e dados iniciais). A ligação entre ferramentas contém os arquivos de saída de uma (*outputs*), que são pré-requisitos para execução de outra.

É igualmente importante destacar que se deve utilizar um *software* adequado para a construção desse mapa. Ele deve possuir uma maleabilidade de edição, facilitando mudanças no mesmo. Outro ponto importante é garantir que todos os nós envolvidos na cadeia de processo tenham acesso a esse mapa atualizado, mas não possam editá-lo – para melhor organização.

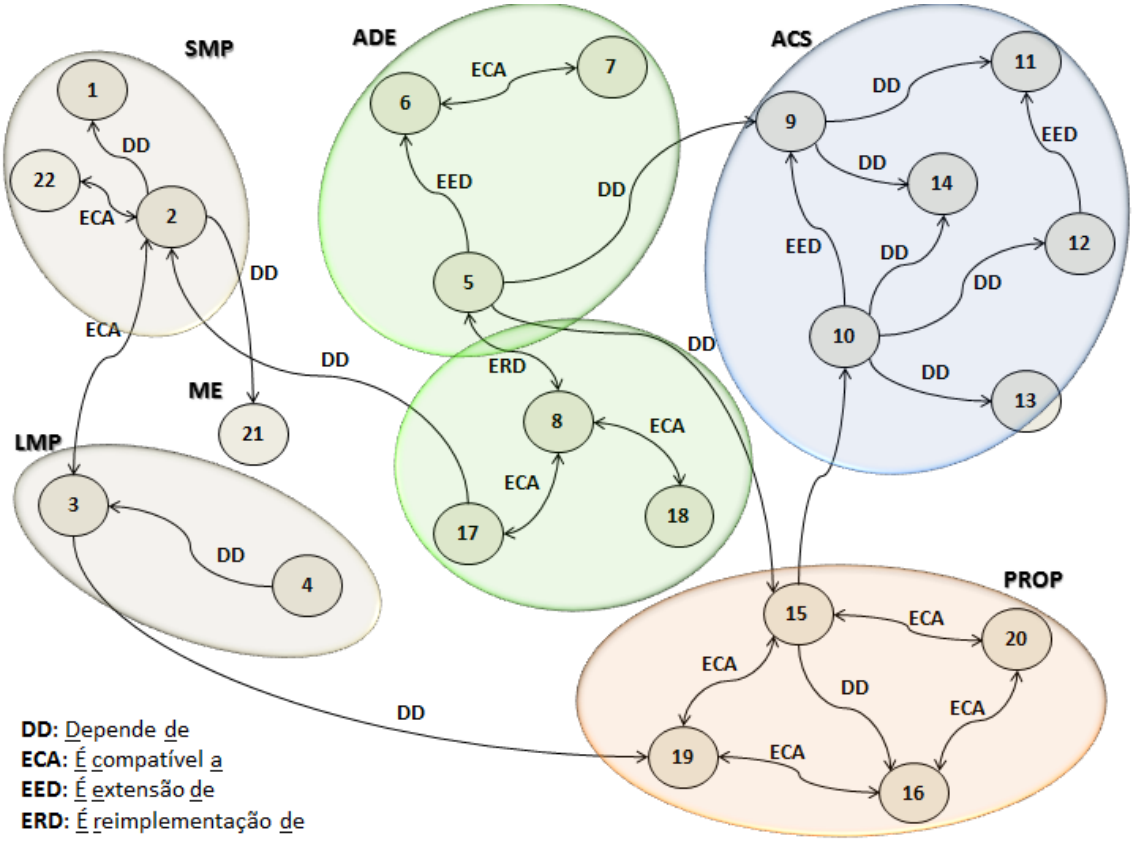
Figura 3.8 – Mapa de relações desenvolvido para o presente estudo.



Fonte: Produção do autor.

Para facilitar a visualização das disciplinas (ou equipes) que desenvolvem / operam / atualizam conjuntos de ferramentas, foram destacadas as regiões (Figura 3.9). A Tabela 3.1 revela quais ferramentas fazem parte de cada disciplina, quais seus nomes e terminações (plataforma de simulação).

Figura 3.9 – Mapa de relações desenvolvido para o presente estudo com destaque para as ferramentas.



Fonte: Produção do autor.

Tabela 3.1 – Elementos contidos em cada ferramenta das Figuras 3.8 e 3.9 com seus formatos.

SMP	1	*Geometry_and_material_param.m*	AUX.	18	*ADE_load*
	2	*Input_mass_prop.m*		19	*PROP_param.m*
LMP	3	*Liquid_properties.m*		20	*PROP_load*
	4	*LMP.mdl*	ME	21	*Mission_and_environment.m*
ADE	5	*Attitude_dynamic_and_env.ame*	SMP	22	*Input_mass_prop.sce*
	6	*Attitude_dynamic.ame*			
	7	*Environment.ame*			
	8	*Attitude_dynamic_and_env.mdl*			
ACS	9	*Gains_K_satellite.m*			
	10	*Gains_K_satellite_and_actuator.m*			
	11	*Control_satellite.mdl*			
	12	*Control_satellite_and_actuator.mdl*			
	13	*Actuator_param.m*			
	14	*Reference.m*			
PROP	15	*Propulsion_sys.ame*			
	16	*Thruster.ame*			
AUX.	17	*ADE_param.m*			

Fonte: Produção do autor.

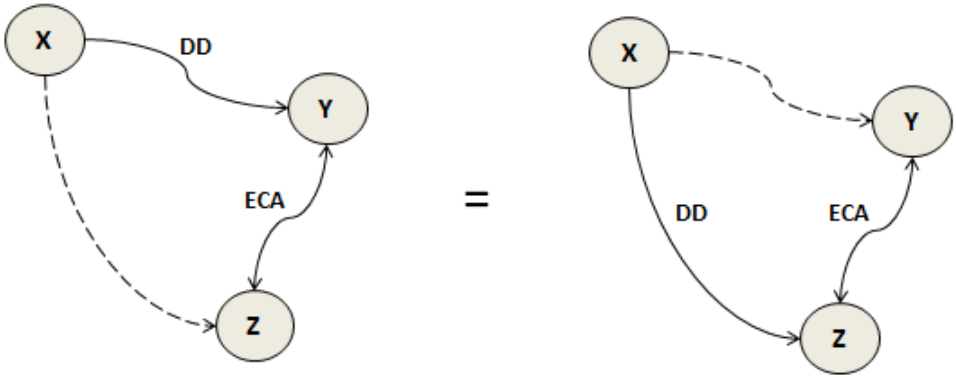
Algumas observações podem ser feitas quanto às relações presentes nos diagramas das Figuras 3.8 e 3.9. A primeira delas consiste em observar a fluidez dessas relações.

Vamos supor a existência de três ferramentas (x, y e z) inter-relacionadas da seguinte forma:

a) X depende de Y;
b) Y (ou Z) é compatível a Z (ou Y).

O diagrama dessa relação pode ser visualizado na Figura 3.10. Em termos mais formais, pode-se afirmar: “Se X depende de Y e Y é compatível a Z, então X pode depender de Z.”

Figura 3.10 – Primeiro exemplo de equivalência entre relações.

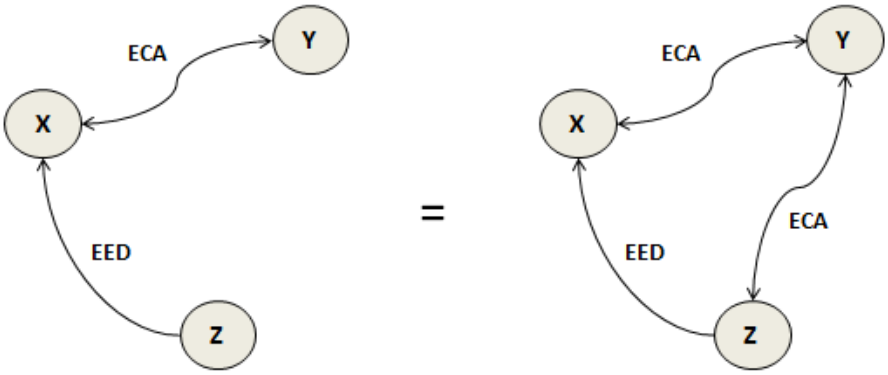


Fonte: Produção do autor.

Uma segunda transformação do diagrama pode ser feita em relação à compatibilidade (ECA) e extensão (EED).

Tomando como exemplo as mesmas ferramentas genéricas X, Y e Z com relações de EED (entre X e Y) e ECA (entre Y e Z), pode-se fazer uma segunda alteração de forma sem alterar a natureza do diagrama, na Figura 3.11.

Figura 3.11 – Segundo exemplo de equivalência entre relações.

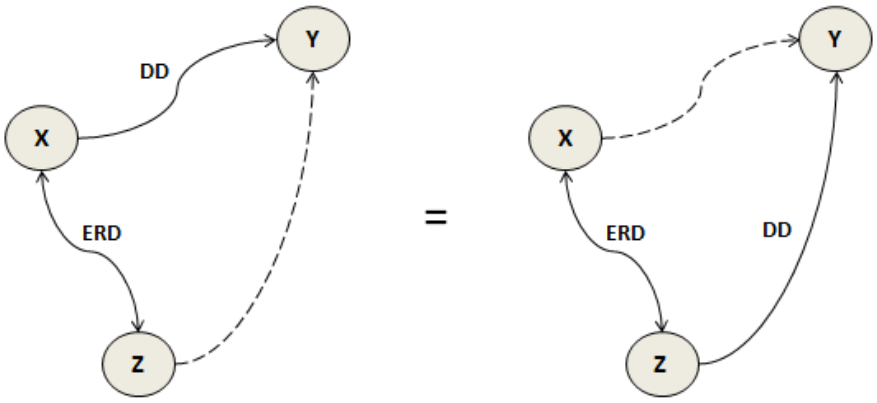


Fonte: Produção do autor.

Em termos mais sistemáticos, pode-se afirmar: “Se Z é extensão de X, e X é compatível a Y, então Z também é compatível a Y.”

Há ainda um terceiro caso de interesse – envolvendo as relações DD e ERD, na Figura 3.12.

Figura 3.12 – Terceiro exemplo de equivalência entre relações.



Fonte: Produção do autor.

Nesse caso percebe-se que se pode usar tanto X quanto Z para construir uma cadeia de processo, e nesse caso é necessário explicitar a nova dependência – que tanto pode ser entre X e Y ou entre Z e Y.

Com base nas equivalências apresentadas, percebe-se que os Mapas de Relações das Figuras 3.8 e 3.9 possuem uma fluidez. É preciso que os desenvolvedores saibam lê-los de modo a tirar proveito dos mesmos. As relações não são fixas: elas podem alterar dentro de uma faixa de possibilidades. A combinação e natureza dessas relações irão determinar as leis que permitem a intercambialidade entre os elementos, apresentando alternativas de desenvolvimento de um processo.

Poder-se-á, através de um aplicativo propício (ex: Visio), colocar os diagramas das Figuras 3.8 e 3.9 de modo a serem editáveis apenas por um só nó central, cujo foco é manter uma só base de dados. Qualquer nó na rede que deseje integrar outra ferramenta deve se reportar para esse nó central, que, por sua vez, irá inserir o novo elemento na Tabela 3.1 e no diagrama. A partir das descrições da ferramenta, quem edita pode estabelecer novos tipos de relações e carregar o novo diagrama, atualizado, disponível para todos os nós da rede, que poderão utilizar os modelos conforme sua conveniência.

Uma política de reuso deve se basear num critério. Sempre que se inicia um novo projeto é necessário levantar novos requisitos de sistema. Muitos componentes novos provavelmente terão modelos já construídos. Para observar se é mais vantajoso a construção de um novo modelo a partir do zero ou se é mais viável fazer uso de um

modelo utilizado em projetos anteriores, deve-se, antes de tudo, fazer uma busca inteligente. Essa busca (de modelos) deve se basear no diagrama construído anteriormente, nas Figuras 3.8 e 3.9, que classificam modelos (ou ferramentas de cálculo ou de manuseio de arquivos) conforme certos princípios. É necessário haver um banco de dados disponível a todos os nós da rede. Isso pode ser feito fazendo-se uso da ferramenta *Database* do RCE.

A política de reuso visa aperfeiçoar o processo de desenvolvimento, eliminando a necessidade de construir modelos (ou roteiros de cálculo, de conversão de unidades, manuseio de arquivos, etc.) a partir do início, com estabelecimento de premissas.

O primeiro passo para saber se é ou não necessário elaborar novos modelos é: (a) saber o tipo de sistema a ser desenvolvido; (b) quais ferramentas já estão disponíveis – considerando-se todas as equipes de todos institutos envolvidos no projeto; (c) qual diagrama relaciona todas essas ferramentas. A partir daí, pode-se partir para:

O segundo passo consiste em mapear quais ferramentas devem ser utilizadas.

O terceiro passo consiste em elaborar um diagrama que mapeie o processo de desenvolvimento-integração-interconexão das ferramentas, destacando as interações máquina-máquina (M-M), humano-máquina (H-M) e humano-humano (H-H), conforme exibido na Figura 3.3.

O quarto passo é ver se existe alguma ferramenta do segundo passo que não está no diagrama de relações da empresa. Caso não esteja, deve-se desenvolvê-la.

Para aquelas que estão disponíveis, pode-se averiguar um modo mais adequado de relacioná-las conforme as necessidades e facilidades das equipes. Nesse ponto, podem ser usadas as transformações das Figuras 3.10 a 3.12. Ter-se-ão assim múltiplos diagramas, cada um revelando um arranjo entre ferramentas.

3.3. **Identificação dos Interessados**

O processo de desenvolvimento da Figura 3.3 guiado pelo diagrama apresentado na Figura 3.6 mostra que a primeira etapa consiste em identificar possíveis interessados no

desenvolvimento do sistema (neste caso o de controle de atitude de um satélite). O problema inicial já foi apresentado: **conceber um sistema através da criação de um novo processo, com possíveis adaptações que visem facilitar a compreensão de diversas configurações de parâmetros (que respeitem as restrições impostas pelos requisitos) de um sistema usando múltiplas ferramentas integradas num ambiente colaborativo**. Aqui é dado o primeiro passo, com a identificação dos interessados nesse sistema.

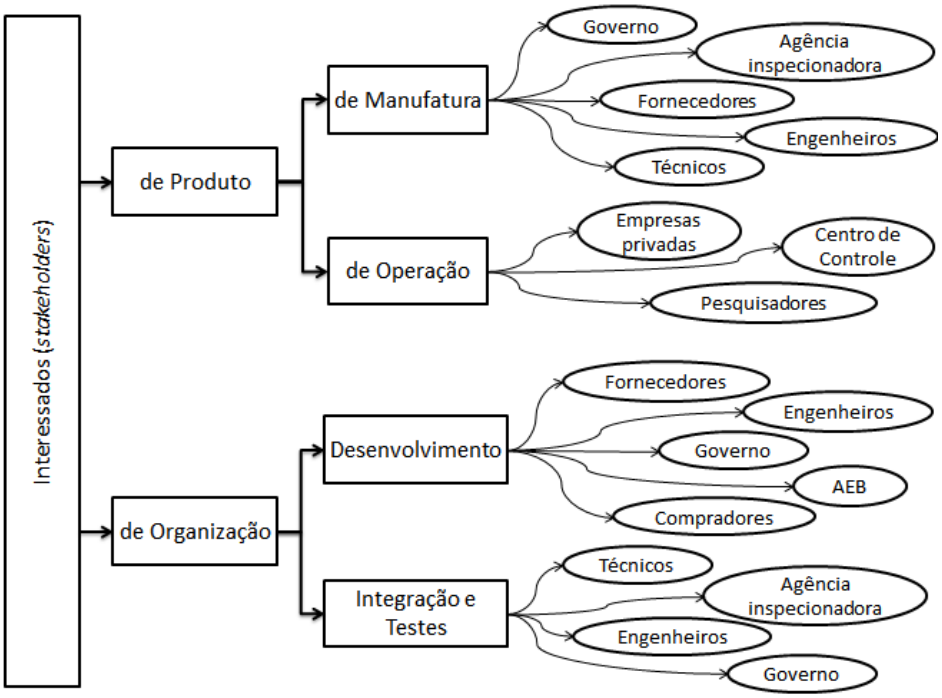
Para fazer um levantamento dos interessados deve-se partir do princípio de que exista algo e/ou alguém interessado no método a ser desenvolvido, seja por motivos econômicos, políticos, culturais, científicos ou técnicos. Esses motivos podem estar – e estão – relacionados. Mas essa relação possui uma ordenação, isto é, uma centralidade, de modo que existe uma hierarquia a ser seguida. Em situações críticas essa lógica hierárquica fica evidente, se revelando através das atitudes dos interessados.

Como o escopo deste trabalho é desenvolver um processo colaborativo e melhorar o projeto de um subsistema específico através de potencialidades e possibilidades de processo, os requisitos levantados aqui foram restritos ao subsistema em questão e outros diretamente relacionados (dimensões, massa e controle de atitude).

O levantamento dos interessados foi baseado em trabalho apresentado por (OLIVA, 2012). Eles podem ser divididos em dois tipos: de produto e de organização. Cada qual por sua vez se subdivide em dois: os de produto em processos de operação e de produção; e o de organização em processos de desenvolvimento e de integração e testes, visível na Figura 3.13.

Nesse estudo foram considerados apenas os interessados no processo de **desenvolvimento** (fornecedores, engenheiros, governo, AEB e compradores), cada qual com seu interesse específico – que pode coincidir com os de outros interessados, inclusive em outros ramos (ex: engenheiros têm interesses tanto no processo de desenvolvimento quanto de manufatura).

Figura 3.13 – Diagrama dos interessados no SCA.



Fonte: Produção do autor.

A partir do conjunto de interessados, serão estabelecidos os atributos e suas métricas (ou medidas de efetividade, do Inglês *Measures of Effectiveness-MOEs*) que servirão de guia condutor para a elaboração dos requisitos dos subsistemas em questão. Antes de adentrar esta elaboração, será apresentado brevemente um satélite de porte convencional, cujo subsistema propulsivo já foi validado em partes num trabalho anterior (OLIVA, 2012).

3.4. **Atributos e suas Métricas**

Conforme dito anteriormente, serão desdobradas apenas os atributos e suas métricasdos interessados no **desenvolvimento do sistema** (fornecedores de ferramentas, compradores, engenheiros de sistemas, projeto e simulação, governo como um todo e a AEB). Esses atributos e suas métricas irão balizar a obtenção dos requisitos de sistema. A Tabela 3.2 estabelece as conexões entre os interessados, os atributos (ou interesses de cada interessado) e suas métricas. Serão estas últimas que irão delinear em linhas gerais a elaboração dos requisitos do subsistema de interesse (micropropulsão).

Pode-se perguntar o que levou a escolha dos interessados. Isso pode ser explicado se fizermos a seguinte pergunta: *quais os principais profissionais e instituições envolvidas no desenvolvimento de um subsistema, desde a sua concepção até sua operação e descarte?* A partir daí chega-se aos sujeitos principais, diretamente interessados.

Tabela 3.2 – Atributos e suas métricas de cada interessado no desenvolvimento do subsistema em questão.

INTERESSADOS - DESENVOLVIMENTO DO PRODUTO (SCA)			
Interessados	**Interesses**	**Métricas**	**Medidas**
Fornecedores (indústria em geral)	Demanda por produtos que sejam já consolidados na indústria.	Demanda por serviços ou bens (métodos, ferramentas, simulaçõe, análise, testes, integração, manufatura, etc).	Quantidade de serviços demandados por período.
Engenheiros (sistemas, simulação e projeto)	Métodos de modelagem práticos. Ferramentas de simulação boas. Infraestrutura (laboratórios para testes). Comunicação entre áreas rápida e eficaz.	Tempo de desenvolvimento geral; Reuso dos modelos; Técnicas de mapeamento.	Tempo de desenvolvimento) de componentes e subsistemas; Quantidade de modelos (simbólicos / de simulação) reusados; Tempo dispendido para realização de testes. Respostas do subsistema (empuxo, vazão, tempo de estabilização, *overshoot,* velocidades angulares, etc); Propriedades inerciais.
Governo	Inovação tecnológica. Aplicação adequada dos recursos. Domínio de tecnologias estratégicas.	Quantidade de produtos considerados inovativos; Relação entre investimentos e retorno a médio e longo prazo (comparado a outras nações); Plano de execução.	Relação global custo-benefício (quanto foi investido em todo processo versus quais os ganhos econômicos do novo produto e metodologia).
AEB	Inovação tecnológica.	Produtos considerados inovativos.	Grau de impacto do novo método se aplicado em outros processos de desenvolvimento (ex: satisfação dos engenheiros com o processo colaborativo em termos de ganho de tempo, facilidade de mapeamento de subsistemas, comunicação entre diferentes áreas, melhoria de características de projeto, etc).
Compradores (governo, empresas privadas, empresas estatais)	Adaptação de tecnologias (métodos, ferramentas, componentes,etc) em seu nicho de mercado.		Relação monetária custo-benefício.

Fonte: Produção do autor.

É interessante ter um quadro de funcionários capazes de realizar medições de cada uma dos atributos ao longo do tempo. Para alguns casos, como a medição será de longo prazo (Governo e AEB), basta o registro de dados brutos que, posteriormente, poderão ser organizados por um especialista que fará um relatório indicando o sucesso ou não do método e produto.

Estabelecidas os atributos e suas métricas dos interessados, pode-se iniciar o levantamento dos requisitos dos interessados e na sequência de sistema. Essa etapa irá

levar em consideração as características gerais dos satélites e as particularidades (escala, que implicam em diversas questões) do subsistema em questão.

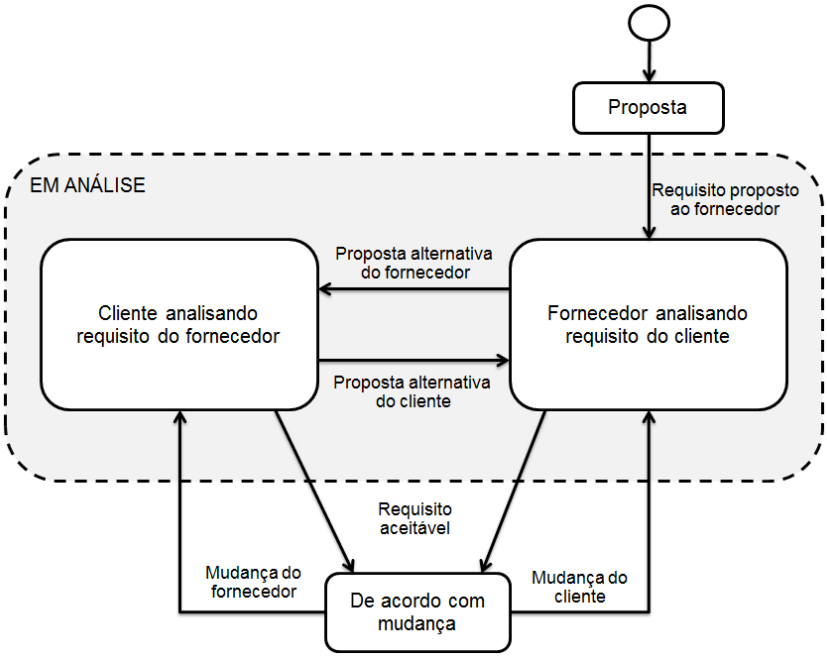
3.5. **Requisitos de Sistema**

O processo de levantamento de requisitos de um sistema parte dos requisitos dos interessados no sistema. Na prática, trata-se de um processo iterativo entre um **cliente** (aquele que demanda os requisitos para começar o desenvolvimento do sistema) e um **fornecedor** (aquele que irá elaborar verbalmente os requisitos, em linguagem clara e sem ambiguidades). Este é um ERS, enquanto aquele é um proponente.

Aquele que propõe o requisito em forma bruta é, geralmente, um interessado envolvido memo sem os conhecimentos específicos de elaboração dos requisitos. Mas ele possui mais autoridade, uma vez que formula e fornece o requisito em seu aspecto mais geral – numa linguagem não-técnica. Um diagrama descritivo pode ser visto na Figura 3.14.

O objetivo do ERS é capturar os **requisitos dos interessados**, organizá-los e deixá-los expressos da forma mais clara e completa possível. A seguir, esse documento será entregue a cada interessado e ao grupo, que deverão dar sugestões ou indicar equívocos de comunicação. Esse processo social iterativo perdura até o ponto em que os interessados e o ERS chegam a um meio-termo, indicando que os documentos estão suficientemente detalhados, podendo ser transformados em **requisitos de sistema**.

Figura 3.14 – Diagrama que estabelece o processo de elaboração de requisitos entre cliente e fornecedor.



Fonte: Adaptado de Hull et al. (2005).

É importante lembrar que o ERS não é dono dos requisitos, mas apenas um receptor de necessidades de sujeitos externos à sua organização. Sua missão é captar essas necessidades e trabalha-las até se chegar a requisitos de sistema, passando de uma linguagem informal e subjetiva para outra formal e objetiva, lançando as bases para o EP e o EMS desenvolverem seus subsistemas específicos.

Existem diversas formas de elicitar os requisitos dos interessados, que são necessidades humanas, expressas em linguagem não técnica. As fontes para a elaboração dos requisitos podem ser diversas, dentre as quais, de acordo com ALEXANDER & STEVENS (2002) podem ser elencadas em:

a) Entrevistas;
b) Observação de usuários no trabalho;
c) Relatórios de problemas;
d) Melhorias feitas pelos usuários;
e) Usos de um produto de modo diverso do esperado;
f) Projetos e especificações preexistentes;

g) Sugestões e reclamações de usuários.

Conforme será visto no Capítulo 4, o SCA de satélites é mais vulnerável às incertezas. De modo geral, podemos elencar as caraterísticas gerais (todos os satélites) e particulares (satélites médios) que influenciam em larga medida o projeto do SCA.

Para esse estudo será considerado um satélite a operar em órbita baixa (do Inglês *Low Earth Orbit*-LEO, entre 350 e 1400 km). Isso aumenta o efeito dos torques externos (aerodinâmicos) e de campo, (de gravidade e magnético), que foram incorporados ao modelo ambiental.

A Tabela 3.3 ajuda a delinear quais são as prioridades na elaboração dos requisitos do SCA em questão. Percebe-se que os requisitos de sistema devem se concentrar em aspectos geométricos, que levam à redução de massa e consequente aumento da influênciada variação do momento de inércia causada pelo consumo de propelente. Além disso, existem os torques de campo, a variação do momento de inércia e requisitos de estabilização particulares (quadrados em rosa). Isso não significa que os fatores também presentes (custos, geometria, carga útil e torques de perturbação) não devam ser levados em consideração. Trata-se apenas do que cada classe deve considerar como importante de ser incorporado ao seu respectivo modelo (marcado com ‘x’).

O fato de reduzir dimensões e massa do satélite gera particularidades a serem consideradas nos modelos – para o projeto adequado do veículo e de seus subsistemas.

A seguir, foram esboçados os requisitos referentes aos interessados: governo, AEB e instituto desenvolvedor (pesquisadores, engenheiros e técnicos).

Tabela 3.3 – Fatores de influência diferenciados (x) para diferentes classes de satélites.

FATORES DE INFLUÊNCIA (CONTROLE) PARA DIFERENTES CLASSES DE SATÉLITES				Classe do satélite		
				Convencional [>500 kg]	Minisat [500-100 kg]	Microsat [100-10 kg]
FATORES DE INFLUÊNCIA: SCAO	Missão	Órbita	Torques de distúrbio	x	x	x
			Campo gravitacional			x
			Campo magnético			x
			Carga útil	x	x	x
			Variação de momento de inércia			x
			Geometria	x	x	
			Custos	x	x	x
			Requisitos de estabilização		x	x

Fonte: Produção do autor.

Requisitos dos Interessados – Geral / Estrutura

a) A maior dimensão do satélite não pode exceder a menor dimensão da coifa do veículo lançador;
b) A massa do satélite deve ser pequena, já incluso todos seus subsistemas, propelente e módulos;
c) O volume do satélite deve ser inferior a 64 litros.

Requisitos dos Interessados – SCA

a) O subsistema deve ser robusto e estabilizar o veículo dentro de uma faixa de tempo próxima àquela dos satélites convencionais.

Requisitos dos Interessados – Micropropulsão

a) O subsistema deve fornecer uma ampla faixa de empuxo, fixa, ao longo de toda vida útil;

b) A eficiência do propulsor deve ser a maior possível para as dimensões consideradas;

c) O consumo energético do subsistema deve ser mínimo;

d) O subsistema deve ter fabricação relativamente simples quando comparado a outros;

e) A massa de propelente deve ser a menor possível considerando a vida útil, eficiência e empuxo necessários.

A partir desses requisitos foram elaborados requisitos de sistema. Estes devem ser estruturados em linguagem formal, especificando quantidades mensuráveis e não possuir ambiguidades de nenhuma natureza. A Prática Reomendada Aeroespacial do Inglês *Aerospace Recommended Practice*) SAE-ARP 4754 lista as classes de requisitos mais importantes para um sistema. Os requisitos devem ser considerados nas várias fases de desenvolvimento de um sistema (ex: nível funcional, de sistemas, subsistemas, componentes, software, entre outros). As classes de requisitos foram extraídas de HULL (2005). São elas: (a) Requisitos de Segurança; (b) Requisitos Funcionais; (c) Requisitos do Cliente; (d) Requisitos Operacionais; (e) Requisitos de Desempenho; (f) Requisitos de Instalação e Físicos; (g) Requisitos de Manutenabilidade; (h) Requisitos de Interface e; (i) Requisitos de Certificação.

Nesse trabalho foram considerados apenas os requisitos de desempenho – que incluem especificidades das funções tais como: acurácia, fidelidade, espectro, resolução, velocidade e respostas no tempo – e requisitos de instalação e físicos – que incluem tamanho, provisão para montagem, potência consumida, calor dissipado, massa, ventilação, restrições ambientais, acesso, manipulação e armazenamento. Estes foram considerados a título de delinear o desenvolvimento de um satélite do ponto de vista de sua estrutura e controle.

<u>Requisitos de Instalação e Físicos (Estrutura e Propulsão)</u>

a) A maior dimensão do satélite em configuração de lançamento não pode exceder 60 centímetros;

b) Os equipamentos / componentes externos à estrutura principal devem ter dimensão máxima inferior a 20 centímetros;
c) O satélite como um todo (módulo de serviço e de operação) deve ter massa inferior a 100 kg;
d) O volume do satélite deve ser inferior a 0,1$\boldsymbol{m^3}$ (100 L);
e) O subsistema deve fornecer um empuxo na faixa de 100 até 1000 μN durante sua fase de operação;
f) O impulso específico deve ser próximo a 140 s, com faixa de tolerância de 10%;
g) O subsistema deve ter um consumo energético inferior a 2 W;
h) O subsistema deve ser fabricado usando tecnologias já existentes;
i) A massa de propelente do subsistema não deve exceder 7% da massa do satélite e do propelente somadas.

Requisitos de Desempenho (Lei de Controle)

Em modo nominal, a atitude do satélite deve ser controlada nos três eixos para cumprir com os seguintes requisitos:

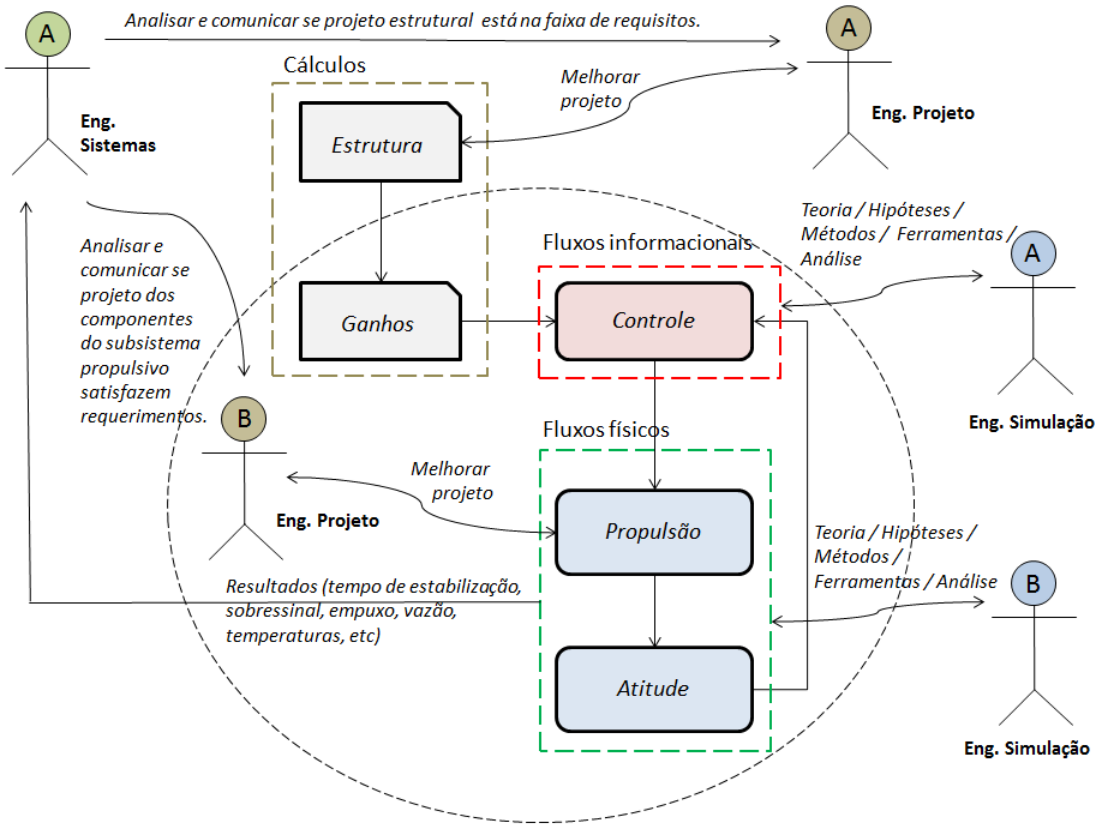
a) Erro de determinação de atitude menor que 0,005°;
b) Taxa de amortecimento $\boldsymbol{0,3 < \xi < 0,8}$;
c) $\boldsymbol{t_s = 100\ s}$ [$\boldsymbol{5\%\ do\ valor\ final}$];
d) Deriva menor do que $\boldsymbol{0,001^\circ\ /s}$.

3.6. **Conexão entre Ferramentas e Profissionais**

A cadeia de desenvolvimento é um processo que deve estabelecer uma conexão entre modelos, especialistas e métodos. A Figura 3.15 sintetiza o processo colaborativo elaborado no presente estudo, em linhas gerais. Cada ferramenta deve ser melhorada ao longo do tempo, assim como as interconexões da rede podem ser alteradas – o que dependerá da aprovação de usuários relacionados diretamente com a ferramenta em questão. Isso dá um dinamismo à rede que permite não apenas resultados de simulação atualizada e mais fiel, assim como um projeto mais orientado. O escopo da rede é coordenar o desenvolvimento do sistema de tal forma que cada profissional seja capaz

de trabalhar com dados atualizados, mapear submodelos e fazer uso de ferramentas mais adequadas a seu subsistema e componentes.

Figura 3.15 – Síntese de relações entre abordagens, especialistas e modelos no desenvolvimento colaborativo em questão.



Fonte: Produção do autor.

Percebe-se que existe uma comunicação tanto entre desenvolvedores e ferramentas, apenas entre desenvolvedores – e apenas entre ferramentas. Ou, em outras palavras, relações humano-máquina (H-M), humano-humano (H-H) e máquina-máquina (M-M). O ambiente colaborativo visa, através da integração M-M, viabilizar a comunicação eficiente, que poderá ser feita de várias formas. Para isso, faz-se necessário ter uma metodologia de processo, cujo centro de gravidade reside nas possibilidades de interconexão M-M.

Observando as interações H-H e H-M na Figura 3.15, é possível constatar que o Eng. de Modelagem e Simulação participa no processo através da seleção de ferramentas, na adoção de teorias e hipóteses adequadas, nos parâmetros de simulação e nas análises. Já o Eng. de Projeto se preocupa com o estabelecimento de parâmetros de projeto – sendo portanto muito interessado nas análises – e também nos resultados de simulação, que podem servir para validar os modelos após a construção de partes do subsistema. O Eng. de Sistemas se comunica diretamente com o EP. A diferença é que este inicia o detalhamento dos subsistemas e partes, que darão tom às etapas de desenvolvimento posteriores, que detalham até ao nível de componentes e fenômenos correlatos; e aquele faz uma mediação entre os requisitos de sistema e as diretrizes gerais do projeto, se preocupando não apenas em atender as demandas dos interessados, mas fazê-lo da melhor forma possível.

3.7. **Requisitos de Processo Colaborativo**

O processo colaborativo está inserido num determinado ambiente – neste caso o RCE. Viu-se que ele deve conjugar várias ferramentas de natureza distinta, e que há diversas formas de interligá-las. Além disso, o tipo de ligação pode ser de três tipos: (a) Requisitado (*Required* - R); (b) Requisitado se Conectado (*Required if Connected* - RIC) e; (c) Não Requisitado (*Not Required* - NR).

A ligação do tipo (i) significa que a conexão entre uma ou mais saídas da ferramenta A e uma ou mais entradas da ferramenta B (ou vice-versa) é indispensável, sem a qual o processo não pode ser executado. A ligação do tipo (ii) permite que saídas/entradas de uma ferramenta fiquem livres (sem conexão). Isso é interessante se uma ferramenta fornece várias saídas diferentes que ora podem ser necessárias ou não. Quando se tem o caso (iii) não é necessária a conexão, em nenhum caso. A diferença entre essa e o caso (ii) é que neste, se é feita a conexão entre a saída n de A com a entrada m de B, A deve fornecer a saída específica para B; ao passo que se a conexão é (iii), a ligação não influenciará na execução.

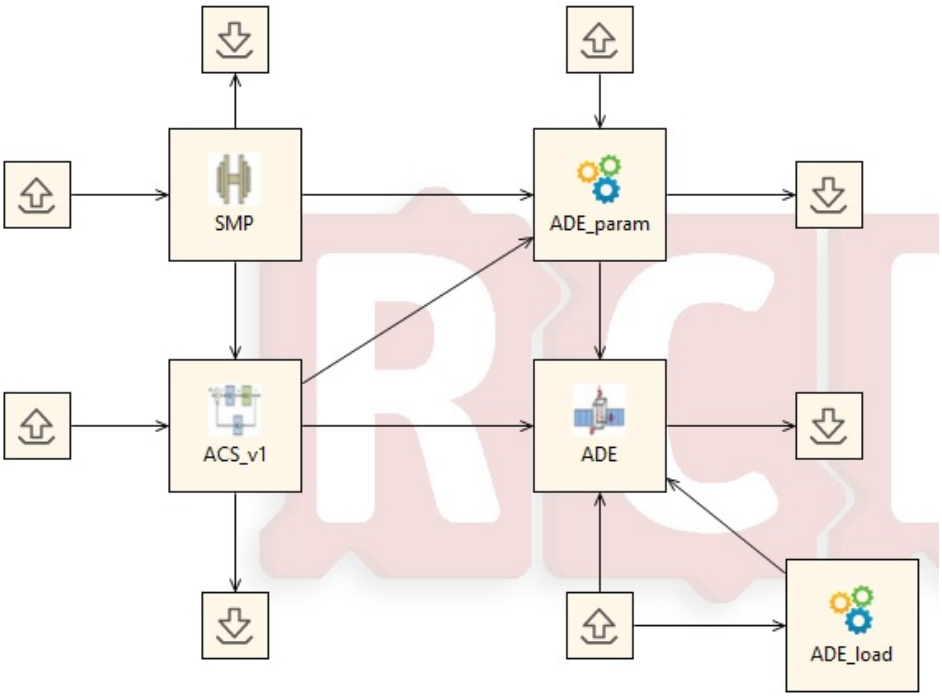
REQUISITOS DE CONEXÃO

Para o caso de estudo, foram estabelecidos requisitos de conexão conforme a expansibilidade do processo colaborativo. Desse modo, pode-se ter todas as ferramentas inseridas sem necessariamente estarem no ciclo. (1) *O primeiro requisito diz respeito aos tipos de ligação (NR, R, RIC).*

Para as conexões entre as ferramentas **Missão_e_Ambiente** e **SMP**, **ADE_param** e **ACS_v1**, exibidas na Figura 3.16, existe obrigatoriedade, uma vez que todas as ferramentas de sistema (SMP e ACS_v1) e de dinâmica de atitude (ADE_param) dependem das informações fundamentais. Logo, trata-se de uma ligação do tipo R ("*Required*"). Neste caso, a própria ausência de conexão ao se construir a cadeia de processo irá revelar que pode estar faltando uma informação

Em termos de variáveis e parâmetros que circulam na cadeia de processo, deve-se definir – para cada ligação – qual é a grandeza fluindo e qual a natureza do fluxo: se ele é um número real, inteiro, uma tabela, um vetor, uma matriz ou um arquivo de dados. Geralmente os números reais e inteiros que circulam pela cadeia são usados em execuções com retroalimentações, em que uma ferramenta recebe e fornece diversos resultados – nesse caso, os números seriam contadores do processo. Os fluxos de informações entre ferramentas se dão na forma de arquivos, cujos formatos podem ser dos mais variados, sendo de grande serventia para qualquer processo.

Figura 3.16 – Cadeia de processo 1 simplificada.



Fonte: Produção do autor.

As conexões restantes devem ser classificadas como RIC.

As saídas (*outputs*) capturam arquivos ou valores dados pela ferramenta em sua pós-execução. Toda informação que se deseje compartilhar na cadeia deve garantir que a ferramenta fornecedora dessa informação esteja conectada a uma saída – que irá armazená-la num diretório específico (definido no roteiro de pós-execução da ferramenta em questão).

REQUISITOS DE SEQUENCIAMENTO

Para iniciar o processo deve-se ter uma ferramenta que dê início à execução – que é justamente a **Missão e Ambiente**. Neste caso, o próprio ato de colocar a entrada das outras ferramentas como ligação tipo R já estabelece que, sem as informações de missão e ambiente, com suas constantes universais, órbita, unidades, dimensões e materiais, as outras ferramentas não iniciam sua execução.

Como se pode observar na Figura 3.16, todas as ferramentas estão interconectadas. Dessa forma não é necessária a imposição da ligação tipo R para todas as ferramentas,

mas apenas àquelas que estejam na primeira camada – ou seja, na segunda sequência de execução. Os requisitos esboçados pelo autor foram dois:

(**2**)*"Todo processo deve possuir uma ferramenta de inicialização que irá fornecer dados iniciais (arquivos, números, tabelas) a um ou mais elementos da cadeia."*;

(**3**) *"Todas as ferramentas conectadas diretamente à ferramenta de inicialização devem possuir ao menos uma grandeza (das diversas possíveis) que seja do tipo R (Requisitada), de tal forma que sem o fluxo dessa informação o processo não ocorra."*

REQUISITOS DE TEMPORIZAÇÃO

Para alinhar os parâmetros de simulação entre ferramentas que executem atividades com subsistemas relacionados na operação, foi necessário imbricar as principais informações referentes à simulação. Isso foi feito a partir de uma ferramenta-base (a 1ª a executar uma simulação) que, por sua vez, deve gerar um arquivo com dados de simulação (tempo inicial, tempo final, método, passo, intervalo de exibição de dados, etc.) em formato já adaptado às ferramentas que sejam executadas. Para isso é necessário conhecer previamente o tipo das outras ferramentas, especificamente a plataforma de simulação.

No caso do presente estudo, foi necessário que as ferramentas **ACS_v1** e **ACS_v2** gerassem arquivos com parâmetros de simulação (_.sim) para os modelos em *AMESim* contidos nas ferramentas **ADE**, **PROP_1** e **PROP_2**.

De modo geral, podemos destacar três requisitos importantes, esboçados pelo autor, relativos à temporização:

(**4**) *"É necessário que as informações relativas à simulação tenham uma ferramenta-base, na qual os parâmetros referentes a ela sejam repassados de forma a alinhar a atividade sendo feita."*

(**5**) *"A ferramenta-base pode passar quantidade menor, igual ou maior de informações para as outras ferramentas, a depender da natureza das receptoras (plataforma de simulação), a ser previamente estabelecida."*

(**6**) *"Ferramentas cuja execução não esteja relacionada a simulações (progressão no tempo) não devem receber informações de simulação."*

REQUISITOS DE AUTOMAÇÃO

A automação do processo está relacionada a um ciclo que se repete ("*loops"*), sendo sua interrupção (geralmente) condicionada a alguma condição pré-determinada.

O número de iterações irá depender da finalidade da cadeia de processo. Numa análise paramétrica, por exemplo, ter-se-á uma condição na qual o número de repetições está diretamente relacionado ao número de combinações de casos de variações de parâmetros. O requisito esboçado pelo autor foi:

(**7**) *"Para cada cadeia de processo deve-se definir se haverá alguma avaliação específica. Se sim, deve-se especificar qual (ex: otimização, análise paramétrica, estudo de convergência, etc)."*

É necessário definir quais serão as comparações feitas. Isso pode ser feito através de um código, cuja ferramenta já é disponibilizada pelo próprio ambiente (*script*).

3.8. **Desenvolvimento Colaborativo**

Os benefícios de um projeto de desenvolvimento colaborativo face aos métodos tradicionais serão tanto maiores quanto mais complexo (alto custo) for o sistema a se desenvolver. A Tabela 3.4 elenca as principais medidas de efetividade levantadas pelo autor, e as vantagens/desvantagens de cada tipo de desenvolvimento sob esse prisma.

Em termos de **processo**, o método tradicional (Td) se apoia em baixa intensidade de interação, o que leva a grande quantidade de iterações. Na colaboração (Co), como existe uma rede associada de forma a obter o melhor desempenho – usando a nova metodologia proposta – a interação é mais numerosa e significativa, tendo como consequência a menor necessidade de iterações individuais ou entre grupos.

Tabela 3.4 – Comparação entre desenvolvimentos (tradicional e colaborativo).

Desenvolvimento: Tradicional (Td) *versus* Colaborativo (Co)		
Medidas	**Tipos**	**Características relevantes**
Processo	Pouco interativo (Td)	Pouco intercâmbio de informação gerando iterações numerosas.
	Muito interativo (Co)	Muito intercâmbio de informação minimizando iterações.
Informação	Isolado (Td)	Acesso difícil a relatórios e informação. Pouco intercâmbio de conhecimento. Baixa taxa de aprendizado em equipe.
	Compartilhado (Co)	Agilidade no desenvolvimento. Comunicação fluída. Alinhamento de objetivos. Alta taxa de aprendizado geral (equipe).
Ferramentas	Concentrado (Td)	Mais máquinas e demanda de memória (RAM / HD). Muitas licenças de software (alto custo). Grande dependência.
	Distribuído (Co)	Menos máquinas e demanda de memória. Poucas licenças de software (menor custo). Independência relativa.

Fonte: Produção do autor.

Em termos de **informação** e **ferramentas**, o Co é muito mais eficaz, pois a interconexão entre as ferramentas possibilita a livre-circulação de informação, o que leva a uma maior eficiência no processo. Por ser distribuído, o processo requer menos máquinas instaladas em cada nó (menos licenças), aumentando a independência de software pagos – diminuindo custos. Além disso, a execução fica mais rápida, uma vez que cada nó só precisa executar sua ferramenta.

3.8.1. **Natureza e Classificação das Ferramentas**

As particularidades e diferenciais do desenvolvimento colaborativo incluem: (a) integração das ferramentas; (b) interconexão das ferramentas; (c) execução distribuída; (d) mapeamento de ferramentas; (e) automação de processo via estudos específicos envolvendo a cadeia.

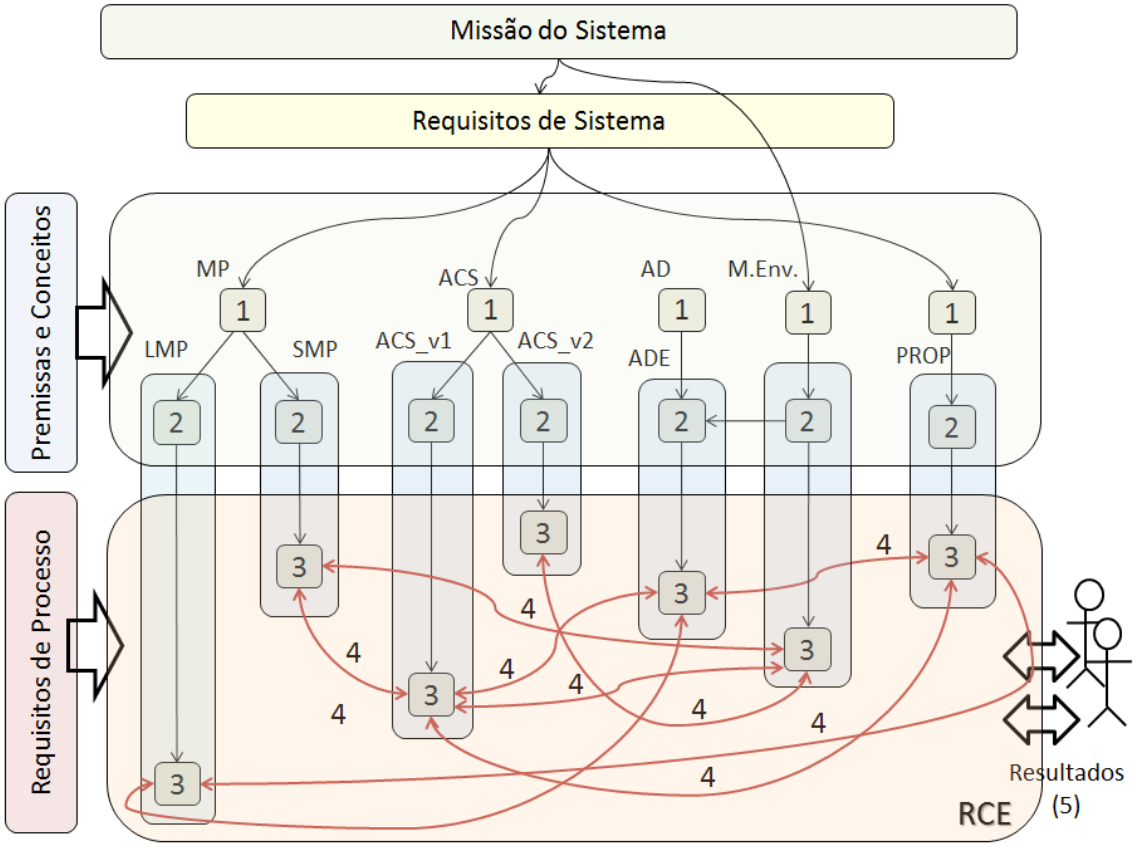
A ideia geral do estudo foi desenvolver um processo que parte de múltiplos desenvolvimentos individuais – a serem realizados pelas equipes; – que se materializam em ferramentas – de cálculo, de simulação, documentos de especificações; – e que se integram e interconectam para formar uma determinada cadeia de processo. No caso deste trabalho, foram construídas cinco ferramentas principais, que representam propriedades de massa, dinâmica rotacional e seus distúrbios, lógica de controle e propulsão; além de quatro auxiliares, referentes a manuseio, geração e carregamento de arquivos.Esse desenvolvimento possui diversos elementos que seguem uma trajetória individual (partes 1 e 2) a princípio, se integrando e interconectando (partes 3 e 4), formando um todo unitário (processo).

Na Figura 3.17, a missão do sistema viabiliza o delineamento dos requisitos do mesmo. Estas duas diretrizes (missão e requisitos) irão delinear que arcabouço de premissas e conceitos serão levantados para a elaboração dos $\boldsymbol{n}$ modelos simbólicos (**parte 1**), organizados numa mesma linha e $\boldsymbol{n}$ colunas. Esses modelos simbólicos, aliados a outras ferramentas – que podem não ser modelos, mas apenas códigos de cálculo ou conversão de dados, entre outros – dão gênese aos modelos e ferramentas computacionais. Inicia-se aí a elaboração individual das mesmas, com suas particularidades como abordagem, plataforma computacional e domínios (**parte 2**). Até esse ponto todo o processo é feito de modo pouco coordenado, podendo ser à distância. Essa fase é destacada com um quadrado transparente cinza.

Ao se estabelecer um ambiente colaborativo (quadrado transparente rosa), é possível, a partir de um roteiro, incorporar as diversas ferramentas num “local virtual” comum. Essas ferramentas devem cumprir certos requisitos para que possam ser executadas (requisitos de ambiente). Essa é a **parte 3** do desenvolvimento geral.

Ao estarem disponibilizadas no *menu* de opções do ambiente, essas ferramentas podem ser combinadas das formas mais diversas (**parte 4**). Essas interconexões podem ser múltiplas, sendo necessário estabelecer uma metodologia e diagramação para o projeto de sistemas complexos.

Figura 3.17 – Sistematização do desenvolvimento colaborativo com as interconexões propostas.



Fonte: Produção do autor.

Quanto maior for a complexidade, mais necessário será o uso de métodos eficazes para conduzir as equipes e o amadurecimento do sistema proposto. Os *requisitos de processo* vêm dar um suporte valioso, estabelecendo fronteiras, necessidades e recomendações aos usuários. O *diagrama de relações inter-ferramentas* tem por foco estabelecer um mapa que pode levar a uma *política de reuso* eficaz. A dinamicidade do diagrama implica a característica fluída das interconexões. Trata-se de *eficiência a nível sistêmico* – ou seja, de desenvolvimento de sistemas complexos.

Para ser executada, cada ferramenta necessita de um roteiro de pré e pós-execução, além de um comando de execução – para acionar a ferramenta. Existe uma diversidade de componentes disponíveis na cadeia de processo como, por exemplo, o componente de base de dados (*database*), que permite o acesso de qualquer usuário à base de dados do projeto; o provedor de entradas (*input provider*), que envia arquivos, números, vetores, matrizes, diretórios de entrada aos componentes; o componente de saídas (*output*

writer), que armazena saídas de outros componentes; o componente de convergência (*converger*), responsável por comparar valores inteiros e reais de execuções passadas e compará-los; o otimizador (*optimizer*), que otimiza variáveis de projeto; entre outros.

Um acoplamento entre componentes é necessário para a execução do projeto. Essa conexão é um canal de dados ligando componentes. Os tipos de dados admissíveis são: textos pequenos, números (inteiros ou reais), booleano (verdadeiro/falso), arquivos, diretórios, tabelas pequenas, vetores e matrizes. Nem todos os componentes admitem os formatos de dados listados. Portanto, há condições para criar um canal de comunicação entre entrada e saída se:

a) O tipo de dado de saída é igual ao de entrada – ou conversível a ele;
b) Uma entrada estiver conectada a uma única saída.

Há também um esquema de execução que deve ser especificado a cada rodada no projeto. Esse processo envolve duas definições: *manuseio de dados* e *restrições de execução*.

O **manuseio de dados** pode se dar de três modos: constante (*constant*), único (*single*) e fila (*queue*, *consumed*). No primeiro, o valor não será consumido durante a execução e será reutilizado na próxima iteração. No modo único, o valor de entrada será consumido durante a execução e não será reutilizado na próxima iteração. No modo fila, o valor de entrada será consumido e não será reutilizado na próxima execução se houver um *loop* na cadeia.

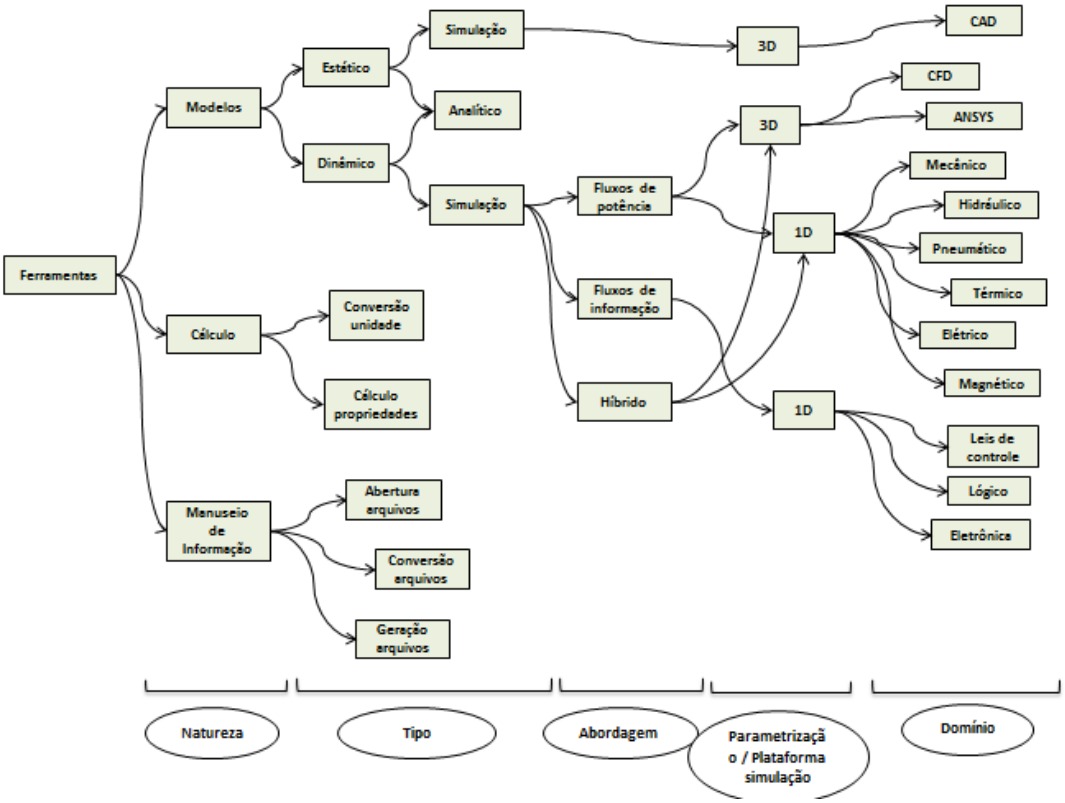
Na execução, há três tipos de restrições: requisitado, requisitado se conectado, e não requisitado. O primeiro indica que a entrada é necessária para a execução do componente; o segundo indica que não é requisitado, mas se uma entrada for conectada ela será tratada como no primeiro caso (essa situação é de indiferença); no terceiro a entrada não é requisitada, e se esta for passada, o valor será repassado ao componente se houver um valor disponível para execução no tempo (obs: valores desse tipo não podem acionar execução de componentes exceto se for a única entrada definida para um componente). No tópico 3.7 foi visto que esse aspecto é crucial para se determinar o sequenciamento do processo, sendo indispensável à elaboração de requisitos desse tipo.

FERRAMENTAS

Além dos modelos computacionais, existem outros tipos de ferramentas, que são aquelas de estudo – já contidas no próprio ambiente – e auxiliares que, no caso deste estudo, tem por escopo: (a) desdobrar arquivos; (b) alterar dados dos mesmos; e; (c) convertê-los.

Elaborou-se um diagrama que ilustra os tipos de ferramentas e as especificidades dos elementos contidos na mesma, conforme pode ser visto na Figura 3.18.

Figura 3.18 – Diagrama de classificação de ferramentas (em azul) e seus elementos.



Fonte: Produção do autor.

As ferramentas possuem uma natureza, que pode ser: modelos, de cálculo e manuseio de informações. Existem mais quatro atributos que podem ser usados para rotular elementos contidos nas ferramentas. Esses atributos estão explicados abaixo.

Natureza:

Especifica se se trata de uma ferramenta cujo enfoque seja a representação (modelamento) de um fenômeno/subsistema – geralmente dinâmico – ou calcula propriedades (escopo de projeto), converte unidades ou realiza manuseio de informações. Esse é o aspecto mais evidente a ser inserido numa busca orientada.

Tipo:

É uma particularidade específica da ferramenta. Todas possuem esse atributo. No caso de modelos deve-se definí-lo como estático ou dinâmico. Se de cálculo deve-se definir qual o escopo: determinar propriedades de um fluido; converter unidades; calcular momento de inércia, etc. Caso se trate de manuseio de informações, deve-se apontar qual o tipo de tarefa a ser executada: será feita uma simples abertura de um arquivo (ex: descompactação); será feita uma conversão (um formato para outro); será gerado um arquivo a partir de informações de outros, etc.

Abordagem:

Essa subclassificação é específica para ferramenta de natureza "modelos", tipo: "dinâmicos de simulação". Deve-se especificar qual será o paradigma a ser adotado para a construção dos modelos. Fluxos de informação (unidirecionais e adimensionais) ou fluxos físicos (bidirecionais com dimensões características). Pode ser que a ferramenta utilize modelos que façam uso das duas abordagens (híbridos).

Parametrização e Plataforma de Simulação

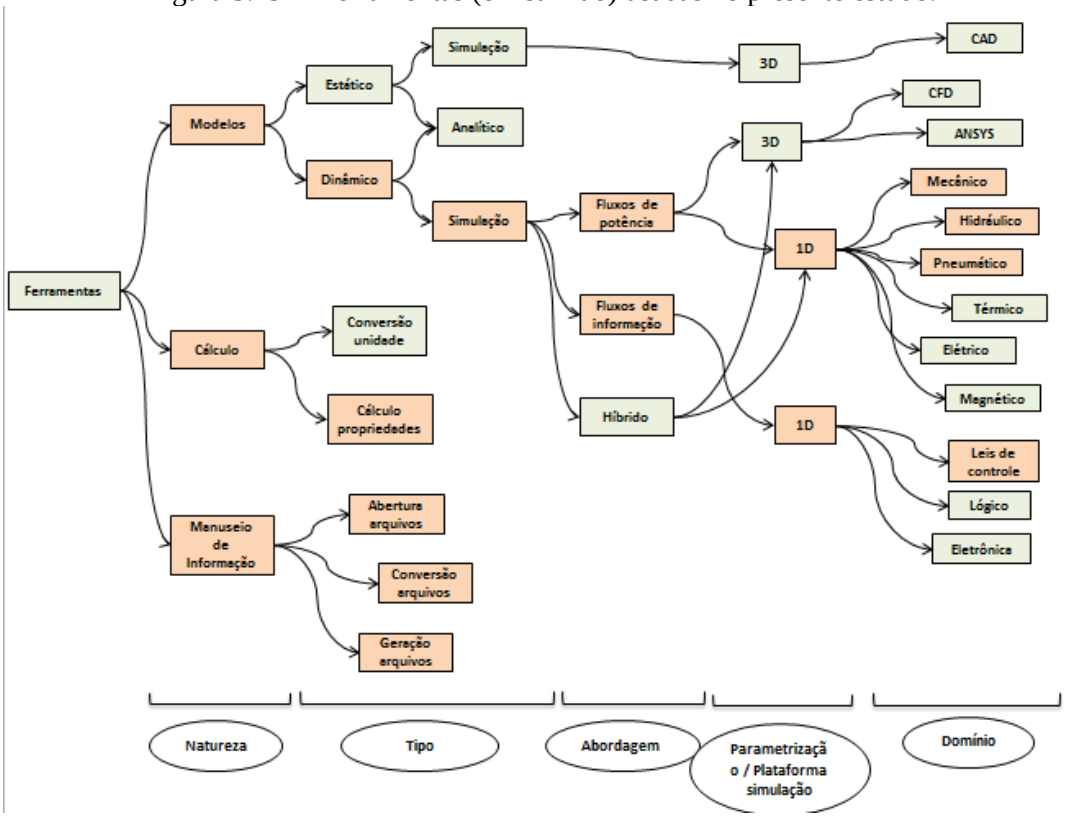
Esse critério diz se os modelos dinâmicos – aí incluídos os de tipo "estáticos de simulação" – estão fazendo um estudo capaz de analisar qualquer ponto no espaço, considerando a continuidade (tridimensionais); ou se modelam e capturam variáveis em pontos específicos de uma dada trajetória na qual circulam variáveis (unidimensionais ou parâmetros concentrados). Essa descrição está imbricada com a plataforma de simulação adotada: cada uma é construída para uma determinada aplicação. Pode haver mais de uma plataforma adequada para um enfoque (contínuo ou concentrado) – mas não é possível aplicar os dois enfoques para uma mesma plataforma. Trata-se de ferramentas muito específicas que possuem modos de operação muito particulares, e portanto distintos.

Domínio:

Esse último critério estabelece a disciplina abordada pelo elemento: térmico, mecânico, hidráulico, pneumático, elétrico, magnético ou uma combinação dessas (domínio físico); leis de controle, lógica, eletrônica (domínio da informação); ou projeto (ex: CAD), análise de escoamento (CFD), de tensões (ANSYS), etc.

Para as cadeias de processo desenvolvidas nessa Tese, foram utilizadas alguns elementos, destacados em cor salmão na Figura 3.19.

Figura 3.19 – Ferramentas (em salmão) usadas no presente estudo.



Fonte: Produção do autor.

De modo geral, podemos elencar duas classes de ferramentas: as principais e suas auxiliares. Estas podem ou não ser necessárias à execução das primeiras – a depender da natureza das mesmas e do escopo da cadeia. Cada uma delas será descrita em detalhes a seguir.

As **ferramentas principais** são aquelas diretamente envolvidas no processo colaborativo. Elas são: as de propriedades de massa (**LMP** e **SMP**); de ganhos e leis de controle de atitude, considerando apenas a planta (**ACS_v1**) quanto a planta e o atuador (**ACS_v2**); da dinâmica de atitude e dos distúrbios (**ADE**); e do atuador (**PROP_1** e **PROP_2**). Essas ferramentas conjugam um ou mais modelos, programas de cálculos ou de manuseio de informações.

As propriedades estruturais de massa, SMP, fixas na operação, variável no projeto, possui um arquivo ou dois: dois se não houver presença da ferramenta de missão e ambiente; um se houver. No primeiro caso, considera-se uma execução simples, na qual dados gerais do ambiente espacial e dimensões estejam atomizados. Essa é uma condição com menos interconexões e, portanto, constitui uma cadeia menos integrada. As propriedades de massa do propelente, LMP, variável na operação e projeto, combinam dois arquivos. A dinâmica veicular, ADE, é única e faz uso de um único arquivo. As leis de controle (ACS_v1 ou ACS_v2) possuem, a depender do caso, dois ou três arquivos cada, sendo um deles o modelo de simulação.

O atuador (subsistema propulsivo), PROP, faz uso de dois arquivos na forma de modelo de simulação: um para o tanque, tubulação e válvula de controle, *Propulsion_sys.ame* (PROP_1); e outro para o propulsor, *Thruster.ame* (PROP_2). Essa separação tem seu motivo: a menor quantidade de arquivos a serem desdobrados pelas ferramentas de carregamento (PROP_1_load e PROP_2_load). Assim, há menos probabilidade de ocorrer um erro na transcrição de arquivos. Soma-se a isso o fato de que a execução de modelos separadamente, em sequência, exige menos memória da máquina. E cada um (subsistema e propulsor) pode ter suas representações construídas em máquinas diferentes.

As **ferramentas auxiliares** foram desenvolvidas para permitir o concatenamento entre as ferramentas principais. Foi preciso reescrever os arquivos de *parâmetros globais* (**modelo_.gp**) e de *simulação* dos modelos (**modelo_.sim**) por fluxos físicos baseados nas informações calculadas / simuladas / impostas por outras ferramentas. Essa adaptação permite afirmar que as cadeias de processo sejam minimamente integradas, estabelecendo uma comunicação entre si.

O primeiro tipo de ferramenta auxiliar é a que desdobra os arquivos dos modelos AMESim, **Modelo_load**, no qual "Modelo" é o nome do modelo. Logo, tem-se **ADE_load** e **PROP_load**. Essas ferramentas são importantes para permitir o controle dos modelos *AMESim*, pois pode-se escolher quais arquivos carregar. No caso, como existem parâmetros de modelos e de simulação fluindo pela cadeia, os arquivos de parâmetros globais (**modelo_.gp**) e de simulação (**modelo_.sim**) devem ser gerados separadamente, tendo por base os resultados de outras ferramentas.

A ferramenta auxiliar que lê informações e gera os arquivos que serão adotados para a execução dos modelos físicos é denominada **modelo_param** e foi escrita em Matlab (.m). Da mesma forma, nesse estudo, existem duas: **ADE_param** e **PROP_param**. Elas fazem uso de um *script* que pode ser escrito em qualquer linguagem computacional (C++, *MatLab*, Scilab, Octave, etc.), e tem a função de adaptar as informações de uma ferramenta para outra.

As ferramentas auxiliares são adaptáveis e foram usadas no intuito de criar uma interconexão entre as ferramentas principais – elas servem de intermediárias no processo. Elas estabelecem relações entre ferramentas de cujo escopo seja a simulação de uma parte do sistema (operação). As do tipo "modelo_param" estabelecem uma ponte entre uma ferramenta operando por fluxos informacionais e outra por fluxos físicos. Elas devem gerar arquivos específicos para a execução dos modelos operando sob o paradigma físico, que irão ler suas informações (parâmetros globais) e, a partir desses, executarem uma corrida de simulação.

As ferramentas do tipo "modelo_load" se relacionam única e exclusivamente com a ferramenta principal por fluxos físicos, que desdobra os inúmeros arquivos do modelo em questão – exceto aqueles que serão alimentados por outros elementos da cadeia. Isso pode ser feito graças à funcionalidade dos roteiros de pré e pós-execução das ferramentas, cujos roteiros podem selecionar os arquivos desdobrados e repassar apenas aqueles de interesse. Manuseia-se assim de forma a aprofundar as interconexões.

Graças a essa funcionalidade foi possível não apenas fazer os parâmetros de projeto fluir por toda a cadeia, como estabelecer uma base comum de simulação (parâmetros de

simulação). Enquanto os parâmetros de primeiro tipo (projeto) tem um circuito mais amplo, o segundo fica restrito a um círculo menor (âmbito da simulação).

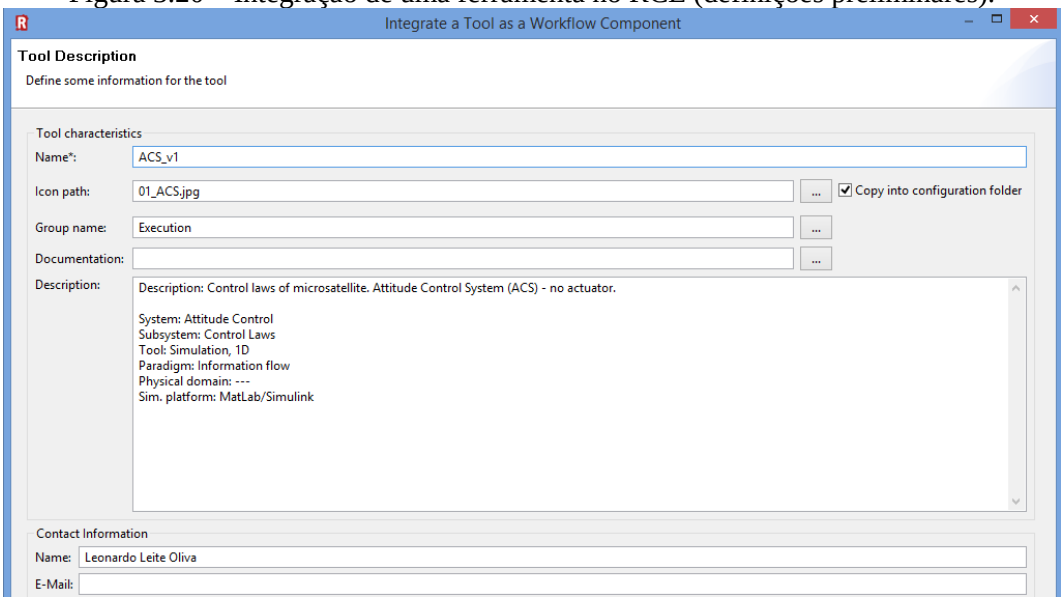
3.8.2. **Integração das Ferramentas**

O primeiro passo para a construção de uma cadeia de processo é integrar as ferramentas dos usuários. Os constituintes das ferramentas (modelos, *scripts*, tabelas) já devem existir (Capítulo 6). É necessário criar um novo projeto, no qual serão inseridos todos os arquivos relacionados a ele – visualizáveis por todos os nós da rede.

Para integrar uma ferramenta deve-se dar um nome a ela – com possibilidade de inserção de um ícone ilustrativo, como um arquivo .jpeg – e apontar a espécie a qual ela pertence (ex: *Data flow*, *Data*, *Evaluation*, *Execution*, *CPACS*,...). Pode-se associar uma documentação (pasta, arquivos) a ela, além de haver a opção de uma breve descrição, como sua função, limites, interfaces, plataforma de execução, entre outros, mostradas na Figura 3.20.

Na descrição, são inseridas as informações mais importantes da ferramenta que permitem mapeá-la no diagrama de ferramentas, já apresentado nas Figuras 3.8 e 3.9 para o caso de estudo a ser desenvolvido. Existe a possibilidade de anexar arquivos (.pdf ou .txt) a cada ferramenta. É recomendável que todas as ferramentas no ambiente colaborativo – e não apenas na cadeia – possuam um mesmo núcleo de arquivos anexos, cujo conteúdo deve ser o mapa de relações e suas tabelas correspondentes. Isso visa facilitar a busca e o reuso de modelos.

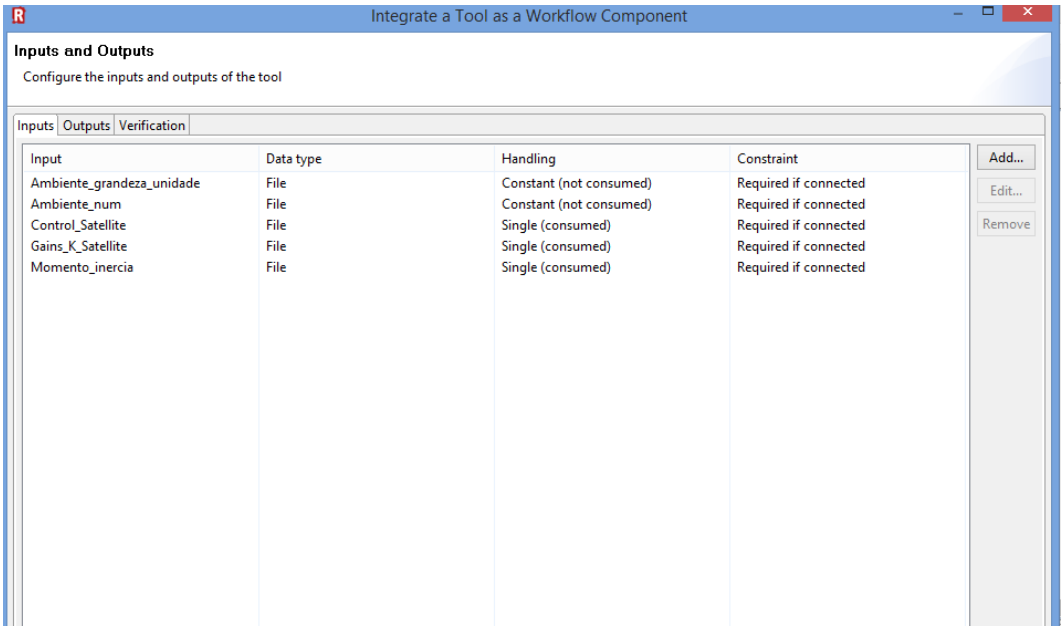
Figura 3.20 – Integração de uma ferramenta no RCE (definições preliminares).



Fonte: Produção do autor.

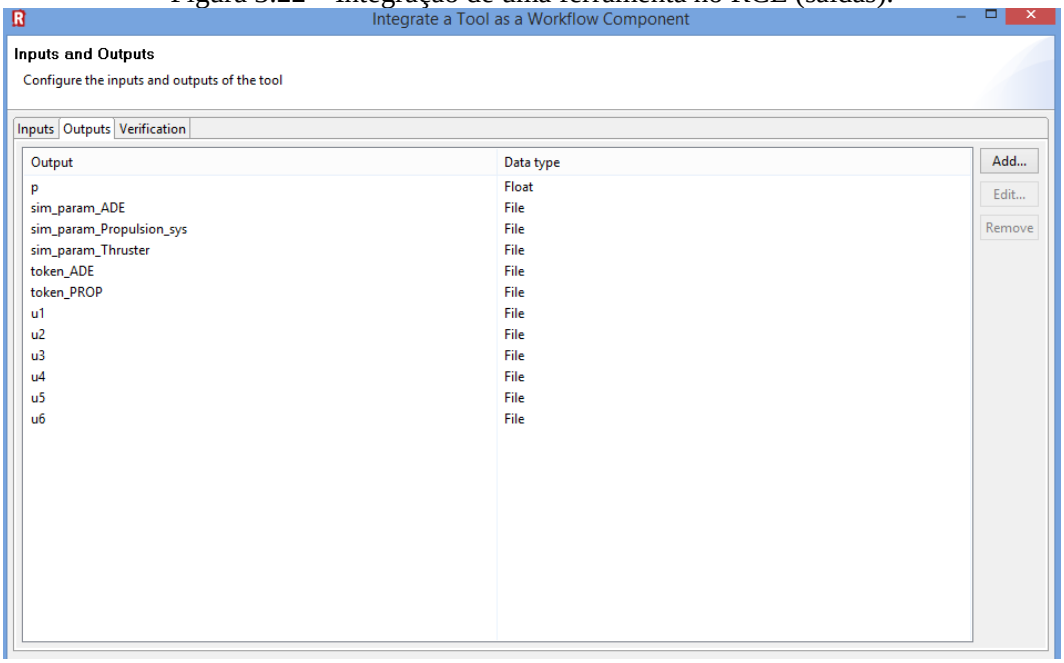
O próximo passo consiste em definir as entradas e saídas, como visto nas Figuras 3.21 e 3.22. Por exemplo, a ferramenta **ACS_v1** tem – dentre outras – uma entrada denominada *Ambiente_num.data* e fornece várias saídas denominadas *ui*.data (i = 1 a 6), que são arquivos que contém o vetor dos sinal.

Figura 3.21 – Integração de uma ferramenta no RCE (entradas).



Fonte: Produção do autor.

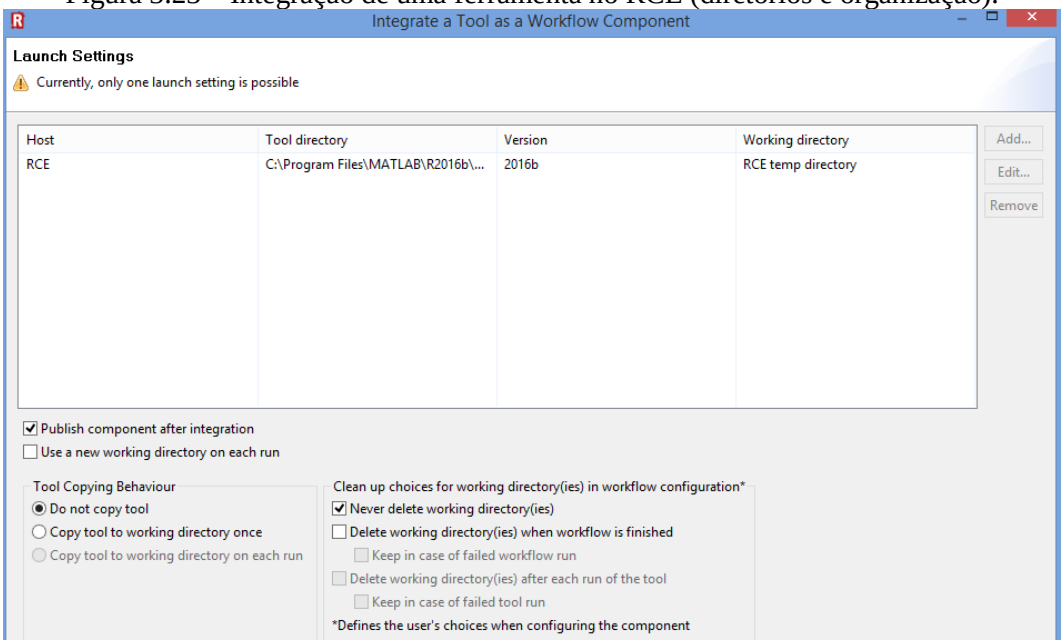
Figura 3.22 – Integração de uma ferramenta no RCE (saídas).



Fonte: Produção do autor.

Em seguida, os diretórios de execução (local do executável da ferramenta) e de trabalho (inserção de arquivos pertinentes de entrada e saída) devem ser inseridos nessa etapa, como ilustrado na Figura 3.23.

Figura 3.23 – Integração de uma ferramenta no RCE (diretórios e organização).



Fonte: Produção do autor.

A Figura 3.23 mostra a escolha do diretório da ferramenta. É nessa etapa que deve ser especificado o diretório em que a ferramenta está (ou será acionada). Ele determinará onde irá ser feita a execução – pelo *script* de execução. Além da definição do local da ferramenta e do local de trabalho, observa-se nas opções do menu inferior, que há quatro formas de organizar os diretórios de ferramentas e arquivos envolvidos (execução, entradas, saídas). Cada uma delas tem sua particularidade, conforme pode ser visualizada na Tabela 3.5.

Tabela 3.5 – Alternativas para as escolhas do diretório de trabalho e da política de reprodução de ferramentas.

		Escolhas			
		Usar novo diretório de trabalho a cada execução	Não copiar ferramenta	Copiar ferramenta no diretório de trabalho (uma vez)	Copiar ferramenta no diretório de trabalho (a cada execução)
Opções	#1		x		
	#2			x	
	#3	x		x	
	#4	x			x

Fonte: Adaptado de *RCE User Guide* (2018).

Cada opção traz uma particularidade:

a) Na opção #1, o diretório da ferramenta está apartado do de trabalho e não se usa uma nova pasta (diretório) a cada execução;

b) Na opção #2, a ferramenta é copiada ao diretório de trabalho uma única vez mas todas as execuções ficam na mesma pasta;

c) A opção #3, mantém a ferramenta no diretório de trabalho mas a cada execução criam-se novos diretórios de trabalho;

d) Na opção #4 tanto a ferramenta quanto o diretório de trabalho são copiados / criados (respectivamente) a cada execução.

Como se percebe, existe um aumento de repetitividade à medida que se passa da opção #1 a #4; de 1 para 2, funde-se a ferramenta com a entrada, saída e configuração; de 2 para 3, repetem-se os arquivos a cada execução; de 3 para 4, repetem-se não apenas os arquivos mas a própria ferramenta.

O passo final consiste em escrever os roteiros, cuja finalidade é:

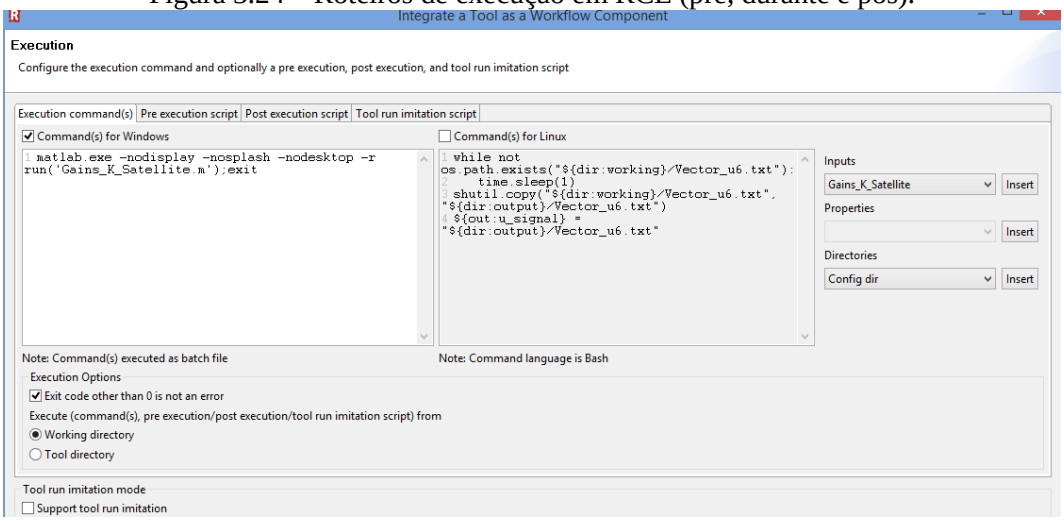
Pré-execução: copia todos os modelos, figuras, documentos e demais arquivos que serão (ou poderão ser) utilizados pela ferramenta no diretório de trabalho.

Execução: aciona cada modelo a ser usado pela ferramenta em questão.

Pós-execução: copia as saídas (desejadas) da ferramenta para um diretório específico, em formado predefinido.

No presente estudo, foram elaboradas várias ferramentas que intercambiam informações entre si. A Figura 3.24 mostra a janela para inserção dos códigos de pré-execução, execução e pós execução.

Figura 3.24 – Roteiros de execução em RCE (pré, durante e pós).



Fonte: Produção do autor.

A listagem das ferramentasdesenvolvidas com uma breve descrição se encontra a seguir:

a) ACS_v1: modelo de simulação do controle de atitude sem detalhes do atuador (controlador + planta);
b) ACS_v2: modelo de simulação do controle de atitude com atuador (controlador + +atuador + planta);
c) ADE: modelo da dinâmica de atitude do satélite por fluxos físicos;
d) ADE_load: ferramenta que abre os arquivos para edição/operação do modelo ADE;
e) ADE_param: ferramenta que converte informações de ferramentas em parâmetros e gera um arquivo correspondente a ser incorporado pelo modelo ADE durante a execução;
f) PROP_1: modelo da parte hidráulica do subsistema propulsivo (fluxos físicos);
g) PROP_2: modelo do propulsor (termodinâmica) (fluxos físicos);
h) PROP_param: ferramenta que converte informações de ferramentas em parâmetros e gera um arquivo correspondente para uso da ferramenta PROP_1;
i) PROP_load: ferramenta de abertura dos arquivos para edição/operação dos modelos PROP_1 e PROP_2;
j) LMP: ferramenta de cálculo das propriedades inerciais do fluído;
k) SMP: ferramenta de cálculo das propriedades inerciais do satélite;
l) Param_Analysis: ferramenta que administra análises paramétricas, definindo quais e de que forma certos parâmetros de entrada (projeto) variam ao longo de múltiplas execuções;
m) Mission_Env: ferramenta com informações fundamentais do projeto (parâmetros de entrada fixos).

As ferramentas, após desenvolvidas e integradas ao ambiente, ficam disponíveis no *menu* a todos usuários, que podem utilizá-las em seus processos. No entanto, a edição fica restrita ao nó que a integrou (o desenvolvedor).

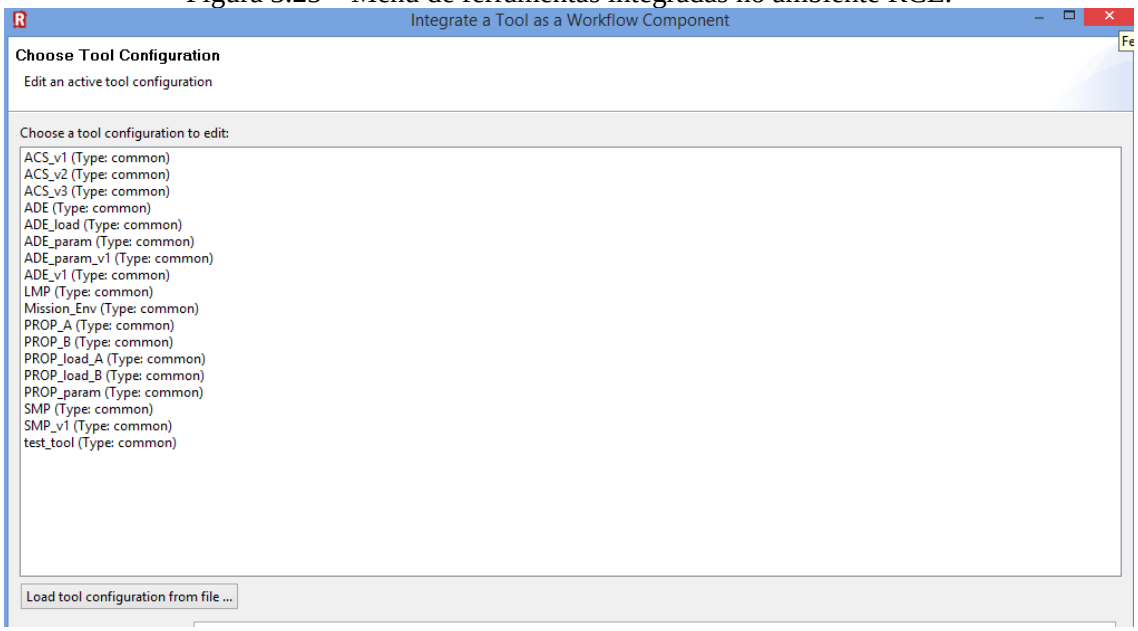
3.8.3. **Interconexão de Ferramentas**

Com todas as ferramentas inseridas no *menu* de execução do RCE, mostrado na Figura 3.25, pode-se construir uma ou mais cadeias de processo. Nesse trabalho, optou-se por experimentar duas configurações (processos) ligeiramente diferentes; e, a seguir, fazer uma análise paramétrica fazendo-se uso de, pelo menos, uma delas. Dessa forma, foram construídas três cadeias.

Podemos fazer uma analogia entre integração e interconexão de ferramentas com o mundo gastronômico: a integração seria equivalente à compra e inserção de ingredientes na dispensa. Quanto maior for a variedade de insumos, maior o repertório de receitas que podem ser feitas. As receitas diversas seriam as diferentes formas de combinar (interconectar) ingredientes (ferramentas). O que define o tipo de ligação é o estudo a ser realizado (objetivo). Assim percebe-se que a intersubjetividade – citada no início deste trabalho – está implícita no processo. Ela nada mais é do que a decisão do estudo preliminar a ser feito (Fases 0 e A) baseado em interações humanas (prioridades do ponto de vista técnico, de disponibilidade, de imposição financeira, etc).

É nessa etapa que as possibilidades são variadas. Trata-se do aspecto sistêmico da engenharia, no qual as interconexões são objeto de atenção maior do que as especificidades técnicas de cada elemento, conforme a Tabela 3.6. É nesse nível que ocorre o trabalho colaborativo – o desenvolvimento de cada ferramenta cabe mais às próprias equipes, que conhecem melhor do que as outras os fenômenos com os quais lidam, além de suas representações, métodos e plataformas de simulação.

Figura 3.25 – Menu de ferramentas integradas no ambiente RCE.



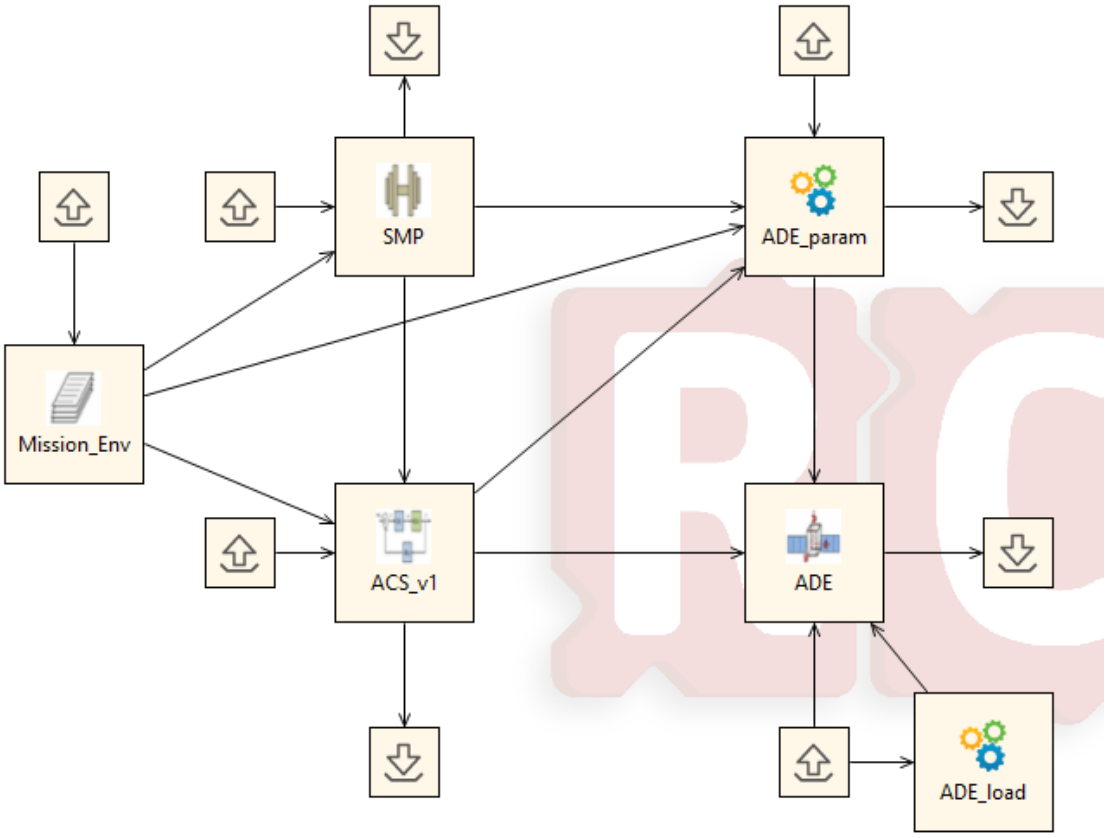
Fonte: Produção do autor.

A seguir, ver-se-ão as especificidades das configurações dos processos fundamentais (1 e 2), ressaltando qual a dinâmica está implícita em cada um.

PROCESSO 1

O processo 1, exibido na Figura 3.26, visa integrar três disciplinas: propriedades de massa/ inerciais; leis de controle e dinâmica de movimento (atitude) de um satélite artificial. O objetivo é fazer as ferramentas operarem em sintonia, fornecendo informações relevantes umas às outras. Nessa cadeia existem quatro ferramentas principais: uma fundamental (Mission_Env) e três representando as disciplinas (SMP, ACS_v1 e ADE) e duas auxiliares (ADE_param e ADE_load) que servem para carregar arquivos e gerar arquivos de parâmetros.

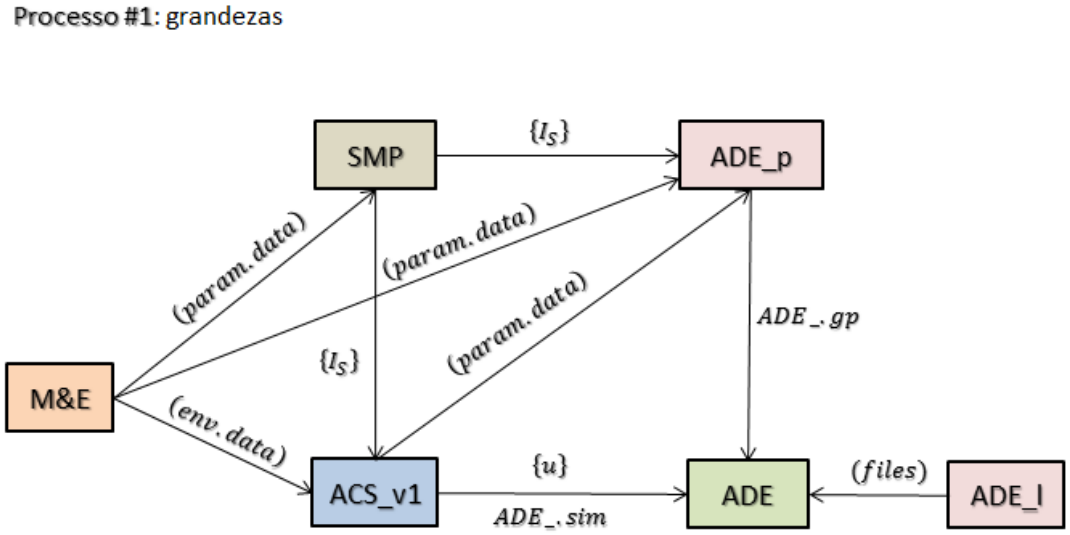
Figura 3.26 – Cadeia de processo 1.



Fonte: Produção do autor.

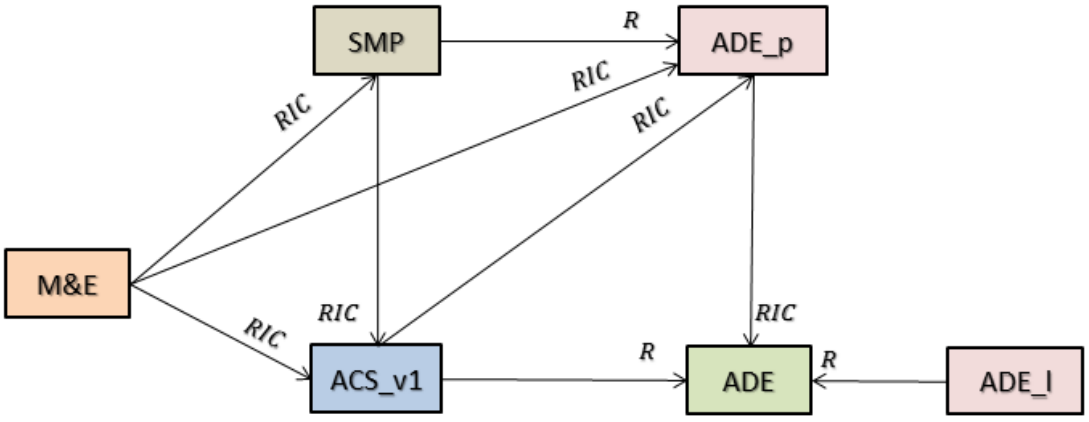
Foi construído um diagrama simplificado que explicita as grandezas e as conexões fluindo através da cadeia, mostradas nas Figuras 3.27 e 3.28.

Figura 3.27 – Diagrama simplificado do processo 1: grandezas.



Fonte: Produção do autor.

Figura 3.28 – Diagrama simplificado do processo 1: conexões.



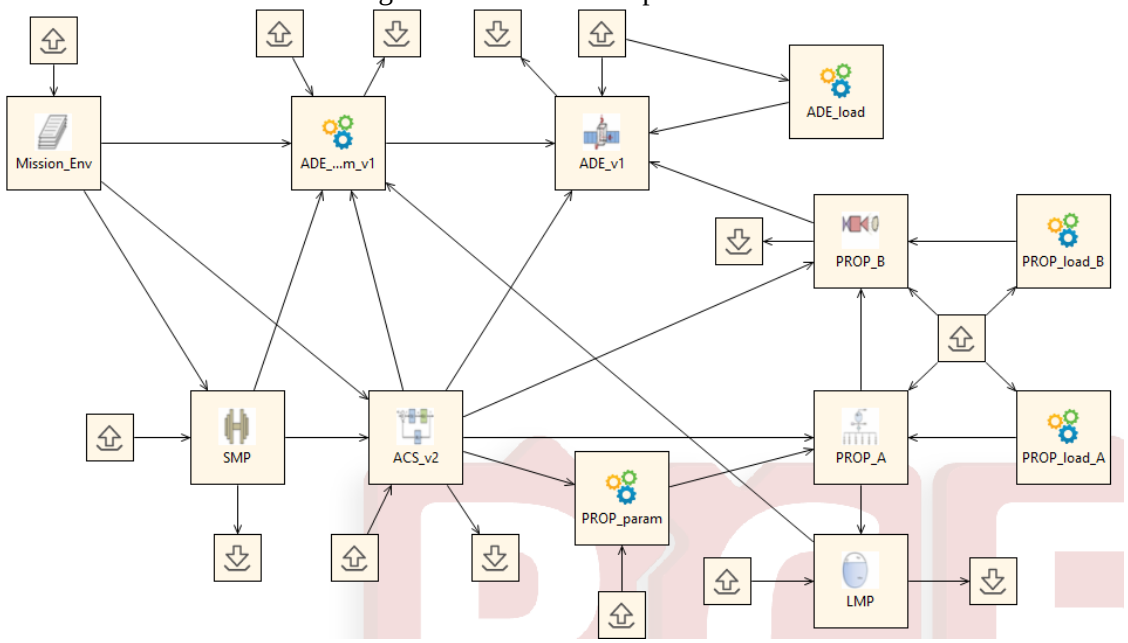
Fonte: Produção do autor.

PROCESSO 2

O processo 2, ilustrado na Figura 3.29, é um aprofundamento do primeiro. Ele leva em consideração o modelo específico do atuador e seu subsistema (propulsivo), além de considerar as propriedades mássicas do fluido. As ferramentas PROP_A, PROP_B, PROP_loadA, PROP_load_B e PROP_A_param, além de LMP, foram adicionadas ao processo, tornando-o mais complexo em termos de requisitos e execução.

Por envolver mais ferramentas e conexões, a probabilidade de haver falhas é maior – há mais arquivos envolvidos, mais conversões e o tempo de execução é maior. Para facilitar o desenvolvimento, optou-se pela criação de cadeias de processo auxiliares para testar um subconjunto de ferramentas – assim foi possível identificar erros em tempo menor.

Figura 3.29 – Cadeia de processo 2.

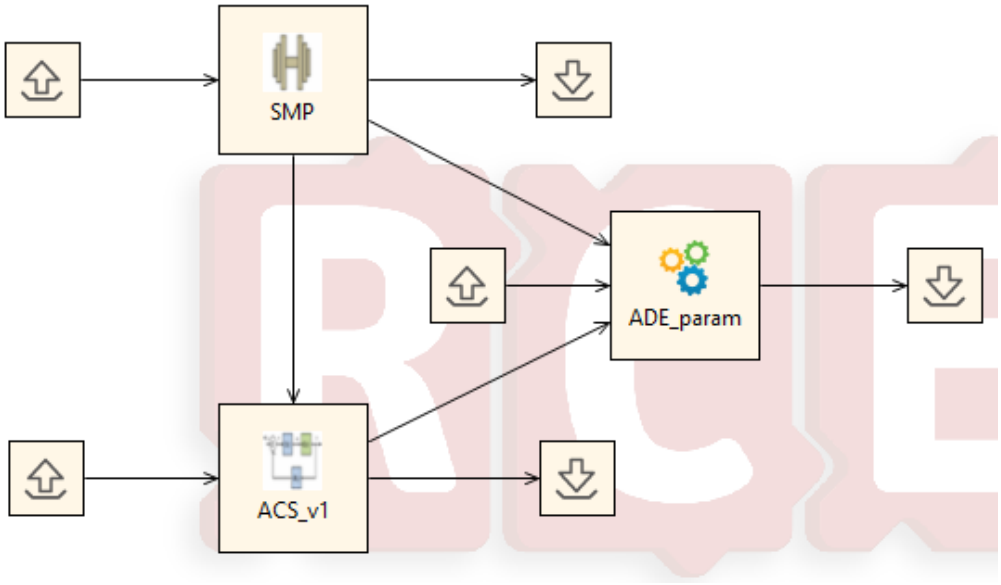


Fonte: Produção do autor.

O maior número de ferramentas-interconexões passa a exigir mais do desenvolvimento. Diversas partes, ao serem integradas ao ambiente, foram testadas para garantir que não havia problemas de funcionamento e que os resultados estavam sendo direcionados ao local especificado nos roteiros. Para isso foram criados processos isolados, mais

simples, sem objetivo de projeto a não ser observar a funcionalidade das ferramentas e de suas conexões, exemplificado na Figura 3.30.

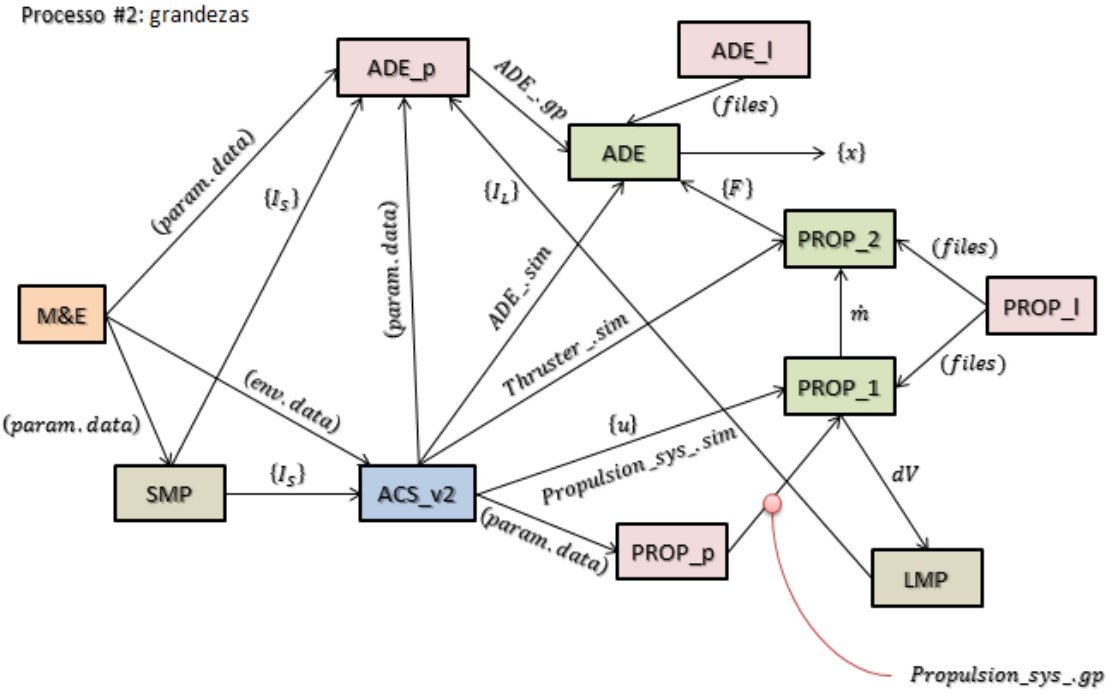
Figura 3.30– Um processo construído para teste de ferramentas.



Fonte: Produção do autor.

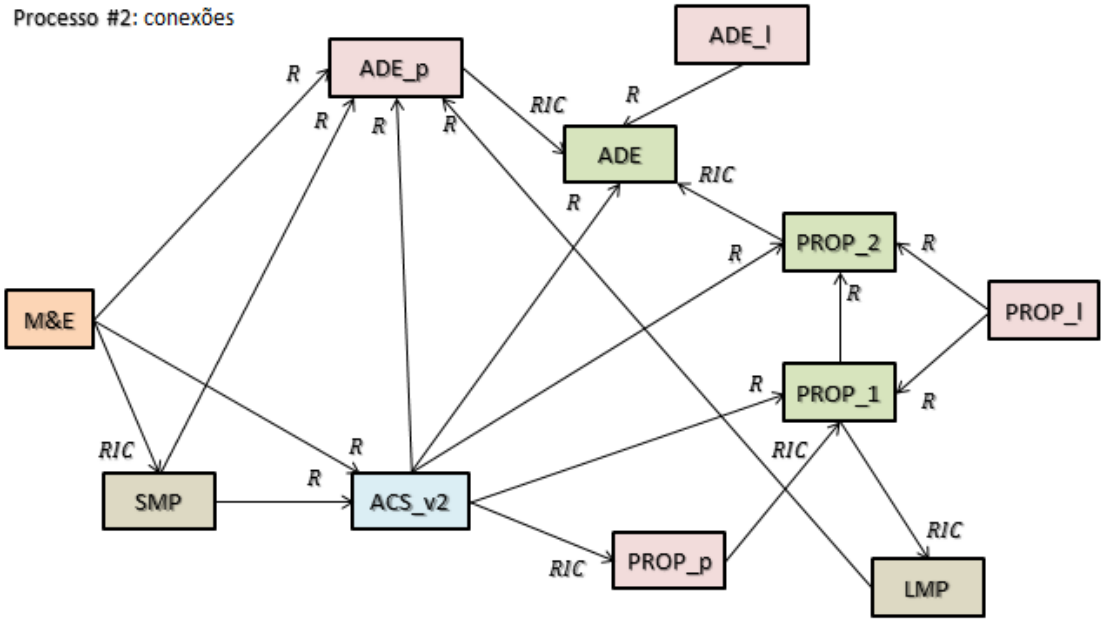
Da mesma forma, construiu-se um diagrama com grandezas fluindo e tipos de conexões, vistos nas Figuras 3.31 e 3.32.

Figura 3.31 – Diagrama simplificado do processo 2: grandezas.



Fonte: Produção do autor.

Figura 3.32 – Diagrama simplificado do processo 2: conexões.



Fonte: Produção do autor.

Os diagramas foram anexados a cada ferramenta envolvida no processo, com a denominação do mesmo. Assim, é possível saber qual a posição (função) da ferramenta em cada processo, dando uma ideia geral para aqueles que estejam realizando uma busca para um objetivo específico.

4 FORMULAÇÃO DO CASO DE ESTUDO

4.1. Características Gerais dos Microssatélites

Antes de caracterizar os microssatélites é necessário defini-los: microssatélites são uma família de satélites de dimensões e massa reduzidas contida dentro de um universo de satélites de dimensões e massa reduzidas (satélites miniaturizados).

De forma geral, os satélites miniaturizados se dividem em cinco subcategorias, mostradas na Tabela 4.1.

Tabela 4.1 – Classificação dos satélites de pequeno porte.

Classe de satélites	Faixa de massa [kg]
Minissatélites	100~500
Microssatélites	10~100
Nanosatélites	1~10
Picosatélites	0,1~1
Femtosatélites	<0,1

Fonte: Neri (1999).

PESSOTTA (2018)[1] também aponta as várias propostas de classificação de satélites na literatura técnica espacial, e corrobora a ausência de consenso entre elas. A Tabela 4.2, a seguir, é resultado do trabalho de Pessotta (2018), que será adotado nesta tese por ser considerada a classificação mais completa, conforme será descrito adiante.

[1] PESSOTTA, Fernando Antonio. **Uma estratégia para tratamento de falhas sistêmicas (FDIR) em ACDHs de satélites de pequeno e médio porte**. 2018 Tese (Doutorado em Engenharia e Tecnologia Espaciais, Área de Concentração em Engenharia e Gerenciamento de Sistemas Espaciais) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2018. Orientador: Dr. Marcelo Lopes de Oliveira e Souza.

Tabela 4.2 – Classificação de Satélites de Acordo com a Massa.

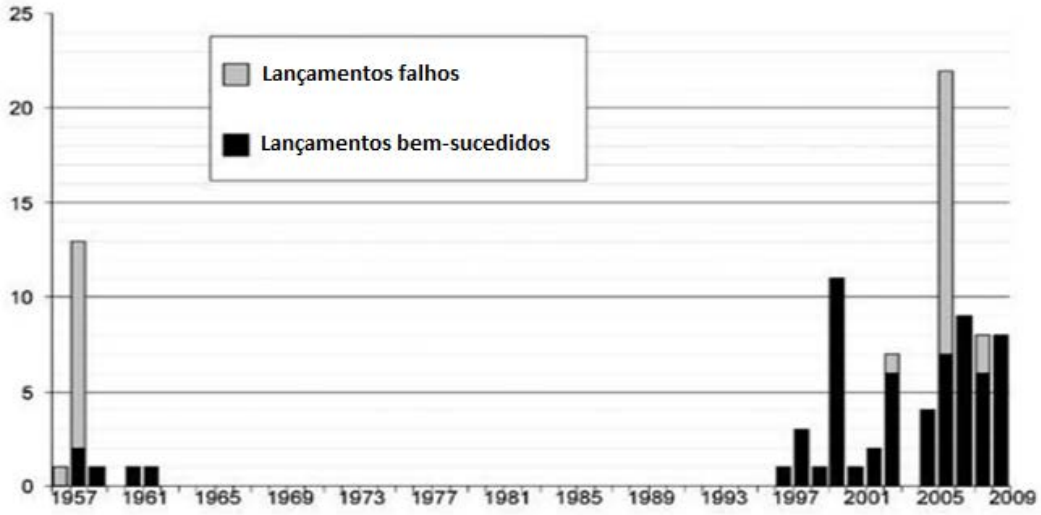
Classificação	Massa (kg)
Satélites Grandes	>1.000
Satélites Médios	500 – 1.000
Minissatélites /Satélites Pequenos	100 – 500
Microssatélites	10 – 100
Nanossatélites	1 – 10
Picossatélites	0,1 – 1
Femtossatélites	<0,1

No início da era espacial, a ex-URSS (com os satélites Sputnik, Luna e Kosmos), os EUA (com o Explorer e o Pioneer) e a França (com o Eole) trabalhavam com pequenos artefatos. Entre 1957 e 1987, foram lançados por volta de 270 pequenos satélites (TRW Space LOG, 1998), embarcando experimentos científicos. Isto representa cerca de 9% do total de lançamentos com sucesso no período.

O que vem ocorrendo nos últimos anos, especialmente após a década de 1990, é o aparecimento de uma nova geração de satélites de pequeno porte, os microssatélites (RADBONE; SWEETING, 1992), que incorporam novos avanços tecnológicos; estes, por sua vez, possibilitam a construção de sistemas espaciais mais confiáveis, de maior eficiência, e mais econômicos (NERI, 1999).

Em fins da década de 1990, renasce um interesse em minissatélites por parte de governos, com o aumento de lançamentos e sucesso de missões, visto na Figura 4.1.

Figura 4.1 – Histórico dos lançamentos de satélites miniaturizados desde 1957.

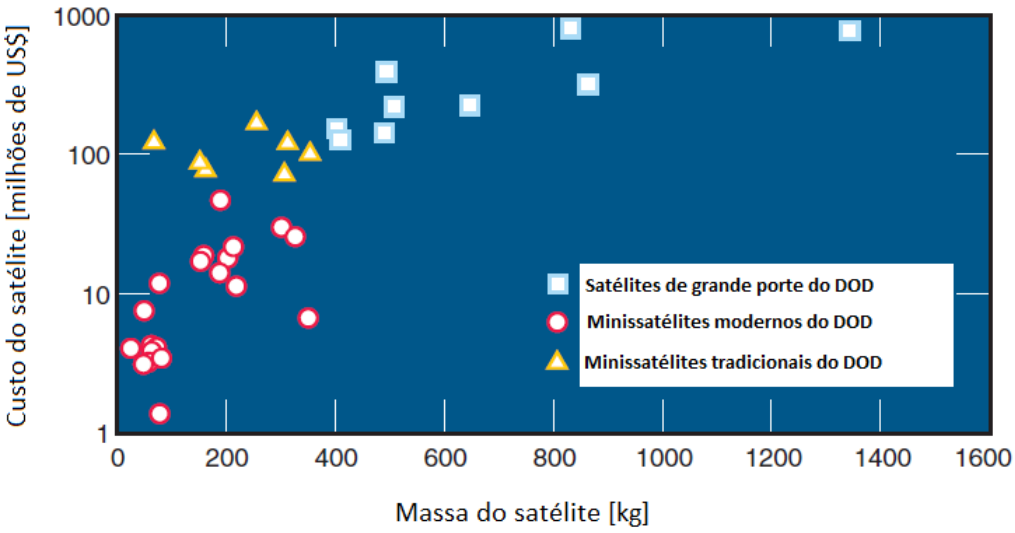


Fonte: Adaptado de Bearden (2001).

Esse novo desenvolvimento se baseou no uso de componentes preexistentes e tecnologias *"Commercial-Off-The-Shelf, COTS"* aliado à minimização dos esforços de desenvolvimento. No final dessa mesma década, o interesse pelos satélites miniaturizados se torna visível nos setores comerciais (devido ao seu baixo custo relativo) e acadêmicos (pesquisa de baixo custo e fins didáticos). Consequentemente, foi organizado um estudo comparativo por parte da *Aerospace Corporation* com o intuito de avaliar a **capacidade** e os **custos** de tais sistemas. Esse estudo gerou novasinformações, dentre elas um mapeamento do custo relativo à massa dos diferentes sistemas, desde os de maior dimensão até os de menor, que pode ser percebido graficamente na Figura 4.2.

Existem três classificações gerais: satélites grandes tradicionais (acima de 500 kg); satélites de pequeno porte tradicionais; e satélites de pequeno porte moderno. Todos os dados são relativos aos satélites do Departamento de Defesa norte-americano (*Department of Defense* – DoD). Compara-se a massa do satélite (kg) com o seu custo unitário (milhões de US$).

Figura 4.2 – Comparação dos custos relativos de satélites grandes [500 US$/kg], pequenos tradicionais [150 US$/kg] e pequenos modernos [100 US$/kg].



Fonte: Adaptado de Bearden (2001).

O gráfico revela que tanto satélites pequenos de concepção tradicional quanto moderna possuem a mesma faixa de massas. No entanto, o custo destes últimos é muito menor que o daqueles. Isso provavelmente se deve às novas tecnologias e formas de abordagem no desenvolvimento dos sistemas.

Feita a **definição**, a **classificação**, um breve **histórico** e a **relação baseada em dados estatísticos** entre tecnologias e custos desses veículos, resta apontar os desafios a serem superados nas próximas décadas – particularmente para os microssatélites, objeto do presente estudo. Quais são eles?

Do ponto de vista estrutural, as modificações são visíveis na geometria reduzida e na massa menor. Essas mudanças levam a uma modificação das propriedades inerciais que, por sua vez, influenciam as leis de controle (controle de atitude e órbita), pois a dinâmica veicular será diferente, porque certos fenômenos dinâmicos passam a ser relevantes no projeto – conforme será visto adiante.

Do ponto de vista da propulsão, podemos afirmar que, a depender da dimensão do microssatélite, os foguetes podem ser à base de monopropelentes ou bipropolentes para microssatélites maiores, ou fazer uso de propulsão elétrica ou à base de gás

comprimido, líquidos vaporizáveis (ex: butano ou dióxido de carbono) ou outros sistema de propulsão inovadores, desde que sejam simples (dimensão facilidade), baratos (dimensão custo) e escaláveis (dimensão adaptabilidade), i.e., microssatélites menores.

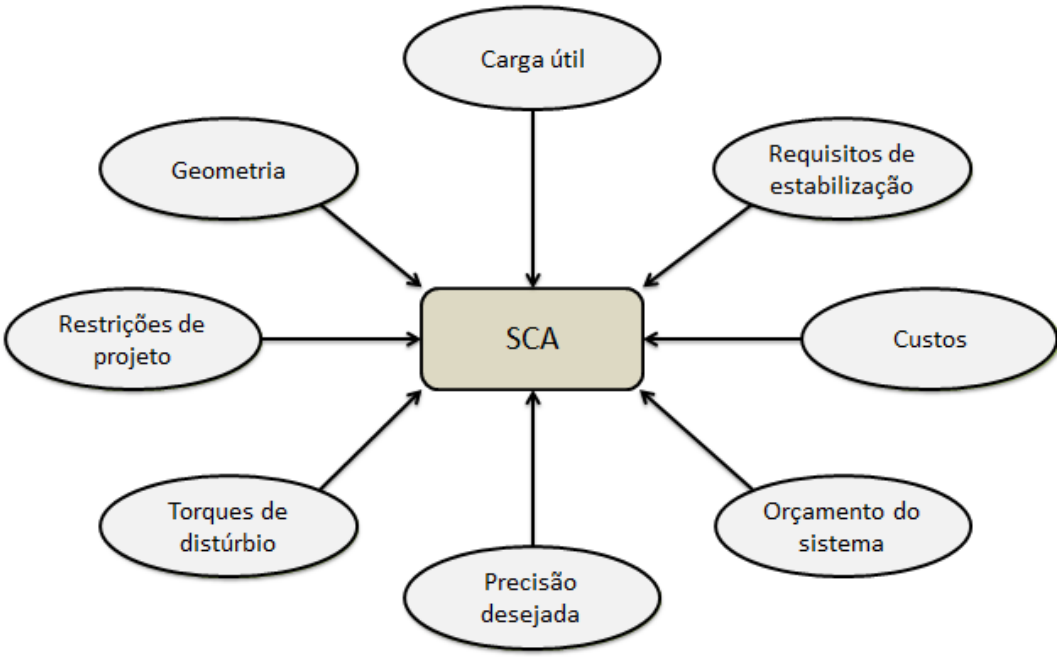
Quanto ao aspecto eletrônico, usam-se tecnologias mais recentes. Quando muito pequenos, os microssatélites carecem de fonte de energia ou tamanho para os *transponders* de rádios. Várias soluções foram propostas para contornar esse problema, tais como receptores a laser, antenas especiais e redes de comunicação satélite-para-satélite. No entanto, poucas dessas propostas foram usadas na prática.

4.1.1. **Leis de Controle**

Devido à sua massa e dimensões reduzidas, os microssatélites possuem algumas particularidades em seu sistema de controle de atitude. Uma das principais é a precisão de apontamento, que é menor em relação a satélites maiores, como consequência de perturbações no ambiente espacial. Além disso, existe a incerteza interna em relação à variação do momento de inércia. Estudos revelam que realizar um controle híbrido do tipo H_2/H_∞pode melhorar tanto a robustez da estabilidade quanto o desempenho calculados por meio do método LQR (YANG, 2002).

Atualmente, a precisão de apontamento é menor do que 1º; e é cada vez mais comum para satélites estabilizados por três eixos (SIAHPUSH, 1988). A seleção de um subsistema de controle de atitude depende de vários fatores, dentre os quais podemos citar: missão (objetivos), ambiente espacial, geometria, distribuição de massa e orçamento total. Segundo Siahpush(1988), podemos ilustrar essas relações de dependência do Subsistema de Controle de Atitude (SCA) graficamente, como visto na Figura 4.3.

Figura 4.3 – Fatores que influenciam o desenvolvimento de um SCA.



Fonte: Adaptado de Siahpush (1988).

Dentro do universo de satélites de porte convencional, o que diferencia o controle de um e de outro satélite são: os torques de perturbação, que são função de sua órbita; esta, por sua vez, é função da missão; a carga útil (também dependente da missão); e requisitos de estabilização (idem). Quando passamos para o universo de minissatélites, a **geometria**, os **torques de perturbação** e os **custos** passam a entrar como fatores diferenciados.

De modo geral, os satélites artificiais estão sujeitos a incertezas de dois tipos: **externas** (paramétricas) e **internas** (dinâmicas), como a variação do momento de inércia. O projeto do SCAO – e, consequentemente, do SCA – deve levar em consideração os dois tipos de incertezas, pois elas podem levar à perda de desempenho e menor robustez. Para satélites com massa inferior a 100 kg, a atitude é muito sensível às perturbações externas e variações no momento de inércia (PINHEIRO et al., 2012). Dessa forma, é necessário projetar um SCA que, simultaneamente, seja robusto e possua bom desempenho.

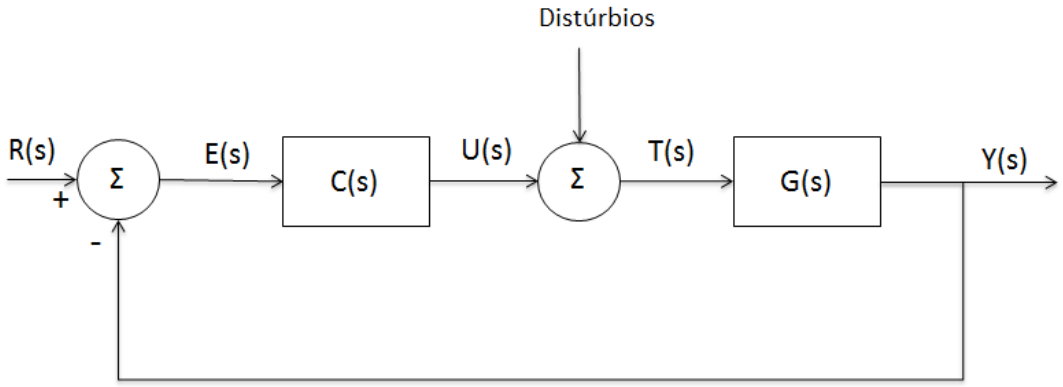
Existem muitos trabalhos sobre o controle de atitude de microssatélites na literatura. No entanto os torques externos, se considerados no modelos, tem uma representação

limitada. No artigo (LUIZ et al.,2005), o Regulador Linear Quadrático Gaussiano (LQG) é usado para controlar a atitude de um satélite; no entanto, não se consideram os torques de perturbação externos. No artigo (PITTELKAU, 1992), é projetado um controlador LQG para controle do ângulo de *pitch* no qual os torques externos são representados como uma constante combinada a uma senóide (PINHEIRO et al. 2012). Neste trabalho será usado um LQG (GREWAL; ANDREWS, 2001) para controle de atitude do microssatélite.

Quanto ao tipo de estabilização, tem-se, segundo a AMSAT (*Radio Amateur Satellite Corporation*), dois tipos de estabilização: a ativa e a passiva. A estabilização ativa é usada em satélites em que é necessária uma alta precisão no apontamento – o erro permitido é menor. A estabilização é concretizada por meio de motores, giroscópios e jatos; porém, as vantagens da utilização são essencialmente devidas ao aumento de precisão, porque as oscilações inerentes ao uso destes meios de estabilização diminuem o tempo de vida do satélite bem como aumentam consideravelmente o custo final do satélite – que já é elevado (PROCHNOW, 2007). A estabilização passiva faz uso de fenômenos físicos para efetuar a estabilização. O uso da estabilização aerodinâmica, estabilização por rotação, estabilização gravitacional e estabilização magnética permitiu diminuir a complexidade de sistemas do satélite bem como o seu custo. O uso destes métodos de estabilização permite um grau de precisão que varia de 1º a 4º. Segundo Siahpush (1988):"A configuração do veículo pode impor restrições de projeto no SCA na forma de orçamento de sistema para massa, potência e volume. Essas restrições são frequentemente impostas ao veículo, especialmente no caso de microssatélites."

Quanto à lógica de controle em si (diagramas, relações, métodos), trabalha-se com os mesmos conceitos da Teoria de Controle. Segundo Ogata (2010), "o diagrama de blocos de um sistema é a representação figurativa das funções realizadas pelos componentes e o fluxo de sinais". A Figura 4.4 apresenta o diagrama de blocos de um sistema de controle em malha fechada (realimentado), com uma entrada e uma saída (SISO – *Single Input Single Output*).

Figura 4.4 – Diagrama de blocos de um sistema com controleem malha fechada.



Fonte: Adaptado de Ogata (2010).

Os elementos listados na Figura 4.4 são:

a) O sistema físico G(s), também chamado de atuador + planta, que representa a entidade a ser controlada – no caso deste estudo, o satélite como um corpo rígido;
b) O sensor + controlador C(s), que contém a lógica de controle a ser implementada em G(s);
c) O sinal de controle U(s), que visa levar o sistema a um novo estado;
d) A referência R(s), que é o sinal desejado para a saída do sistema;
e) A saída do sistema Y(s), que é sua resposta ao longo do tempo a uma dada entrada U(s) – posicionamento e velocidade angular do satélite;
f) O sinal de erro E(s), que é a diferença entre o sinal de saída desejado, R(s), e o sinal de saída atual, Y(s);
g) O torque total T(s) impresso na planta G(s) é o resultado do torque de controle U(s) e dos torques externos (perturbações).

O sistema físico considerado será o atuador + a planta do ACS (micropropulsão, em modelo 1D físico); os sensores + as leis de controle estarão representadas no controlador C(s), que é a parte informacional.

A Transformada de Laplace indica que uma variável F(s) é dependente de outra variável, s, no domínio da frequência. De acordo com Ogata (2010), dada uma função f(t) dependente de uma variável no domínio do tempo, t, é possível encontrar uma

função F(s) equivalente no domínio da frequência. A expressão que indica essa transformação é dada por:

$$\boldsymbol{F(s) = \int_0^{\infty} f(t)\, e^{-st} dt} \tag{4.1}$$

A Função de Transferência entre um sistema com realimentação é a relação entre sua saída, Y(s), e sua entrada desejada, R(s):

$$\frac{\boldsymbol{Y(s)}}{\boldsymbol{R(s)}} = \frac{\boldsymbol{G(s) \cdot C(s)}}{\boldsymbol{1 + G(s) \cdot C(s)}} \tag{4.2}$$

Maiores desenvolvimentos serão feitos à medida que os requisitos do sistema forem levantados.

4.1.2. **Conceitos de Micropropulsão**

Sistemas propulsivos são um dos possíveis tipos de atuadores do subsistema de controle de um satélite. Eles se encontram dentre os sistemas de estabilização ativa – com maior custo, portanto. A escolha de utilizá-lo como atuador do ACS irá depender da missão do satélite (ex: sensoriamento remoto, telecomunicações, científico, etc) e de sua natureza (dimensões, massa, geometria, custo). Sua missão irá, em larga medida, determinar sua órbita que, por sua vez, irá se desdobrar em (por exemplo) períodos expostos à luz solar, determinando assim os requisitos do sistema de suprimento de energia. Apesar de conhecidas, é importante destacar essas relações que se desdobram e se relacionam a título de ilustrar que alterações em fases iniciais do projeto podem levar à redefinição dos requisitos de sistema (tolerância a erros de apontamento, tempo máximo de resposta, período de uso contínuo máximo da bateria), que podem alterar a natureza dos subsistemas (ex: tipo de controle de atitude, uso ou não de propulsão, qual tipo de propulsão, momentos de inércia, órbita, etc), que levam à alteração de parâmetros (ex: comprimentos e diâmetros dos dutos, filtros, tipo de válvulas) que, por sua vez, influenciam as variáveis de interesse (empuxo, impulso específico, tempo de resposta).

A propulsão química é muito comum em satélites convencionais, especialmente sistemas mono e bi propelentes (gás quente) e por gás comprimido (gás frio). A Figura

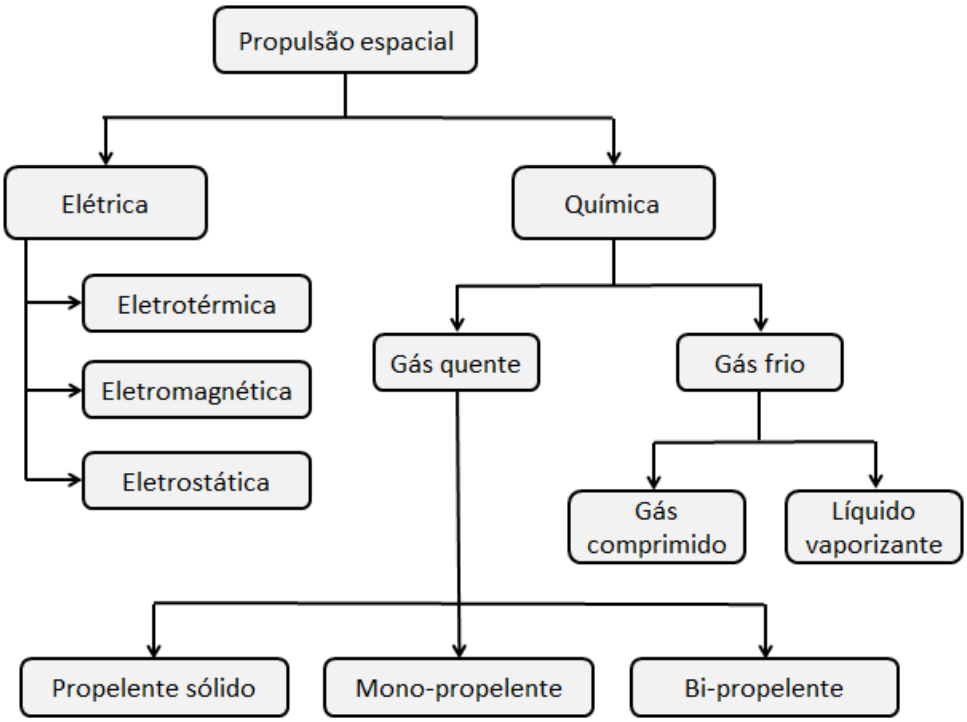
4.5 desdobra os tipos de sistemas de propulsão comumente adotados na indústria aeroespacial, divididos em propulsão elétrica e química.

Sistemas propulsivos de microssatélites devem ser capazes de prover níveis de empuxo e impulso extremamente baixos (na ordem de 100 μN e ~0,100 N.s, respectivamente) e, concomitantemente, satisfazer estreitas restrições de geometria, massa, consumo de energia e custo (HITT et al., 2001). Devido às dimensões reduzidas, fenômenos como radiação solar e falta de uniformidades gravitacionais (HITT et al., 2001) começam a influenciar o apontamento do satélite, levando à necessidade de um sistema de controle particular. Para contornar esses problemas há a necessidade de correções periódicas (ou contínua) de empuxo.

Microssatélites possuem requisitos de sistema (propulsivo) singulares: empuxo ou impulso específico baixos relativos a veículos de maior porte, tanto para manobras orbitais quanto controle de atitude, segundo Blandino e Cassady(1998) e Pollardet al.(1999). Além destas limitações, existem outras referentes à massa, volume e potência (Tabela 4.2). Dentre os parâmetros, existe o impulso – que mensura o total de empuxo aplicado ao longo de um intervalo de tempo, dado em $N \cdot s$; e o impulso específico– que mede a eficiência desse impulso, comparando-o relativamente ao produto da vazão mássica dos gases pela sua velocidade de ejeção, dado em segundos (ERICHSEN, 2006).

A maior parte da tecnologia propulsiva não é capaz de atender às restrições impostas ao projeto de microssatélites (MUELLER, 1997). Com isso, começaram a surgir estratégias para desenvolver novas tecnologias de micropropulsão. Dentro desse quadro, Sistemas Micro-Eletro-Mecânicos (*Micro Electro Mechanical Systems* – MEMS) direcionados a sistemas propulsivos têm sido identificados como potenciais soluções (KETSDEVER; MUELLER, 1999). Propulsores que utilizam tecnologias MEMS são interessantes por oferecerem a possibilidade de miniaturizar efetivamente diversos componentes utilizados em sistemas propulsivos de satélites – no entanto, pode ser necessário o uso de física de microescala em alguns componentes.

Figura 4.5 – Tecnologias de propulsão espaciais mais comuns.



Fonte: Adaptado de Erichsen (2006).

Tabela 4.2 – Faixas de operação para parâmetros importantes de subsistemas propulsivos de microssatélites.

Parâmetro do subsistema propulsivo	Faixa de operação
Empuxo	1-1000 μN
Impulso	1-100 Ns
Impulso específico	160 s
Massa	< 0,1 kg
Consumo de energia	< 1 W
Volume	< 1 cm^3
Temperatura de operação	< 1700 K

Fonte: Adaptado de Hitt et al. (2001).

As vantagens de desenvolver tecnologias MEMS para sistemas propulsivos são diversas. Dentre elas podemos, segundo Hitt et al.(2001), citar as seguintes:

a) Os componentes podem ser feitos na escala de mícron;
b) O avanço contínuo nas técnicas de fabricação de micro e nano componentes podem viabilizar aparatos com nano-propulsão;
c) Propulsores MEMS podem ser produzidos em massa com um baixo custo relativo;
d) Através dos MEMS são possíveis novos tipos de abordagens, que permitirão extrair benefícios de aparatos com micro geometria e microfísica – opções que não estavam no horizonte de possibilidades para o projeto de propulsores em escala macro.

A Tabela 4.3 dá uma visão geral de alguns conceitos de micropropulsão sendo desenvolvidos por diversos grupos de pesquisa ao redor do mundo. A finalidade desta tabela é fornecer pistas que possam indicar qual o melhor tipo de tecnologia para atender objetivos e restrições especificados – ela não dá um nível de compreensão mais profundo.

Cada tipo de missão irá fornecer um conjunto de restrições. Cada tipo de tecnologia possui suas vantagens e limitações. Micropropulsores elétricos (ex: pulsos de plasma, motores iônicos, *Field-Emission Electrical Propulsion* - FEEP) são capazes de fornecer níveis extremamente baixos de empuxo mas possuem dimensões e massa consideráveis. Além disso, os requisitos de potência dessa tecnologia são muito exigentes (altos) para propulsão de microssatélites – cuja potência de barramento total não ultrapassa os 10 W. Nos fenômenos químicos, a propulsão sólida é limitada essencialmente por sua natureza: o reuso não é possível uma vez acionado.

Em micro e nano satélites, a localização dos propulsores pode produzir perturbações adicionais na atitude do satélite, que devem ser consideradas em cada jato (HITT et al., 2001). Propelentes químicos do tipo líquido (ex: hidrazina N2H4, peróxido de hidrogênio) tem sido extensivamente usados em satélites devido à sua alta quantidade de energia química por unidade de massa (J/kg). Esses benefícios podem ser conjugados em dispositivos que utilizam a tecnologia MEMS devido à relação de escala recíproca entre volume e superfície. Essa relação é explicada a seguir.

Tabela 4.3 – Tecnologias de micro propulsão sendo desenvolvidas por institutos de pesquisa ao redor do mundo.

Tipo de propulsão	Impulso específico [s]	Empuxo [N] (Impulso [μ Ns])	Instituto
H2O2 monopropelente	160	1-1000	NASA/GSFC,
Gás frio	40–80	500 - 50 000 (0,5)	MIT, NASA/JPL, Aerospace Corp.
Sólido digital	200	(10 – 100 000)	NASA/GRC, TRW, CNES
Bi propelente turbo bomba	300	15x 10E6	MIT, NASA/GRC
Bi propelente digital	200	3 – 50	Princeton, Honeywell
Jato por resistência	45–100	100 - 1000	AFRL, USC, Aerospace Corp.
Líquido vaporizante	75–125	1 – 100	NASA/JPL
FEEP	17 000	10 – 200	SRI, Itália, MSU
Micro coloidal	450–1350	20 – 100	Stanford, MIT
Micro-PPT	800-1000	(0,1 – 10)	UI, NASA/GSFC, EPLI, Primex, NASA/GRC
Motor de íon	1400-2000	0,1 – 10	Aerospace Corp. NASA/JPL

Fonte: Adaptado de Hitt et al. (2001).

A massa de propelente é uma quantidade relacionada ao volume, que se reduz cubicamente em relação à dimensão linear; por outro lado, o empuxo é uma quantidade relacionada à superfície, que é função da área de saída do bocal, ou seja, ele se reduz quadraticamente em relação à dimensão linear. Consequentemente a relação entre empuxo e massa de propelente aumenta linearmente com a diminuição da estrutura e componentes dos subsistemas. Podemos visualizar essa relação utilizando um modelo matemático, que relaciona a massa de propelente com o volume – e pressão – do tanque (OLIVA, 2012):

$$\boldsymbol{m_p = \frac{V_{N_2}^0 \cdot P_{MEOP} \cdot \rho_P}{P_t} - \rho_p \cdot (V_t - V_P^0)} \qquad (4.3)$$

Em que:

a) $\boldsymbol{V_{N_2}^0}$: volume inicial de gás Nitrogênio no tanque;

b) $\boldsymbol{P_{MEOP}}$: pressão máxima de operação (*Maximum Established Operating Pressure* – MEOP);

c) $\boldsymbol{\rho_P}$: densidade do propelente;

d) $\boldsymbol{V_P^0}$: volume inicial de propelente no tanque;

e) $\boldsymbol{P_t}$: pressão no tanque (no tempo);

f) $\boldsymbol{V_t}$: volume do tanque (no tempo);

g) $\boldsymbol{m_p}$: massa de propelente no tanque (no tempo).

E outro que relaciona empuxo com área do bocal:

$$\boldsymbol{F = \lambda_e \cdot \dot{m} \cdot V_e + (P_S - P_a) \cdot A_S} \quad (4.4)$$

Em que,

a) $\boldsymbol{\lambda_e}$: fator de correção de empuxo (função da geometria do bocal);

b) $\boldsymbol{\dot{m}}$: vazão mássica de propelente (no tempo);

c) $\boldsymbol{V_e}$: velocidade de ejeção dos gases (no tempo);

d) $\boldsymbol{P_S}$: pressão de saída do bocal;

e) $\boldsymbol{P_a}$: pressão do ambiente (espacial);

f) $\boldsymbol{A_S}$: área de saída do bocal;

g) $\boldsymbol{F}$: empuxo.

Nessa análise, o interesse é relacionar variáveis geométricas com as variáveis dinâmicas "empuxo" e "massa de propelente".

Como a área do bocal varia com o quadrado do diâmetro, o empuxo irá variar da mesma forma – se considerarmos que a vazão e velocidade de escape permaneçam próximas. À luz do mesmo raciocínio, podemos concluir que a massa de propelente no tanque irá variar com o cubo do diâmetro do tanque, pois ela é diretamente proporcional ao volume de propelente. Logo, o empuxo de um microssatélite irá diminuir menos em relação à diminuição da massa de propelente. Ou seja, nos satélites de pequeno porte, o empuxo é relativamente maior do que a massa de combustível se forem mantidas as mesmas proporções dos satélites convencionais.

Outra vantagem da propulsão química líquida é que ela oferece uma grande faixa de valores de impulso total, específico e empuxo (ver Tabela 4.3) em relação à sólida. Quanto à propulsão química em geral, podemos elencar vantagens, destacadas por Hitt et al.(2001):

a) Estabilidade química e consequente segurança;
b) Compatibilidade entre materiais e reagentes químicos;
c) Prazo de validade relativamente longo.

Observando as particularidades citadas, construiu-se uma tabela listando as características desejáveis em subsistemas micropropulsivos para satélites de 10 a 100 kg (1ª coluna da Tabela 4.4) e sistemas propulsivos existentes, que são enumerados nas colunas restantes da Tabela 4.4, baseado no universo de valores apresentado na Tabela 4.3.

Para satélites do porte estudado aqui (10 a 100 kg) os sistemas monopropelentes (ainda) são os mais adequados para micropropulsão. Apesar de algumas dificuldades em atingir formas geométricas específicas a nível microscópico (μm) no processo de produção, e com isso existirem efeitos de viscosidade que prejudicam a eficiência dos propulsores (HITT et al., 2001), os outros conceitos de propulsão ou são limitados pela falta de reusabilidade / controle (sólida), ou são difíceis de fabricar, tem peso relativamente alto e consomem muita potência (elétrica), ou apresentam faixa limitada de empuxo (gás frio, líquido vaporizante). Isso indica que sistemas líquidos bipropelentes ainda são mais adequados a esse tipo de veículo e missão.

Tabela 4.4 – Mapeamento de vantagens desejáveis em subsistemas propulsivos para microssatélites.

QUALIDADES PRESENTES EM DIVERSOS CONCEITOS PROPULSIVOS PARA MICROSSATÉLITES		CONCEITOS DE MICROPROPULSÃO							
		QUÍMICA					ELÉTRICA		
		FRIA		QUENTE					
		Gás frio	Líq. Vaporizante	Monopropelente	Bipropelente	Sólida	Eletrotérmica	Eletromagnética	Eletrostática
CARACTERÍSTICAS DESEJÁVEIS [medidas de efetividade]	Baixo empuxo	x		x	x	x	x	x	x
	Dimensão e massa reduzidas	x	x	x					
	Faixa ampla: empuxo [N]		x	x		x			
	Faixa ampla: Impulso [Ns]		x	x		x			
	Faixa ampla: Impulso específico [s]		x	x		x			
	Alta razão empuxo-propelente			x					
	Reusabilidade	x	x	x	x		x	x	x
	Baixo consumo energético [W]	x	x	x					
	Fabricação simples	x							

Fonte:Produção do autor.

Observemos as características de seus processos termoquímicos (câmara catalítica) e termodinâmicos (bocal convergente-divergente).

O catalisador mais usado é Ag com pureza 100%. Existem dois tipos de propelentes (reagente) muito usados: o peróxido de hidrogênio ($\boldsymbol{H_2O_2}$) e a hidrazina ($\boldsymbol{N_2H_4}$). A reação envolvendo esta foi estudada, modelada e analisada exaustivamente em trabalho anterior (OLIVA, 2012). Mas ambas as reações são de mesma natureza e, em linhas gerais, tratam do mesmo fenômeno: a geração de calor, com liberação e produtos, descritas analiticamente por uma reação química exotérmica.

A decomposição de 100% de peróxido de hidrogênio líquido gera água e oxigênio.

$$\boldsymbol{2H_2O_2(l) \rightarrow 2H_2O(l) + O_2(g) + Q}$$

A reação pode ser modelada através de uma cinética química de primeira ordem, com velocidade altamente dependente da temperatura, pureza e concentração do propelente. Dados empíricos revelam que $\boldsymbol{H_2O_2}$ com 85% de pureza libera 586 kcal/g a 25 ºC (HITT et al., 2001).

As perdas por condução podem ser consideradas desprezíveis devido à grande diferença na escala de tempo – como a reação exotérmica é da ordem de centésimos de segundo e a condução se dá em segundos, as perdas por esta podem ser consideradas desprezíveis no modelo da câmara catalítica.

O bocal convergente-divergente para micropropulsores apresenta algumas particularidades devido a limitações nas tecnologias de fabricação. A primeira delas diz respeito à separação de fluxo nas extremidades do bocal (divergente). Isso ocorre porque a microfabricação desse componente é limitada em termos de geometria: não é possível construir uma seção de saída arredondada (ex: cônica ou em forma de sino), que é aquela geralmente empregada em bocais – por obter o melhor aproveitamento de escoamento possível. A seção de saída é retangular, isto é, não possui simetria tridimensional em relação a um eixo, o que gera irregularidades no empuxo. Este tende a ter menos componentes uniaxiais, normais à seção de saída, o que leva a uma redução de eficiência do bocal – tem-se menos empuxo para uma mesma massa de propelente, considerando a proporcionalidade entre macro e micropropulsão. Outro empecilho decorrente é a separação do fluxo nas vizinhanças da garganta, além do surgimento de ondas de expansão na parte divergente do bocal (HITT et al., 2001).

Outro aspecto crítico em sistemas MEMS diz respeito aos fluxos supersônicos do bocal, que podem ser influenciados por efeitos viscosos. Isso é melhor visualizado através da relação de Reynolds:

$$\boldsymbol{Re} = \frac{\boldsymbol{\rho \cdot U \cdot D_H}}{\boldsymbol{\mu}} = \frac{\boldsymbol{\dot{m}}}{\boldsymbol{\mu D_H}}$$

Em que $\boldsymbol{D_H}$ é o diâmetro hidráulico, $\boldsymbol{U}$ é a velocidade de ejeção do gás, $\boldsymbol{\rho}$ a densidade e $\boldsymbol{\mu}$ a viscosidade do fluido. Para micropropulsores que usam peróxido de hidrogênio, a estimativa do número de Reynolds considerando um fluxo mássico de 400 $\mu \cdot \boldsymbol{g \cdot s^{-1}}$ é de 127 a 140 (HITT et al., 2001). Quando esse número atinge valores como esses

(relativamente baixos) as perdas por forças viscosas são significativas. O resultado é a reversão de parte do fluxo mássico, com consequente perda de eficiência e empuxo. Maiores esclarecimentos sobre efeitos viscosos em microbocais podem ser encontrados nos trabalhos de ZELESNIK et al. (1994), BAYT eBREUER (1997) e ALEXEENKO et al. (2001), em termos numéricos; ou, em termos experimentais, nos trabalhos de Bayt et al. (1997) e Choughurri et al. (2001).

Tendo ressaltado as características geométricas e de massa, além dos conceitos de micropropulsão, pode-se iniciar o levantamento dos interessados no desenvolvimento do sistema em questão de forma orientada.

4.1.3. **A Plataforma MultiMissão**

Este tópico delineia em geral a geometria da Plataforma MulltiMissão (PMM) com o intuito de orientar a escolha dos parâmetros de entrada (materiais e geometria) do sistema a ser usado como caso de estudo.

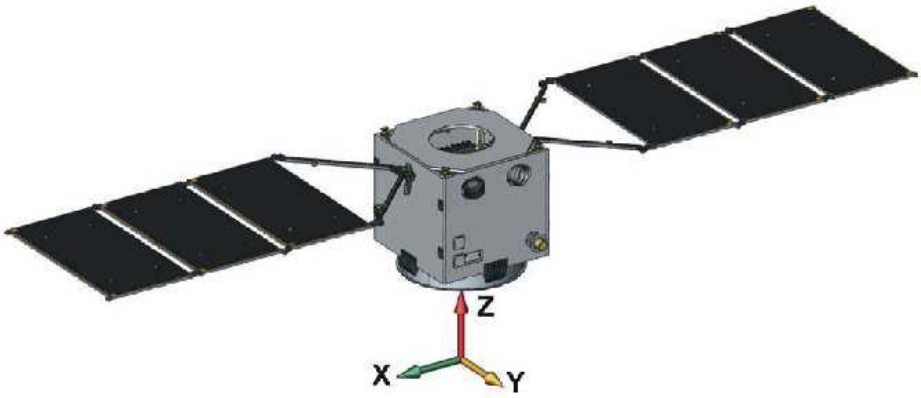
O projeto da PMM é um conceito inovador em arquitetura de satélites, que consiste em reunir em uma única plataforma todos os equipamentos essenciais à operação de um satélite, independente de sua órbita e missão. Nessa arquitetura há uma separação entre a plataforma e o módulo de carga útil. Enquanto esta carrega os dispositivos e subsistemas diretamente relacionados à missão (ex: câmeras), aquele é o conjunto de subsistemas indiretamente relacionados à missão, cuja função é dar suporte à carga útil. Ou seja, a estrutura, que abriga todos os componentes; o subsistema de controle de atitude e órbita, que garante o apontamento correto, juntamente com o subsistema propulsivo; o subsistema de suprimento e armazenamento de energia elétrica para os computadores de bordo, entre outros, todos fazem parte desse grupo.

A separação física entre a PMM e a sua carga útil permite que ambos sejam desenvolvidos, manufaturados e testados separadamente, antes da integração e teste final do satélite, possibilitando assim uma agilização do processo.

A Figura 4.6 ilustra a configuração da PMM em órbita.

A plataforma tem como finalidade o apoio a atividades relacionadas à observação terrestre, experimentos científicos, e a comunicação – todas em LEO (*Low Earth Orbit*).

Figura 4.6 – PMM em configuração em órbita



Fonte: Silva (2017).

A PMM é constituída dos seguintes subsistemas:

a) Estrutural: provê suporte físico e proteção a todos os demais subsistemas;

b) Suprimento e Armazenamento de Energia Elétrica: converte energia da radiação solar em energia elétrica através de células fotovoltaicas, podendo armazená-la em baterias e suprir os subsistemas (plataforma e carga útil);

c) Controle Térmico: viabiliza a distribuição térmica, direcionando os fluxos de calor para aquecer ou resfriar equipamentos com o intuito de mantê-los dentro de sua faixa operacional térmica e, consequentemente, em condições operacionais plenas;

d) Controle de Atitude e Gerenciamento de Dados: responsável pela orientação do satélite e seu posicionamento orbital, estabilizado em três eixos. O subsistema também armazena e processa dados através do computador de bordo;

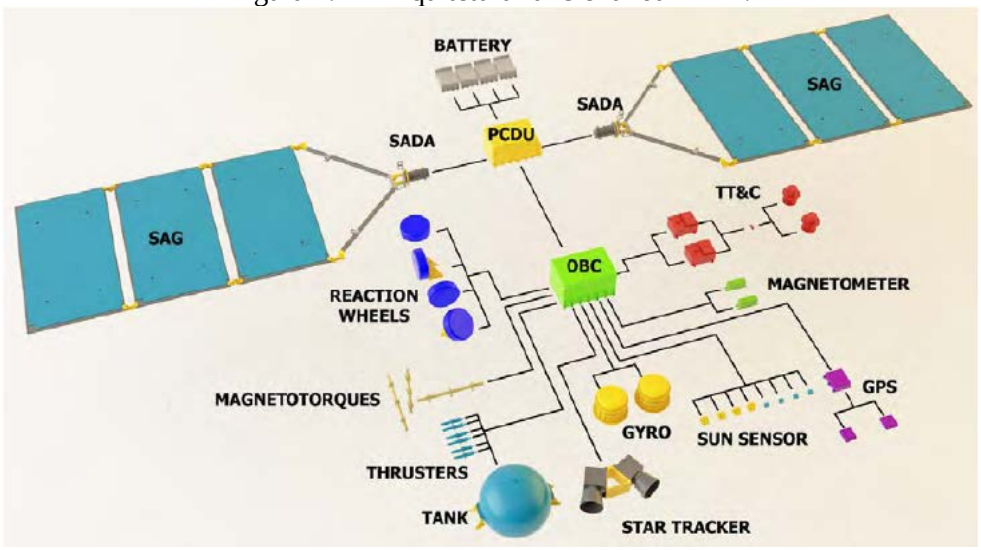
e) Propulsivo: atuador responsável pela manutenção da atitude e órbita. Viabiliza as informações do subsistema de controle de atitude e órbita;

f) Telemetria e Telecomando: provê comunicação entre a PMM e postos de comunicações na Terra.

Os subsistemas descritos anteriormente podem ser visualizados no diagrama da Figura 4.8, que revela a arquitetura funcional da PMM, isto é, as relações entre os subsistemas e seus componentes.

Descritos os seis subsistemas que compõem o sistema PMM, serão apresentados dados relevantes à estrutura da PMM, com sua geometria, massa e propriedades inerciais, que servirá de base para obter a estrutura preliminar do microssatélite.

Figura 4.7 – Arquitetura funcional da PMM.



Fonte: Souza (2009).

O subsistema de Controle de Atitude e Gerenciamento de Dados é conhecido como *"Attitude Control and Data Handling"* (ACDH) e implementa as seguintes funções a bordo do satélite: (ba) Controle de Atitude e Órbita; (a) Gerenciamento de Dados..
Observando em maior detalhe o subsistema de Controle de Atitude e Órbita, pode-se elencar as seguintes funções específicas:

a) Controle de atitude estabilizado em três eixos no Modo Nominal, permitindo o apontamento para a Terra, "Anti-Terra", Inercial e para o Sol;

b) Controle no Modo de Contingência, cujo objetivo é manter o apontamento adequado, garantindo a segurança do sistema como um todo, logo após a fase de lançamento ou após alguma falha;

c) Controle do posicionamento dos painéis solares;

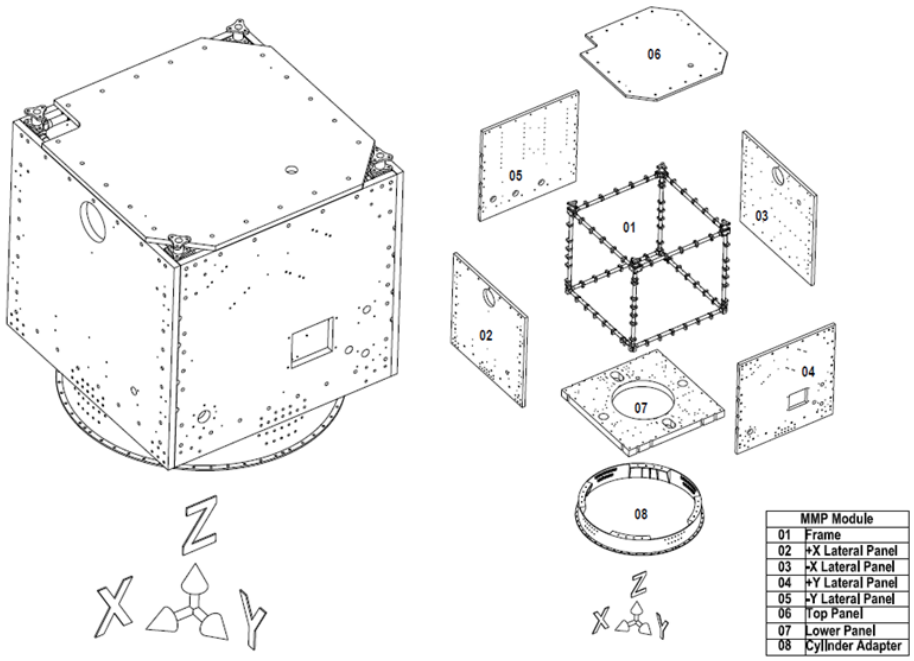
d) Controle dos propulsores, via comandos, para aquisição e manutenção de órbita;

e) Dessaturação das rodas de reação através de bobinas magnéticas ou dos propulsores;

As propriedades inerciais e geometria do satélite Amazônia 1 (PMM e módulo de serviço) foram obtidas a partir de dados técnicos disponibilizados por servidores do INPE responsáveis pela integração e testes do satélite Amazônia-1.

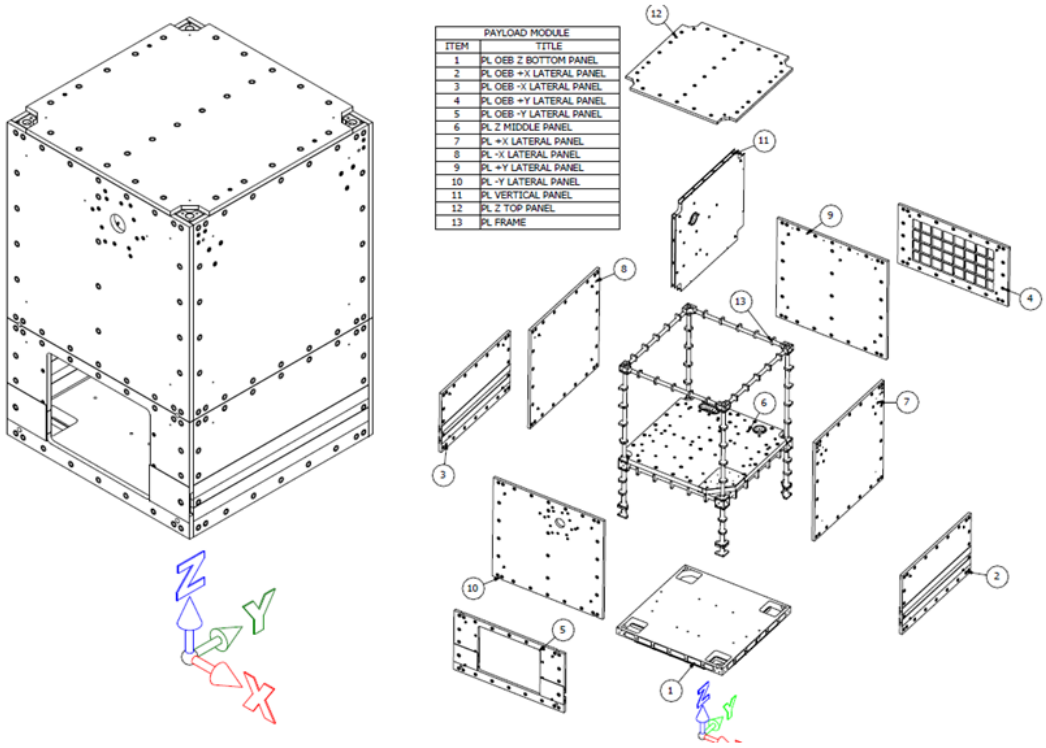
O módulo de serviço (PMM) é composto de 8 partes, Figura 4.8, formando sua estrutura. O módulo de carga útil é mais elaborado, composto de 13 partes, Figura 4.9. As dimensões de cada peça se encontram no apêndice deste trabalho, assim como informações referentes ao tipo de material.

Figura 4.8 – Vista em perspectiva dimétrica (conjunto montado e explodido) do módulo de serviço da PMM.



Fonte: Silva (2017).

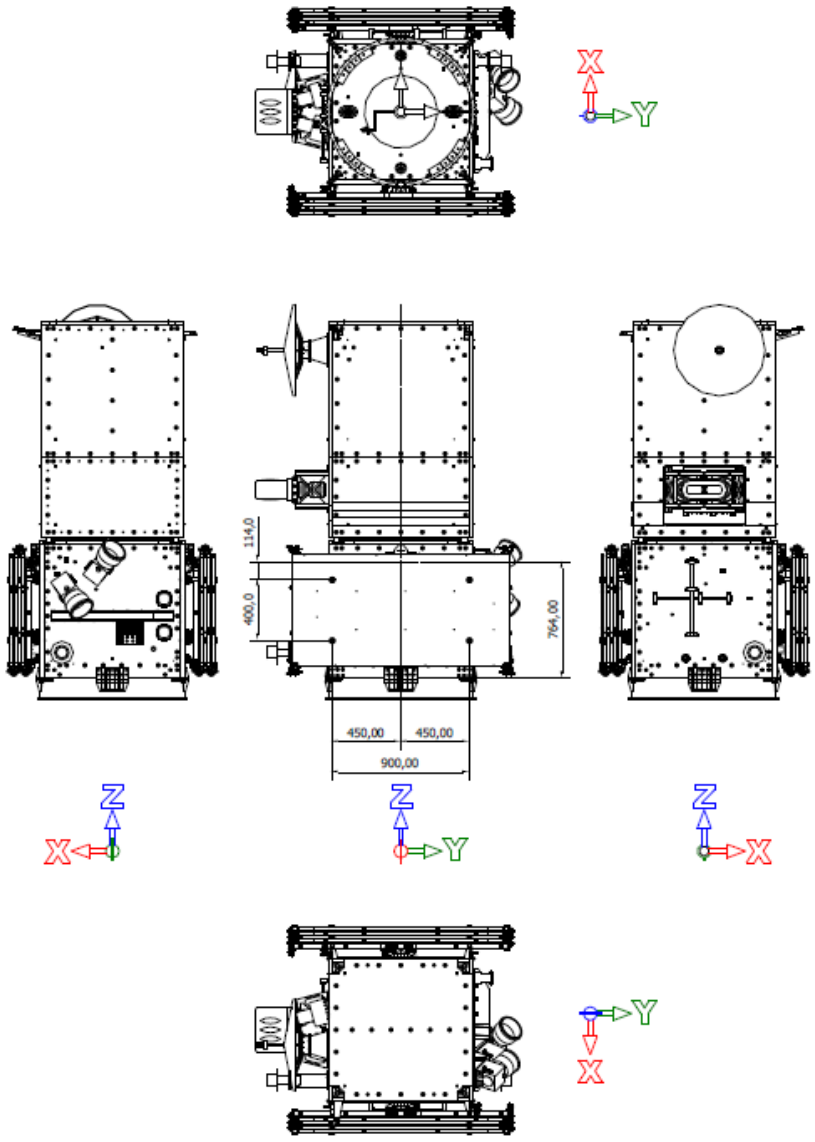
Figura 4.9 – Vista em perspectiva dimétrica (conjunto montado e explodido) do módulo de carga útil da PMM.



Fonte: Silva (2017).

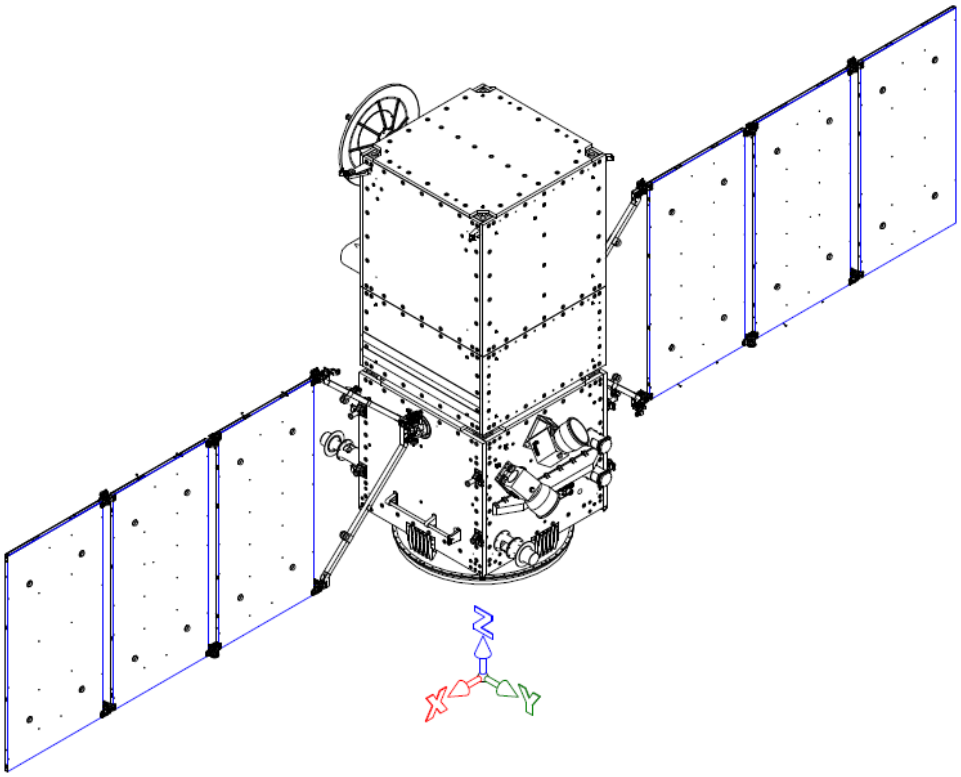
O envelope dimensional ao qual o satélite está circunscrito deverá ser adaptado para um lançador de microssatélites – possivelmente o VLM. A Figura 4.9 mostra cinco vistas do Amazônia-1 em configuração de lançamento; as Figuras 4.12 e 4.13 mostram, respectivamente, a vista 3D, em perspectiva isométrica, e as três vistas em 1º diedro, do satélite em configuração de órbita (operação).

Figura 4.10 – Vistas do satélite Amazônia-1 em configuração de lançamento.



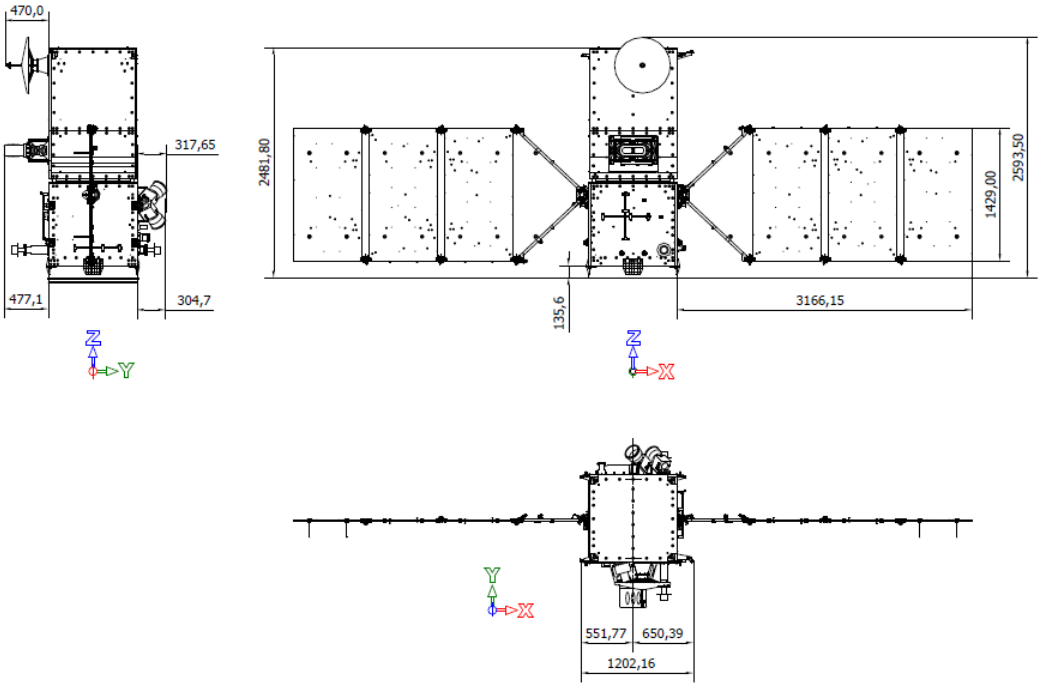
Fonte: Silva (2017).

Figura 4.11 – Amazônia-1 em configuração de órbita – perspectiva isométrica.



Fonte: Silva (2017).

Figura 4.12 – Três vistas do Amazônia-1 em configuração de órbita.



Fonte: Silva (2017).

Os dados apresentados servirão de base para se chegar as dimensões do microssatélite, que deve ter configuração semelhante ao Amazônia-1 (módulo MultiMissão, PMM, adaptável a diversas missões, encaixado num módulo de serviços, diferente para cada missão).

4.2. **Modelos Simbólicos**

4.2.1. **Ambiente Espacial**

Para modelar o ambiente espacial deve-se considerar que características deste influem direta ou indiretamente, de forma relevante, no desenvolvimento do modelo do subsistema proposto. Conforme a literatura apresentada e comentada na Seção 4.3, podemos elencar três submodelos de ambiente a serem considerados para o SCA (e outros subsistemas):

a) Campo gravitacional;
b) Campo magnético;
c) Torques de pertubações (aerodinâmicos).

Serão analisados cada um dos torques, partindo dos fenômenos que os causam.

Torque gravitacional

Se o campo gravitacional que age sobre o corpo é uniforme, então o centro de gravidade se torna o centro de massa, e o torque gravitacional sobre o centro de massa é zero. Mas, no espaço, sabe-se que o campo gravitacional não é uniforme. Essas variações irão causar um torque em torno do centro de massa do satélite.

Supondo que o centro de massa do satélite está em uma órbita kepleriana circular e a Terra é esférica, o torque do gradiente de gravidade em torno dos eixos do satélite é dado por YANG e SUN (2005).

$$\boldsymbol{T_{g1} = 3n^2(I_3 - I_2) \cdot \cos^2(\theta) \cdot \cos(\varphi) \cdot \mathrm{sen}(\varphi)} \quad (4.5)$$

$$\boldsymbol{T_{g2} = 3n^2(I_3 - I_1) \cdot \cos(\theta) \cdot \cos(\varphi) \cdot \mathrm{sen}(\theta)} \quad (4.6)$$

$$\boldsymbol{T_{g3} = 3n^2(I_1 - I_2) \cdot \cos(\theta) \cdot \mathrm{sen}(\varphi) \cdot sen(\theta)} \quad (4.7)$$

em que $\boldsymbol{n}$ é a frequência orbital do satélite, dada por $\boldsymbol{n = \sqrt{\mu/R^3}}$, e os ângulos $\boldsymbol{\theta}$, $\boldsymbol{\varphi}$ são ângulos de Euler. Para efeitos de simplificação do modelo foi considerado o satélite como um corpo rígido descrevendo uma órbita circular. Neste caso a velocidade tangencial é dada por $\boldsymbol{v = \sqrt{\mu/R^3}}$. Consideram-se pequenas variações nos ângulos de Euler (Seção 5.3), ou seja: $\boldsymbol{\theta \cong \varphi \cong 0}$. Consequentemente, os componentes trigonométricos da equação (4.4) a (4.6) se tornam: $\boldsymbol{sen(\theta) \cong \theta}$, $\boldsymbol{sen(\varphi) \cong \varphi}$, $\boldsymbol{\cos(\varphi) \cong 1}$ e $\boldsymbol{\cos(\theta) \cong 1}$. Logo o modelo do gradiente gravitacional não apenas passa a ser linear, mas invariante no tempo:

$$\boldsymbol{T_{g1} = 3n^2(I_3 - I_2) \cdot \varphi} \quad (4.4)$$

$$\boldsymbol{T_{g2} = 3n^2(I_3 - I_1) \cdot \theta} \quad (4.5)$$

$$\boldsymbol{T_{g3} = 3n^2(I_1 - I_2) \cdot \varphi \cdot \theta = 0} \quad (4.6)$$

Torque magnético

O torque magnético resulta da interação entre materiais magnéticos dos veículos com o campo magnético terrestre. Ele pode ser expresso por:

$$\boldsymbol{\vec{T}_m = \vec{M} \times \vec{B}} \quad (4.7)$$

No qual $\boldsymbol{\vec{M}}$ é o momento magnético residual do veículo, causado pelo magnetismo permanente e induzido, e $\boldsymbol{\vec{B}}$ é a densidade de fluxo magnético geocêntrico. De acordo com WERTZ e LARSON (1999), pode-se aproximar o módulo do campo magnético terrestre por:

$$\boldsymbol{B = \frac{\mu_E}{r^3}(1 + 3sen^2(\theta))^{1/2}} \quad (4.8)$$

Sendo $\boldsymbol{r = 6878\ km}$ a distância do dipolo do satélite ao centro da Terra; $\boldsymbol{\mu_E = 7,96 \times 10^{15} tesla \cdot m^3}$ a intensidade do vetor de momento magnético ao longo da direção magnética axial; e $\boldsymbol{\theta}$ a latitude magnética medida a partir do equador geomagnético. Com essas informações tem-se que o pior caso – maior módulo do campo magnético –

será quando a latitude seja 90 graus, ou seja, $\boldsymbol{B = 5 \times 10^{-5}}$ Tesla. Considerando o dipolo magnético do satélite igual a $\boldsymbol{M = 1A \cdot m^2}$, obtêm-se o torque magnético.

$$\boldsymbol{T_m = 5 \times 10^{-5}\, N \cdot m} \quad (4.9)$$

Supondo uma órbita na qual o campo magnético só atua no plano que contém a órbita do satélite, pode-se considerar um vetor torque magnético com orientação do tipo:

$$\boldsymbol{\vec{T}_m = T_m sen(nt)\overrightarrow{b_1} + T_m cos(nt)\overrightarrow{b_2}} \quad (4.10)$$

Torque aerodinâmico

Para órbitas terrestres baixas (*Low Earth Orbit* – LEO), situadas entre 350 e 1400 km de altitude, existe a possibilidade de a atmosfera exercer influência significativa em veículos espaciais. Na região da termosfera em particular (80 a 640 km), devido a alta velocidade dos veículos, a existência (mesmo que de baixíssima densidade) de partículas causam perturbações que devem ser consideradas no projeto do subsistema de controle de atitude.

Segundo Carrara (1982), o valor do coeficiente de arrasto $\boldsymbol{C_d}$ depende de vários fatores que podem levar a valores entre 1,2 e 3,8. A rarefação da atmosfera permite que as moléculas possam ser tratadas individualmente (estatisticamente), e não mais como um fluído, dotado de continuidade. Para modelar o torque aerodinâmico (externo), valores que variam nessa faixa 1,2 a 3,8 serão considerados nas simulações.

A expressão do torque pode ser obtida com Wertz e Larson (1999):

$$\boldsymbol{\vec{T}_a = \frac{1}{2}\rho V^2 C_d A \cdot (\vec{u} \times \vec{s}_{cp})} \quad (4.11)$$

Sendo $\boldsymbol{\rho}$ a densidade atmosférica na altitude; $\boldsymbol{V}$ a velocidade do veículo; $\boldsymbol{A}$ a área do veículo perpendicular a $\vec{\boldsymbol{u}}$, vetor unitário na direção da velocidade; $\overrightarrow{\boldsymbol{s_{cp}}}$ o vetor distância entre o centro de massa e o centro de pressão. Como foge do escopo deste trabalho o estudo pormenorizado da aerodinâmica veicular, não será determinado o centro de pressão do veículo, e, portanto, faz-se $\left|\vec{\boldsymbol{u}} \times \vec{\boldsymbol{s}}_{\boldsymbol{cp}}\right| = \boldsymbol{0,2m}$ – ver Pinheiro e Souza (2012).

A densidade atmosférica típica para a altitude é $\boldsymbol{\rho = 3 \times 10^{-11} kg \cdot m^{-3}}$; o coeficiente de arrasto adotado será $\boldsymbol{C_d = 2}$; a velocidade do veículo é $\boldsymbol{V = 7613\ m \cdot s^{-1}}$.Adotando as hipóteses acima e esses valores, pode-se chegar a um valor confiável e constante do torque aerodinâmico para um microssatélites em LEO.

$$\boldsymbol{T_a = \frac{1}{2} \times 3 \times 10^{-11} \times (7613)^2 \times 2 \times 3 \times 0,2 = 1,04 \times 10^{-3} N \cdot m} \qquad (4.12)$$

Para o modelo, supõe-se que o satélite possui um painel solar frontal à velocidade – que é o caso mais crítico, no qual a área normal ao vetor velocidade será maior, gerando o máximo torque. No entanto, é possível concatenar o modelo de órbita do satélite com a área do veículo normal à velocidade.

$$\vec{\boldsymbol{T}}_{\boldsymbol{a}} = \boldsymbol{T_a} \cdot \vec{\boldsymbol{b}}_{\boldsymbol{3}} \qquad (4.13)$$

Tendo delineado as condições ambientais sob as quais o sistema em questão estará submetido, pode-se iniciar a construção do modelo analítico do veículo – iniciando-se pelo aspecto estrutural.

4.2.2. **Geometria e Propriedades Inerciais**

Conforme os requisitos referentes às dimensões, volume e massa elicitados será esboçada uma geometria inicial. A estrutura será composta de dois módulos: o de serviço e de carga útil. É no primeiro que está o SCA e demais subsistemas responsáveis pela operação do satélite. Para o caso de estudo o número de componentes será igual ao do satélite Amazônia-1. Foi feita uma miniaturização proporcional – nas três dimensões – a partir de dados do projeto da PMM para chegar a uma configuração

geométrica inicial. Considerando que os subsistemas contidos no módulo de operação (e de serviço) permaneçam em larga medida de mesma natureza e com posicionamento semelhante, considerar-se-á que a massa total do microssatélite reduza cubicamente – e que, portanto, a massa do sistema como um todo diminua na mesma proporção cúbica – inclusive a de propelente.

Considerando os requisitos instalativos e físicos listados anteriormente, e usando os desenhos das vistas do Amazônia-1 como ponto de partida, foi adotado uma dimensão máxima da base de 400x400 (mm) para o módulo de serviço e carga útil; uma altura de 250 mm para o módulo de serviço; e 300 mm para o módulo de carga útil. Essa configuração inicial respeita as dimensões máximas (600 mm). Aliado a elas, serão consideradas partes físicas de formato semelhante para estabelecer a estrutura física do microssatélite.

Observando as cotas dos desenhos nas Figuras 4.10 e 4.11, a base foi reduzida de 900 mm para 400 mm (~44%); e a altura de 2.480 mm para 600 mm (~25%). Dessa forma o volume dos dois módulos – sem contabilizar equipamentos externos e painéis solares – sofreu uma redução na faixa de 95%. Considerado que o volume dos dois módulos do Amazônia-1 é de 2,0 $\boldsymbol{m^3}$, o volume do microssatélite é de 0,1 $\boldsymbol{m^3}$ (5% do original). Esse valor é o limite volumétrico estabelecido nos requisitos. Como haverá outros componentes (externos), que irão incrementar esse volume, as dimensões da base foram reduzidas para 380x380 $\boldsymbol{mm^2}$ e as alturas do módulo de serviço e carga útil ficaram iguais a 250 $\boldsymbol{mm}$. Percentualmente tem-se uma redução de 57,8% (original x 42,2%) para as dimensões da base; e 79,9% (original x 20,1%) na dimensão de altura. Assim, o novo volume do microssatélite equivale a 3,6% do original – ou seja, 0,072 $\boldsymbol{m^3}$ – satisfazendo os requisitos dimensionais estabelecidos. Trabalhar-se-á com essas dimensões ao considerar-se a geometria dos componentes estruturais para o caso de estudo.

Como não é escopo deste trabalho fazer um projeto esmiuçado do microssatélite, os componentes adotados serão os mesmos e terão a mesma configuração de encaixe.

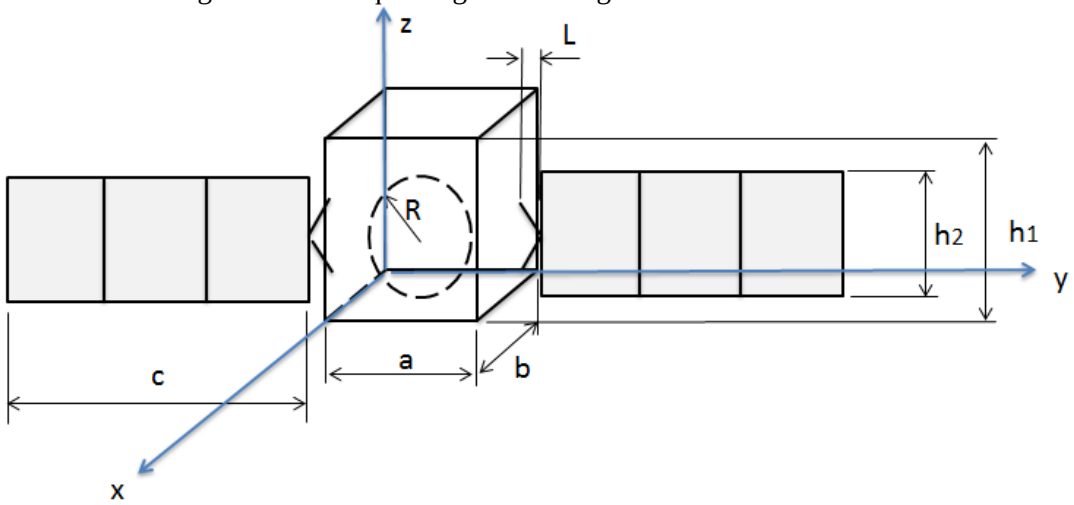
4.2.2.1. Estrutura

Antes de desenvolver o elemento “variável” das propriedades inerciais, será modelado analiticamente o elemento “parâmetro” dessas mesmas.

Baseado nos componentes estruturais do satélite Amazônia-1, visível nas Figuras 4.8, 4.9 e 4.10, foram utilizados cinco volumes elementares para representação das partes: placa plana fina íntegra ou com furo central (faces delimitadoras das fronteiras físicas e painéis solares laterais); casca esférica (tanque); anel (suporte na coifa); e barra delgada (esqueleto da estrutura). Para efeito de simplificação esses volumes foram compactados de forma a restar três formas geométricas elementares: um paralelepípedo oco (corpo do satélite), uma placa fina (painéis solares) e um casco esférico (tanque de propelente). Essa simplificação é útil para as etapas iniciais de desenvolvimento do sistema, que permite realizar o projeto dos subsistemas e ver a influência que um exerce sobre o outro. Etapas posteriores podem envolver o detalhamento de projeto a partir de modelos simplificados preexistentes – como aqueles sendo desenvolvidos neste trabalho.

Na Figura 4.13 tem-se o modelo geométrico conjuntamente com as especificações – desenvolvidas no início do capítulo – que irá servir de guia para a construção do modelo estático de cálculo das propriedades estruturais. Considerar-se-á que as placas solares possuem espessura $\boldsymbol{e_2}$ e as placas que compõem o corpo do satélite espessura $\boldsymbol{e_1}$.

Figura 4.13 – Esquema genérico da geometria de um satélite.



Fonte: Produção do autor.

Definindo o paralelepípedo oco como objeto #1; a placa plana como objeto #2; e o casco esférico como objeto #3, podemos definir os respectivos volumes baseado na Figura 4.13 e nos conceitos.

$$\boldsymbol{V_1 = a \cdot b \cdot h_1 - (a - 2e_1) \cdot (b - 2e_1) \cdot (h - 2e_1)} \quad (4.14)$$

$$\boldsymbol{V_2 = c \cdot h_2 \cdot e_2} \quad (4.15)$$

$$\boldsymbol{V_3 = \frac{4}{3} \cdot \pi \cdot R^3} \quad (4.16)$$

Considerando o sistema de referência da Figura 4.13, pode-se obter o centro de massa de cada um dos objetos relativos a ele (considerando a massa uniformemente distribuída). Temos assim que:

$$\boldsymbol{(CM)_1 = \langle b/2\,, a/2\,, h_1/2 \rangle} \quad (4.17)$$

$$\boldsymbol{(CM)_{2/1} = \langle b/2\,, (a + L + c/2), h_2/2 \rangle} \quad (4.18)$$

$$\boldsymbol{(CM)_{2/2} = \langle b/2\,, (-L - c/2), h_2/2 \rangle} \quad (4.19)$$

$$\boldsymbol{(CM)_3 = \langle b/2\,, a/2\,, R \rangle} \quad (4.20)$$

Obs: $\boldsymbol{(CM)_{2/1}}$ representa o painel solar direito e $\boldsymbol{(CM)_{2/2}}$ o esquerdo.

Conhecendo-se as densidades de cada objeto é possível determinar suas massas ($\boldsymbol{\rho = m/V}$). O centro de massa do conjunto (satélite) é a média ponderada dos centros de massas de cada objeto.

$$\boldsymbol{(CM)_T = \frac{m_1 \cdot (CM)_1 + m_2 \cdot \left[(CM)_{\frac{2}{1}} + (CM)_{\frac{2}{2}} \right] + m_3 \cdot (CM)_3}{m_1 + 2 \cdot m_2 + m_3}} \quad (4.21)$$

$\boldsymbol{(CM)_T}$ se refere ao centro de massa do satélite sem propelente. Posteriormente será adicionado o combustível.

Para calcular o momento de inércia de cada corpo em relação ao centro de massa do sistema (satélite) é preciso usar o conjunto de equações que relaciona o momento de inércia do corpo com o sua posição no conjunto.

$$(\boldsymbol{I_{x'}})_{\boldsymbol{G/i}} = (\boldsymbol{I_x})_{\boldsymbol{i}} + \boldsymbol{m_i} \cdot (\boldsymbol{y_{G/i}^2} + \boldsymbol{z_{G/i}^2}) \quad (4.22)$$

$$(\boldsymbol{I_{y'}})_{\boldsymbol{G/i}} = (\boldsymbol{I_y})_{\boldsymbol{i}} + \boldsymbol{m_i} \cdot (\boldsymbol{x_{G/i}^2} + \boldsymbol{z_{G/i}^2}) \quad (4.23)$$

$$(\boldsymbol{I_{z'}})_{\boldsymbol{G/i}} = (\boldsymbol{I_z})_{\boldsymbol{i}} + \boldsymbol{m_i} \cdot (\boldsymbol{x_{G/i}^2} + \boldsymbol{y_{G/i}^2}) \quad (4.24)$$

Em que:

a) $(\boldsymbol{I_x})_{\boldsymbol{i}}$,$(\boldsymbol{I_y})_{\boldsymbol{i}}$,$(\boldsymbol{I_z})_{\boldsymbol{i}}$ são os momentos de inércia dos objetos $\boldsymbol{i}$;

b) $\boldsymbol{x_{G/i}}$, $\boldsymbol{y_{G/i}}$, $\boldsymbol{z_{G/i}}$ representa a distância entre os centros de massa dos objetos e do conjunto;

c)$\boldsymbol{m_i}$ é a massa de cada objeto;

d) $(\boldsymbol{I_{x'}})_{\boldsymbol{G/i}}$, $(\boldsymbol{I_{y'}})_{\boldsymbol{G/i}}$, $(\boldsymbol{I_{z'}})_{\boldsymbol{G/i}}$ são os momentos de inércia dos objetos $\boldsymbol{i}$ em relação ao conjunto.

A Tabela 4.5 mostra os momentos de inércia de sólidos fundamentais. A partir dela obtém-se o momento de inércia $\langle(\boldsymbol{I_x})_{\boldsymbol{i}}, (\boldsymbol{I_y})_{\boldsymbol{i}}, (\boldsymbol{I_z})_{\boldsymbol{i}}\rangle$ para cada objeto.

Tabela 4.5– Momentos de inércia dos sólidos considerados para o modelo estrutural do satélite.

Elementos principais e suas respectivas propriedades inerciais		**Momentos de inércia**		
		$\boldsymbol{I_{xx}}$	$\boldsymbol{I_{yy}}$	$\boldsymbol{I_{zz}}$
Elementos considerados	Paralelepípedo oco	$\frac{1}{3} \cdot m_1 \cdot (a^2 + h^2)$	$\frac{1}{3} \cdot m_1 \cdot (b^2 + h^2)$	$\frac{1}{3} \cdot m_1 \cdot (a^2 + b^2)$
	Placa plana	$\frac{1}{12} \cdot m_2 \cdot (c^2 + h^2)$	$\frac{1}{12} \cdot m_2 \cdot (h^2)$	$\frac{1}{12} \cdot m_2 \cdot (c^2)$
	Casca esférica	$\frac{3}{2} \cdot m_3 \cdot R^2$	$\frac{3}{2} \cdot m_3 \cdot R^2$	$\frac{3}{2} \cdot m_3 \cdot R^2$

Fonte: Produção do autor.

O cálculo das distâncias é obtido pela diferença,

$$\boldsymbol{x_{G/i} = x_G - x_i} \quad (4.25)$$

$$\boldsymbol{y_{G/i} = y_G - y_i} \quad (4.26)$$

$$\boldsymbol{z_{G/i} = z_G - z_i} \quad (4.27)$$

Sabendo que $\langle \boldsymbol{x_G, y_G, z_G} \rangle$ é o centro de massa do conjunto – já determinado.

Finalmente podem-se calcular os momentos de inércia principais de cada objeto e a seguir somá-los para obter o do conjunto.

4.2.2.2. Propelente

Conforme citado anteriormente, a quantidade de propelente representa aproximadamente 7,0 % da massa do sistema "Amazônia-1" (estrutura e subsistemas), sendo, portanto, considerável a alteração de massa, centro de massa (o posicionamento do tanque está no módulo inferior) e momento de inércia (idem). Sabe-se a quantidade de propelente diminui mais pronunciadamente do que a redução dimensional linear. Isso significa que a porcentagem de propelente relativa a massa do satélite deve diminuir – mas ainda deve ser considerada. Dessa forma é interessante desenvolver uma ferramenta computacional capaz de realizar, a cada tiro de jato (ou conjunto de tiros) em simulação, uma atualização do centro de massa e momentos de inércia do veículo. Tem-se aí o porquê de a matriz de inércia ser considerada uma variável ao invés de parâmetro – ela varia ao longo da vida útil do sistema. Incorporar essa hipótese no conjunto de ferramentas computacionais deve tornar o desenvolvimento menos vulnerável a erros, tornando as análises dos diversos profissionais envolvidos no projeto (estrutura, micropropulsão, controle, sistemas, requisitos, etc) mais confiável. Automatizar o processo de cálculo em ambiente colaborativo deve eliminar a necessidade de um ou mais grupos de profissionais assumirem hipóteses, equacionarem analiticamente (cada qual a seu modo) e fazerem um modelo (cada qual com sua ferramenta) a cada projeto. Tem-se aí uma redução de tempo e recursos dispendidos numa atividade para dedicação a outros tipos de tarefas.

Para modelar a variação do momento de inércia do propelente foramadotadas algumas hipóteses:

a) O posicionamento do satélite – apontamento para o centro da Terra – não influi na variação da direção e sentido do vetor centro de massa do fluido;

b) Os efeitos de oscilação do fluido ("*sloshing*") não são levados em conta.

Isso não significa que, se o ambiente colaborativo esquematizado for construído, ferramentas específicas de cálculo e modelamento não possam ser incorporadas ao mesmo para torná-lo mais fiel à realidade.

Com essas hipóteses pode-se iniciar a elaboração do modelo inercial do fluido. Para esquematizar melhor o que está sendo proposto foi elaborada uma ilustração de um tanque esférico e os fludos que o preenchem: gás pressurizante, Nitrogênio, e propelente, Peróxido de Hidrogênio (Figura 4.14). À medida que são disparados os jatos, a massa (e volume) de propelente se reduz, e o gás Nitrogênio passa a ocupar um volume maior – que antes era ocupado pelo líquido. A pressão no tanque se reduz, assim como o empuxo obtido. Essa redução ocorre porque pressão de entrada (tanque) e o empuxo estão relacionados à vazão mássica através da equação de empuxo (4.4) e da equação de escoamento de um orifício (válvula):

$$\dot{m} = C_d \cdot A_0 \cdot \sqrt{2 \cdot \rho \cdot \Delta P} \tag{4.28}$$

Em que:

C_d: coeficiente de descarga do orifício
A_0: área da seção transversal do orifício
ρ: densidade do fluido
ΔP: diferença de pressão interna-externa (= pressão do tanque)

A vazão mássica se relaciona diretamente ao empuxo através da equação (4.4), sendo responsável pela componente cinética. Ela, por sua vez, é dependente da pressão exercida no combustível, que decresce quadraticamente.

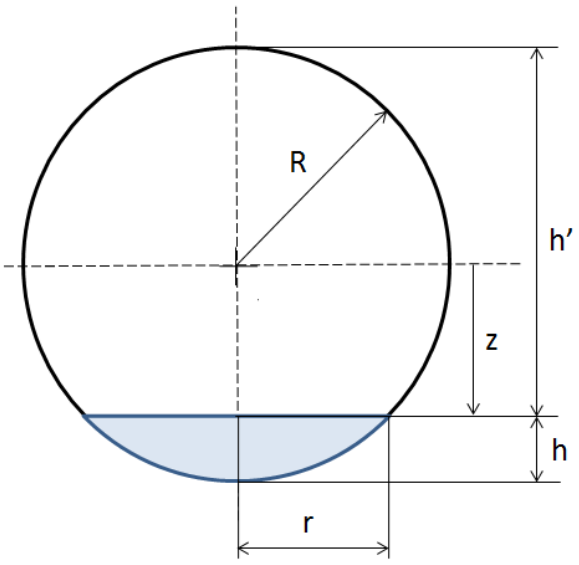
Para obter os modelos do momento de inércia do fluído no tanque iniciar-se-á com um esboço do conjunto.

A Figura 4.14 revela que os volumes ocupados pelo gás e líquido são calotas esféricas. Para saber matematicamente como a variação de volume de propelente se relaciona com a variação do momento de inércia, deve-se inicialmente saber as relações geométricas de um segmento esférico.

A expressão do volume de um segmento esférico (calota) é dada por:

$$\boldsymbol{V_c = \frac{\pi \cdot h'}{6} \cdot (3r^2 + h'^2)} \tag{4.29}$$

Figura 4.14 – Elementos principais de uma calota esférica e um diagrama esquemático bidimensional do tanque esférico.



Fonte: Produção do autor.

Para concatenar de forma mais direta a variação de propelente com a variação do centro de massa e momento de inércia do fluido, a expressão (4.29) deve ter a variável z explicitada. Para fazer essa substituição, basta obter as relações entre z, h, r, h' e R da Figura 4.14. Essas relações permitem a obtenção da função $\boldsymbol{V_c = f(R, z)}$.

$$\boldsymbol{z = h - R} \tag{4.30}$$

$$\boldsymbol{h' = 2R - h = 2R - (z + R) = R - z} \tag{4.31}$$

$$\boldsymbol{r^2 = R^2 - z^2} \tag{4.32}$$

Aplicando o conjunto de equações (4.30) a (4.32) em (4.29) obtém-se o volume de propelente em função da coordenada z e do raio do tanque R.

$$\boldsymbol{V_c = \frac{\pi}{6} \cdot (4R^3 - 6R^2 z + 2z^3)} \tag{4.33}$$

O momento de inércia em relação ao eixo z é obtido através da expressão,

$$\boldsymbol{I_{zz} = \int r^2 dm} \tag{4.34}$$

Em que $\boldsymbol{r}$ é a distância entre o eixo z e a parte infinitesimal do elemento de massa $\boldsymbol{dm}$.

Aplicando a relação (4.32) em (4.34) e sabendo que $\boldsymbol{dm = \rho \cdot dV}$, tem-se que,

$$\boldsymbol{I_{zz} = \int_{V_0}^{V_1} (R^2 - z^2) \cdot \rho \cdot dV} \tag{4.35}$$

Como o interesse consiste em integrar em relação à variável z, podemos realizar a substituição $\boldsymbol{dV = A(z) \cdot dz}$. A área da seção transversal ao eixo z nada mais é do que $\boldsymbol{A(z) = \pi r^2}$. Aplicando (4.32) nessa relação temos a expressão $\boldsymbol{A(z) = \pi \cdot (R^2 - z^2)}$. Logo (4.35) se torna

$$\boldsymbol{I_{zz} = \int_{z=-R}^{z=h} \rho \cdot \pi \cdot (R^2 - z^2)^2 dz} \tag{4.36}$$

Desenvolvendo a expressão acima se obtém o momento de inércia $\boldsymbol{I_{zz}}$ em função do raio do tanque (parâmetro geométrico $\boldsymbol{R}$), da propriedade do propelente (densidade $\boldsymbol{\rho}$) e da quantidade do mesmo – em altura $\boldsymbol{h}$.

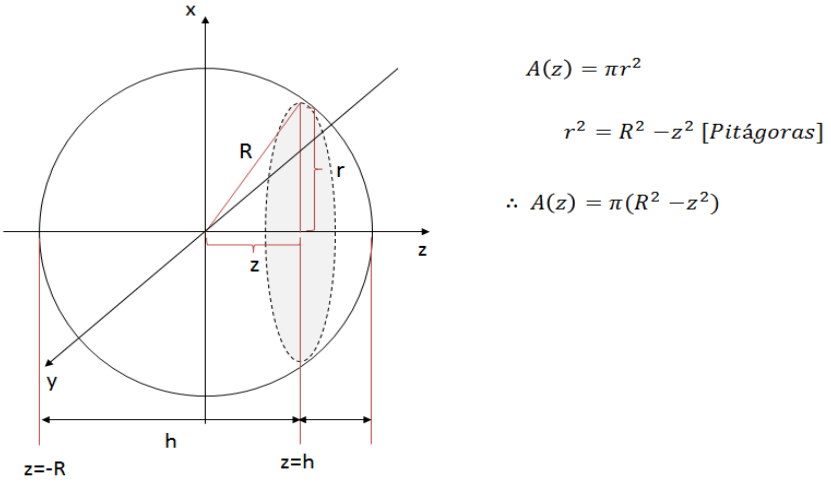
$$\boldsymbol{I_{zz} = \rho \cdot \pi \cdot \left[R^4 \cdot (h + R) - (2/3) \cdot R^2 \cdot (h^3 - R^3) + (1/5) \cdot (h^5 - R^5)\right]} \tag{5.37}$$

A Figura 4.15 ilustra o volume sendo integrado.

Para determinar o centro de massa à medida que a altura $\boldsymbol{h}$ varia, recorre-se à expressão que fornece o centro geométrico de um segmento esférico com uma base plana (caso geral de uma calota):

$$\boldsymbol{z_G = -\frac{3}{4} \cdot \frac{(2R-h)^2}{(3R-h)}} \tag{4.38}$$

Figura 4.15 – Diagrama e equações para determinação do volume da calota.



Fonte: Produção do autor.

Dessa forma sabendo-se (via simulação) a curva de vazão mássica de propelente $\boldsymbol{\dot{m}_p}$ pode-se integrá-la, obtendo a quantidade de combustível no tanque $\boldsymbol{m_p}$, que, por sua vez, fornece o volume do mesmo, $\boldsymbol{V_p}$, via densidade (assumida como constante em função da pressão e temperatura). Assim chega-se a $\boldsymbol{z}$ que está ligada a altura $\boldsymbol{h}$.

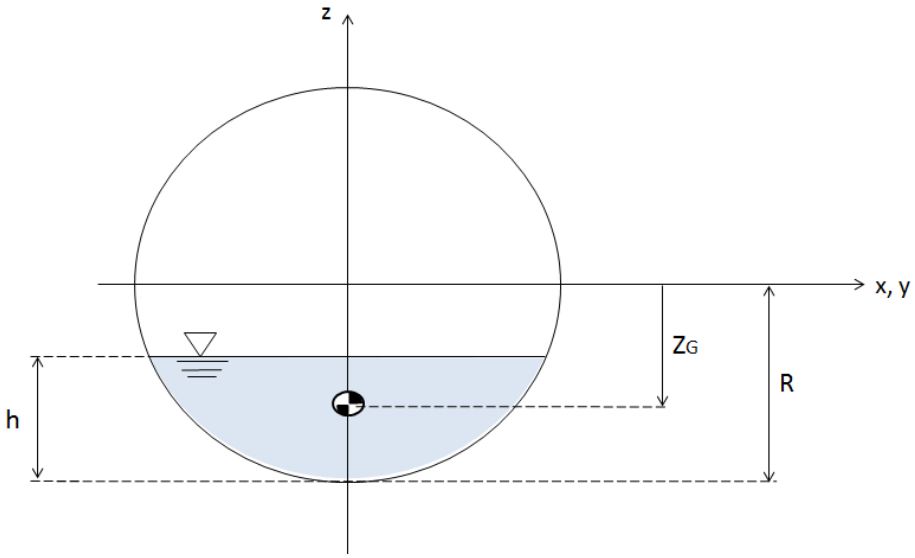
A Figura 4.16 dá uma ideia geométrica do esquema.

Como a distância radial da superfície é igual tanto no eixo x como no eixo y, – e considerando as duas hipóteses – não é necessário determinar o centro de massa em relação a esses eixos x e y. Além disso, a densidade do fluído é uniforme (no espaço) e

constante (no tempo). Dessa forma, o problema do centro geométrico se torna o problema do centro de massa, e determinando um se determina o outro.

A determinação do momento de inércia em relação aos eixos $\boldsymbol{x}$ e $\boldsymbol{y}$ é idêntica pelos motivos expostos acima. Desenvolveu-se o mesmo raciocínio de determinação de $\boldsymbol{I_{zz}}$, usando outro eixo de integração (x ou y).

Figura 4.16 – Diagrama usado para o cálculo do centro de massa do tanque.



Fonte: Produção do autor.

4.2.3. **Dinâmica de Atitude**

As equações de Euler para o movimento de atitude de um satélite são dadas por Yang e Sun (2005):

$$\boldsymbol{I_1} \cdot \dot{\omega}_{\boldsymbol{1}} - (\boldsymbol{I_2} - \boldsymbol{I_3}) \cdot \omega_{\boldsymbol{2}} \cdot \omega_{\boldsymbol{3}} = \boldsymbol{T_{e1}} + \boldsymbol{T_{c1}} \quad (4.39)$$

$$\boldsymbol{I_2} \cdot \dot{\omega}_{\boldsymbol{2}} - (\boldsymbol{I_3} - \boldsymbol{I_1}) \cdot \omega_{\boldsymbol{3}} \cdot \omega_{\boldsymbol{1}} = \boldsymbol{T_{e2}} + \boldsymbol{T_{c2}} \quad (4.40)$$

$$\boldsymbol{I_3} \cdot \dot{\omega}_{\boldsymbol{3}} - (\boldsymbol{I_1} - \boldsymbol{I_2}) \cdot \omega_{\boldsymbol{1}} \cdot \omega_{\boldsymbol{2}} = \boldsymbol{T_{e3}} + \boldsymbol{T_{c3}} \quad (4.41)$$

Sendo:

a) $\boldsymbol{I_1}, \boldsymbol{I_2}, \boldsymbol{I_3}$ os momentos de inércia principais;

b) $\dot{\omega}_{\boldsymbol{1}}, \dot{\omega}_{\boldsymbol{2}}, \dot{\omega}_{\boldsymbol{3}}$ as componentes de velocidade angular nos três eixos do corpo;

c) $\boldsymbol{T_{ei}}$ os torques externos agindo sobre o corpo (aerodinâmico, gravitacional e magnético no caso do presente estudo);
d) $\boldsymbol{T_{ci}}$ os torques de controle (pulsos).

Os torques externos (aerodinâmicos) e de campo (gravitacional e magnético) já foram justificados na elaboração dos requisitos e da missão. Seu equacionamento já foi apresentado. Resta estabelecer o sistema de referência.

Existem diversas maneiras de representar a atitude de um satélite. A orientação geral de um satélite pode ser obtida fazendo sucessivas rotações no espaço em torno de um referencial vertical-local-horizontal-local (VLHL). O número mínimo de rotações para especificar uma rotação, que são os graus de liberdade, é três (TEWARI, 2007). Os três ângulos provenientes dessas rotações sucessivas que representam a atitude são denominados ângulos de Euler, representados por $\boldsymbol{\theta}=[\boldsymbol{\psi}\ \ \boldsymbol{\theta}\ \ \boldsymbol{\Phi}]^{\boldsymbol{T}}$.

Define-se um sistema de referência $\vec{\boldsymbol{B}}$ fixo ao corpo, centrado no centro de massa do satélite e alinhado com os seus eixos principais de inércia, com vetores de base unitários $\{\boldsymbol{b_1}, \boldsymbol{b_2}, \boldsymbol{b_3}\}$. Além dessa referência, considera-se também um segundo sistema $\vec{\boldsymbol{A}}$ (VLHL) com origem no centro de massa do satélite, cujos vetores unitários são $\{\boldsymbol{a_1}, \boldsymbol{a_2}, \boldsymbol{a_3}\}$ em que:

a) $\boldsymbol{a_1}$ se alinha com a direção e sentido da velocidade do satélite;
b) $\boldsymbol{a_2}$é normal ao plano de órbita;
c) $\boldsymbol{a_3}$ completa o triedro (numa órbita cirular, está apontado para fora do centro da Terra).

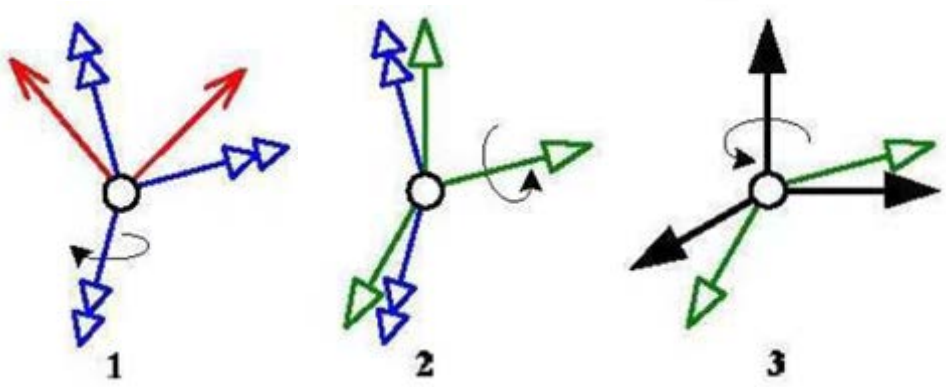
Para colocar o sistema de referência $\vec{\boldsymbol{B}}$ em relação ao $\vec{\boldsymbol{A}}$ em termos dos ângulos de Euler faz-se a transformação de coordenadas (1-2-3). O procedimento é descrito por YANG e SUN (2005):

a) Rotacionar o referencial $\vec{\boldsymbol{A}} = \{\boldsymbol{a_1}, \boldsymbol{a_2}, \boldsymbol{a_3}\}$ em torno do eixo $\boldsymbol{a_1}$ por um ângulo $\boldsymbol{\psi}$, gerando um novo referencial $\{\boldsymbol{a_1'}, \boldsymbol{a_2'}, \boldsymbol{a_3'}\}$;
b) Rotacionar o novo referencial $\{\boldsymbol{a_1'}, \boldsymbol{a_2'}, \boldsymbol{a_3'}\}$ em torno de $\boldsymbol{a_2'}$ por um ângulo $\boldsymbol{\theta}$, gerando outro referencial $\{\boldsymbol{a_1''}, \boldsymbol{a_2''}, \boldsymbol{a_3''}\}$;

c) Rotacionar o referencial $\{\boldsymbol{a_1''}, \boldsymbol{a_2''}, \boldsymbol{a_3''}\}$ em torno de $\boldsymbol{a_3''}$ por um ângulo $\boldsymbol{\Phi}$ resultando no referencial $\{\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}\}$.

Esse processo de rotação em três etapas pode ser visualizado abaixo, na Figura 4.17.

Figura 4.17 – Sequência de rotação dos ângulos de Euler.



Fonte: Manelli Neto (2011).

Essa transformação pode ser descrita pela seguinte matriz de rotação:

$$\boldsymbol{R(\varphi)R(\theta)R(\psi)} = \begin{bmatrix} \boldsymbol{c\theta c\psi} & \boldsymbol{c\theta s\psi} & \boldsymbol{-s\theta} \\ \boldsymbol{s\varphi s\theta c\psi - c\varphi s\psi} & \boldsymbol{s\varphi s\theta s\psi + c\varphi c\psi} & \boldsymbol{s\varphi c\theta} \\ \boldsymbol{c\varphi s\theta c\psi + s\varphi s\psi} & \boldsymbol{c\varphi s\theta s\psi - s\varphi c\psi} & \boldsymbol{c\varphi c\theta} \end{bmatrix}$$

Com as letras $\boldsymbol{s\psi} = \boldsymbol{sen}(\boldsymbol{\psi})$ e $\boldsymbol{c\psi} = \mathbf{cos}(\boldsymbol{\psi})$. Os ângulos $\boldsymbol{\psi}$, $\boldsymbol{\theta}$ e $\boldsymbol{\psi}$ são chamados de *roll*, *pitch* e *yaw*, respectivamente.

Dessa forma, a relação entre o sistema novo e o sistema LVLH, $\vec{\boldsymbol{A}}$, é dada pela equação matricial:

$$\begin{bmatrix} \boldsymbol{b_1} \\ \boldsymbol{b_2} \\ \boldsymbol{b_3} \end{bmatrix} = \begin{bmatrix} \boldsymbol{i} \\ \boldsymbol{j} \\ \boldsymbol{k} \end{bmatrix} = \begin{bmatrix} \boldsymbol{c\theta c\psi} & \boldsymbol{c\theta s\psi} & \boldsymbol{-s\theta} \\ \boldsymbol{s\varphi s\theta c\psi - c\varphi s\psi} & \boldsymbol{s\varphi s\theta s\psi + c\varphi c\psi} & \boldsymbol{s\varphi c\theta} \\ \boldsymbol{c\varphi s\theta c\psi + s\varphi s\psi} & \boldsymbol{c\varphi s\theta s\psi - s\varphi c\psi} & \boldsymbol{c\varphi c\theta} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{a_1} \\ \boldsymbol{a_2} \\ \boldsymbol{a_3} \end{bmatrix} \qquad (4.42)$$

A velocidade angular do referencial fixo ao corpo $\vec{\boldsymbol{B}}$ relativa ao referencial $\vec{\boldsymbol{A}}$ é dado por:

$$\vec{\omega}^{B/A} = \omega_x^{B/A}\boldsymbol{i} + \omega_y^{B/A}\boldsymbol{j} + \omega_z^{B/A}\boldsymbol{k} \tag{4.43}$$

Sendo cada componente de $\vec{\omega}^{B/A}$ as velocidades angulares $\omega_1, \omega_2, \omega_3$ indicadas nas equações 4.39 a 4.41. A relação entre os ângulos de Euler e a velocidade angular de B em relação a A dada por:

$$\begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\end{bmatrix} = \begin{bmatrix}\omega_x^{B/A}\\ \omega_y^{B/A}\\ \omega_z^{B/A}\end{bmatrix} = \begin{bmatrix}1 & 0 & -s\theta\\ 0 & c\varphi & s\varphi c\theta\\ 0 & -s\varphi & c\varphi c\theta\end{bmatrix} \cdot \begin{bmatrix}\dot{\varphi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} - n \cdot \begin{bmatrix}c\theta s\psi\\ s\varphi s\theta s\psi + c\varphi c\psi\\ c\varphi s\theta s\psi - s\varphi c\psi\end{bmatrix} \tag{4.44}$$

Em que n é a frequência orbital do satélite, dada por $n = \sqrt{\mu/R^3}$.

Para pequenos desvios de atitude temos que $\theta \cong \varphi \cong \psi \cong 0$. Dessa forma tm-se que $sen(\theta) \cong \theta$; $sen(\varphi) \cong \varphi$; $sen(\psi) \cong \psi$ e $\cos(\theta) = \cos(\varphi) = cos(\psi) \cong 1$. Substituindo os valores linearizados na equação matricial (5.44) obtemos as equações cinemáticas de atitude linearizadas:

$$\omega_1 = \dot{\varphi} - n\psi \tag{4.45}$$

$$\omega_2 = \dot{\theta} - n \tag{4.46}$$

$$\omega_3 = \dot{\psi} - n\varphi \tag{4.47}$$

Substituindo as velocidades angulares linearizadas nas equações de atitude, e adotando os torques causados pelo ambiente que foram modelados, chega-se ao modelo da atitude do satélite (YANG;SUN, 2005), dado pelas equações diferenciais abaixo, sendo a dinâmica de *roll* acoplada à dinâmica de *yaw*.

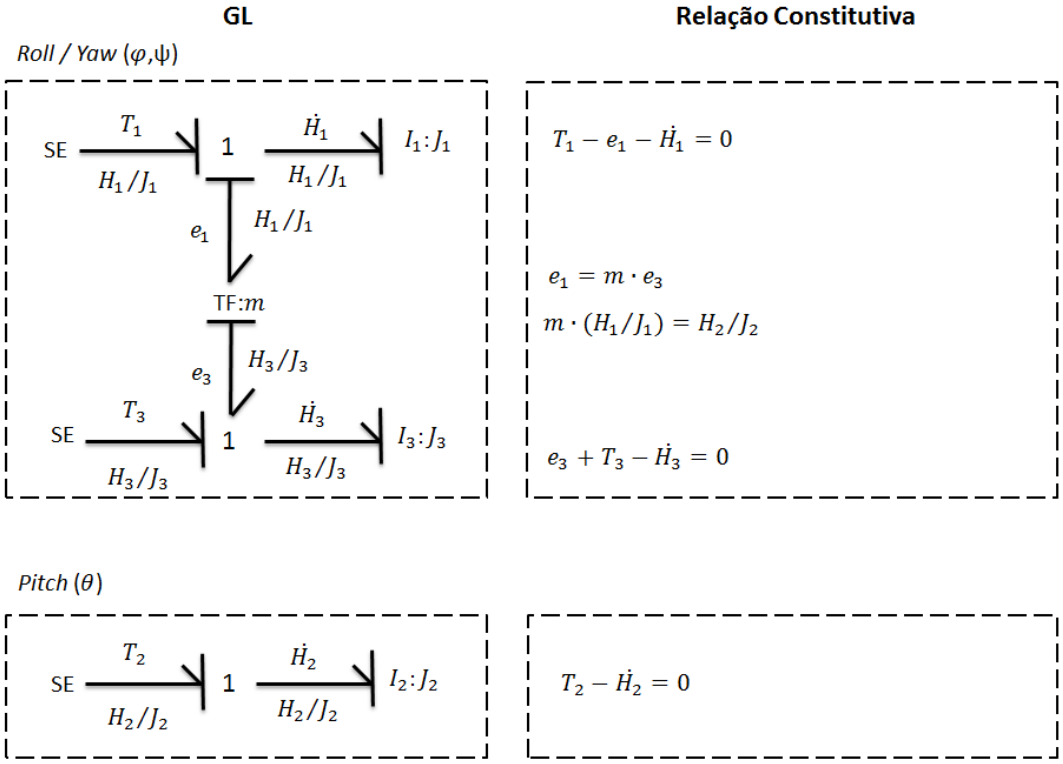
$$I_1\ddot{\varphi} - n(I_1 - I_2 + I_3)\dot{\psi} + 4n^2(I_2 - I_3)\varphi = T_m + T_{c1} \tag{4.48}$$

$$I_2\ddot{\theta} + 3n^2(I_1 - I_3)\theta = T_m + T_{c2} \tag{4.49}$$

$$I_3\ddot{\psi} + n(I_1 - I_2 + I_3)\dot{\varphi} + n^2(I_2 - I_1)\psi = T_a + T_{c3} \tag{4.50}$$

Apesar da M&S de corpos rígidos ser tradicionalmente feita a partir de equações diferenciais e DB, sabe-se que a dinâmica de atitude de um veículo é um fenômeno físico, e que, portanto está sujeito à lei causa-efeito bidirecional. Por esse motivo optou-se por construir um modelo por Gráficos de Ligacão-GL (*Bond Graphs*) – que representa a lógica por trás da ferramenta computacional adotada.

Figura 4.18 – GL e relações constitutivas da dinâmica de atitude do satélite.



Fonte: Produção do autor.

Na Figura 4.18 está representado o acoplamento entre *roll e yaw* (equações. 4.48 e 4.50) através de um elemento transformador TF. As fontes externas (SE) representam o somatório dos torques (controle e externos) atuando sobre o componente. Esse tipo de representação é muito mais intuitiva, sendo de fácil elaboração – além de cextensiva a outros setores relacionados ao desenvolvimento do sistema. No Capítulo 6 será vista a implementação dessa abordagem em ferramenta computacional.

Após estabelecer essas diretrizes pode-se modelar analiticamente a lei de controle (pelo paradigma de fluxos informacionais) e o atuador responsável por materializar essa lei, o micropropulsor e elementos que o alimentam (pelo paradigma de fluxos físicos). Entraremos agora nas leis de controle.

4.2.4. **Controle LQR**

O regulador linear quadrático deve aplicar uma entrada de controle em uma planta (satélite) em que os estados (variáveis intermediárias) são diferentes de zero, e levá-los a um valor próximo ou igual a zero – num intervalo de tempo limitado. Sabe-se que a planta está sujeita a perturbações não desejadas (torques de campo e externos). Para um controlador LQR assume-se que todos os estados estão disponíveis para realimentação.

Supondo um modelo no espaço de estados temos:

$$\dot{\boldsymbol{x}}(\boldsymbol{t}) = \boldsymbol{A}\boldsymbol{x}(\boldsymbol{t}) + \boldsymbol{B}\boldsymbol{u}(\boldsymbol{t}) \tag{4.51}$$

$$\boldsymbol{y}(\boldsymbol{t}) = \boldsymbol{C}\boldsymbol{x}(\boldsymbol{t}) + \boldsymbol{D}\boldsymbol{u}(\boldsymbol{t}) \tag{4.52}$$

Com a matriz $\boldsymbol{A_{6x6}}$, relacionada aos parâmetros da planta; $\boldsymbol{B_{6x6}}$ relacionada aos atuadores de controle da mesma; $\boldsymbol{C_{3x6}}$ relacionada aos sensores; e $\boldsymbol{D_{3x1}}$ usualmente é considerada nula.

O modelo da planta (dinâmica de atitude do satélite) se constrói a partir do conjunto de equações diferenciais 4.48 a 4.50. Definindo o vetor de estados como os deslocamentos e velocidades angulares, ou seja, $\{\boldsymbol{x}\} = \{\boldsymbol{\varphi}, \boldsymbol{\theta}, \boldsymbol{\psi}, \dot{\boldsymbol{\varphi}}, \dot{\boldsymbol{\theta}}, \dot{\boldsymbol{\psi}}\} = \{\boldsymbol{x_1}, \boldsymbol{x_1}, \boldsymbol{x_3}, \boldsymbol{x_4}, \boldsymbol{x_5}, \boldsymbol{x_6}\}$, temos a formulação matricial:

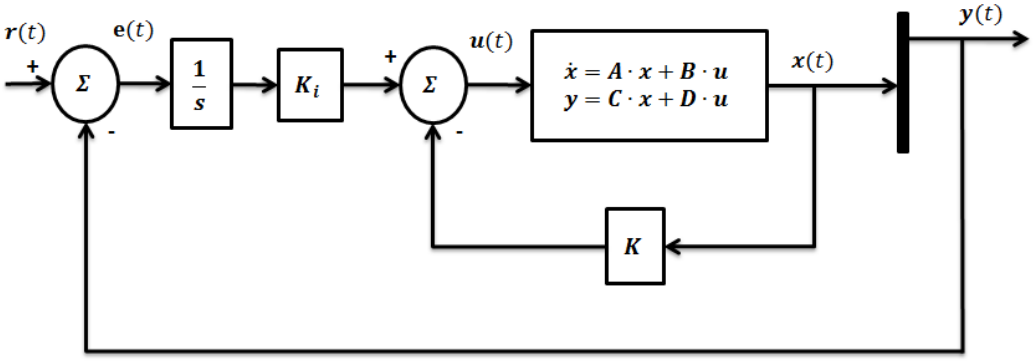
$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \\ \dot{x_4} \\ \dot{x_5} \\ \dot{x_6} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \frac{-4n^2(I_2 - I_3)}{I_1} & 0 & 0 & 0 & 0 & \frac{n(I_1 - I_2 + I_3)}{I_3} \\ 0 & \frac{-3n^2(I_1 - I_2)}{I_2} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{-n^2(I_2 - I_1)}{I_3} & \frac{-n(I_1 - I_2 + I_3)}{I_3} & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} +$$

$$\begin{bmatrix} 0 & 0 & 00 & 0 & 0 \\ 0 & 0 & 00 & 0 & 0 \\ 0 & 0 & 00 & 0 & 0 \\ \frac{d_1}{I_1} & \frac{d_1}{I_1} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{d_2}{I_2} & \frac{d_2}{I_2} & 0 & 0 \\ 0 & 0 & 0 & \frac{d_3}{I_3} & \frac{d_3}{I_3} \end{bmatrix} \cdot \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 00 & 0 & 0 \\ 0 & 1 & 00 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}$$

O diagrama do um sistema de controle proporcional-integrativo (PI) com retroalimentação de estados pode ser visualizado na Figura 4.19.

Figura 4.19 – DB da lógica de controle LQR implementada.



Fonte: Produção do autor.

A Teoria de Controle afirma que se o par $(\boldsymbol{A}, \boldsymbol{B})$ for controlável e o par $(\boldsymbol{A}, \boldsymbol{C})$ for observável, entãoexistirá um controlador LQR dado pela expressão

$$u(t) = K_{LQR} \cdot x(t) \tag{4.53}$$

Que minimiza o funcional J dado por

$$J = \int_0^{\infty} [x^T(t) Q x(t) + u^T(t) R u(t)] dt \tag{4.54}$$

J representa a soma da energia de todos os estados e controles, que deve ser minimizada em função do controle u. A matriz de peso Q é simétrica semi-definida positiva que penaliza os estados; e R é a matriz de peso simétrica definida-positiva que penaliza os sinais de controle. Se Q cresce à medida que R permanece constante, o tempo de estabilização é reduzido e os estados tendem a zero a uma taxa maior. Se $\mathrm{diag}(R) \gg diag(Q)$, a energia de controle é penalizada com maior intensidade.

O ganho é calculado através da seguinte expressão:$K_{LQR} = R^{-1}B^T P$ (4.55)

Em que P é matriz de solução única, simétrica, semi-definida positiva, obtida através da solução da equação algébrica de Riccatti.

$$PA + A^T PT + Q - PBR^{-1}B^T PT = 0 \quad (4.56)$$

Substituindo (4.55) em (4.51) temos o sistema em malha fechada.

$$\dot{x}(t) = (A - BK_{LQR}) \cdot x(t) \quad (4.57)$$

O controlador LQR tem a seguinte propriedade: se o sistema for controlável e observável, então em malha fechada ele será assintoticamente estável. Isso significa que todos os autovalores de $A - BK_{LQR}$ são negativos e reais – isso será visto em detalhes no modelo computacional do computador.

Devido ao escopo, sensores não foram considerados neste trabalho, i.e., C=I.

Em termos de implementação, o primeiro passo para projetar um controlador LQR consiste em determinar seus ganhos – matriz $\boldsymbol{K_{LQR}}$. Existem comandos precisos que realizam esses cálculos em aplicativos voltados a essa finalidade. As entradas para esse comando são as matrizes de espaço de estados $(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D})$. As matrizes $\boldsymbol{A}$ e $\boldsymbol{B}$ são obtidas através das equações (4.48) a (4.50), bastando reordená-las na forma de espaço

de estados para extração dos coeficientes. A matriz $\boldsymbol{C}$ fornece o número de estados observáveis e a matriz $\boldsymbol{D}$ usualmente é nula.

As matrizes aumentadas $\widehat{\boldsymbol{A}}$ e $\widehat{\boldsymbol{B}}$ são dadas por:

$$\hat{A}_{9x9} = \begin{bmatrix} A_{6x6} & 0_{6x3} \\ -C_{3x6} & 0_{3x3} \end{bmatrix}; \quad \hat{B}_{9x6} = \begin{bmatrix} B_{6x6} \\ 0_{3x6} \end{bmatrix} \tag{4.58}$$

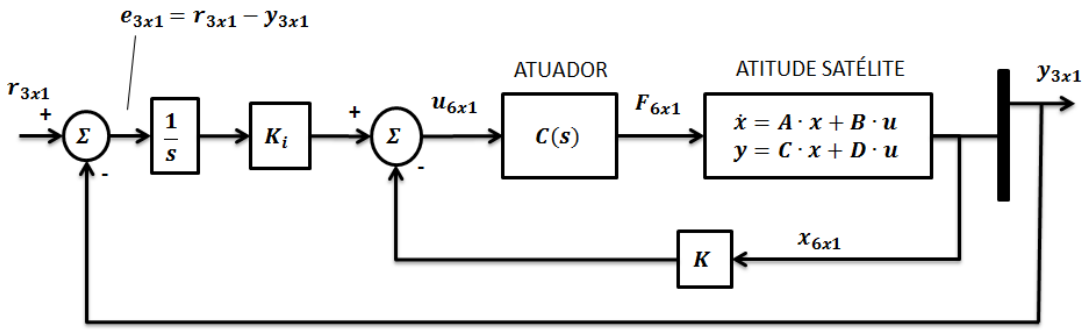
Os pesos $\boldsymbol{Q}$ e $\boldsymbol{R}$ são baseados em Pinheiro e Souza (2012). O ganho aumentado é dado por:

$$\hat{K}_{9x6} = [K_{6x6} \quad -K_{i_{6x3}}] \tag{4.59}$$

Com os coeficientes das matrizes de estado e o cálculo dos ganhos é possível implementar o Diagrama de Blocos-DB (*Block Diagram*) da Figura 4.19 num ambiente computacional específico.

A título de enriquecer o modelo de controle LQR, foi adicionado o conjunto de atuadores (conjunto de 6 propulsores) à malha. Os estados do conjunto de atuadores não foram considerados realimentados. Logo não há necessidade de se adicionar nas matrizes de ganhos (K e K_i) termos referentes ao atuador. Isso é implementável se o subsistema for controlável e não afetar a estabilidade da planta como um todo. A configuração nova passa a ser representada considerando um subsistema atuador com seis modelos em espaço de estados (Figura 4.22).

Figura 4.20 – DB da lógica de controle LQR implementada com atuador.



Fonte: Produção do autor.

Cada atuador (propulsor) foi modelado separadamente em espaço de estados. O número de variáveis de estado supostas foi três. Os domínios envolvidos na modelagem foram: eletromecânico (válvula), hidráulico (orifício) e termodinâmico (empuxo).

A válvula de controle é modelada como um sistema de 2ª ordem com uma força externa (entrada u) que é a força eletromagnética causada pelo solenoide.

$$m_c\ddot{x} + b\dot{x} + kx = F_m \tag{4.60}$$

A equação de escoamento de um fluído (incompressível) num orifício é dada por uma função da diferença de pressão (Δp), área de passagem do fluído (A), densidade do mesmo (ρ) e coeficiente de descarga (C_d), conforme a equação (5.28). Para obter os parâmetros de amortecimento viscoso e rigidez da mola foram usadas as expressões da teoria de vibrações:

$$b = 2 \cdot \zeta \cdot \omega_n \cdot m_c \tag{4.61}$$

$$k = (\omega_n{}^2) \cdot m_c \tag{4.62}$$

O valor da massa do carretel foi baseado em valor de trabalho anterior (OLIVA, 2012) considerando um fator de redução cúbico – devido à redução de dimensões do veículo como um todo. O coeficiente de amortecimento e a frequência natural da válvula ($\zeta\ e\ \omega_n$, respectivamente) foram inicialmente escolhidos arbitrariamente, baseados nos requisitos de projeto, sendo ajustados após algumas corridas de simulações.

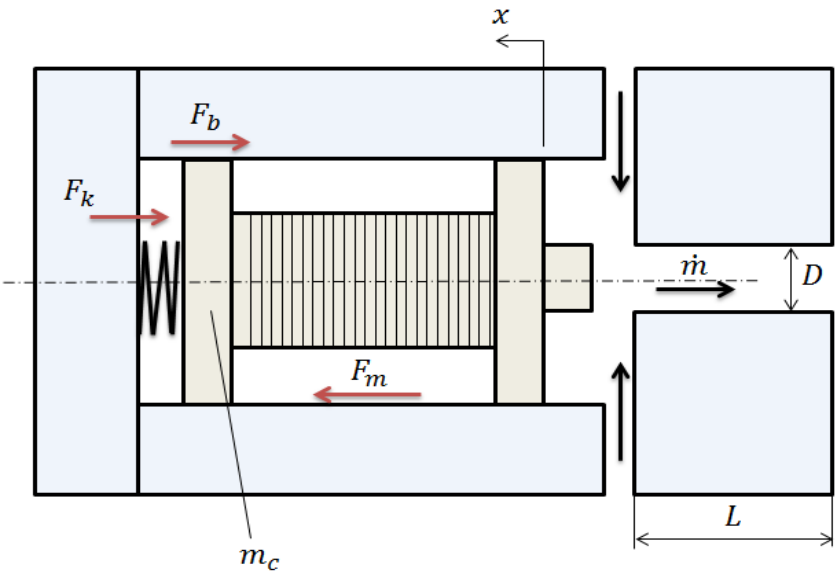
Se considerarmos uma válvula solenoide que possui uma mola que exerce uma força F_k a um carretel de massa m_c, temos um deslocamento x, ilustrado na Figura 4.21. Esse deslocamento se relaciona com a área pela qual o fluído irá escoar, resultando na vazão mássica de propelente.

$$A_0 = A(x) = \pi D x \tag{4.63}$$

Substituindo (4.63) em (4.28) temos uma expressão que relaciona vazão mássica com o deslocamento do carretel.

$$\dot{m} = C_d \cdot \pi \cdot D \cdot (\sqrt{2 \cdot \rho \cdot \Delta P}) \cdot x \tag{4.64}$$

Figura 4.21 – Diagrama esquemático de uma válvula solenoide (sistema de 2ª ordem com força externa).



Fonte: Produção do autor.

Como o modelamento antecede a entrada na câmara catalítica – onde ocorre a reação termoquímica de decomposição do peróxido de hidrogênio – pode-se considerar a densidade constante. A diferença de pressão é equivalente à pressão do tanque, e o coeficiente de descarga será considerado constante, baseado em trabalho anterior do autor (OLIVA, 2012). Dessa forma pode-se considerar que a vazão mássica é diretamente proporcional ao deslocamento do carretel, ou seja, $\dot{m} = K_v \cdot x$, em que:

$$K_v = C_d \pi D \sqrt{2\rho\Delta p} \tag{4.65}$$

A representação termodinâmica do empuxo foi baseada na equação (4.4), com algumas simplificações. A primeira delas foi desconsiderar o empuxo gerado pela diferença de pressão. Esse tipo de hipótese é razoável, uma vez que trabalho anterior do autor demonstrou que essa parcela contribui pouco com a geração de empuxo (OLIVA,

2012). O fator de correção do bocal também foi desconsiderado devido à geometria diversa do bocal, sendo substituído por outro fator que considera as limitações geométricas de uma tubeira de seção retangular / quadrada (HITT et al., 2001). Para permitir a construção do modelo em espaço de estados, a velocidade de ejeção dos gases (V_e) foi considerada constante.

As variáveis de estado consideradas foram: (1) deslocamento do carretel; (2) sua velocidade e; (3) a massa de propelente passando pelo orifício. A força eletromagnética foi considerada como a entrada. Dessa forma é possível obter o modelo linearizado no espaço de estados de cada atuador.

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -k/m_c & -b/m_c & 0 \\ K_v & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m_c \\ 0 \end{bmatrix} \cdot u \tag{4.66}$$

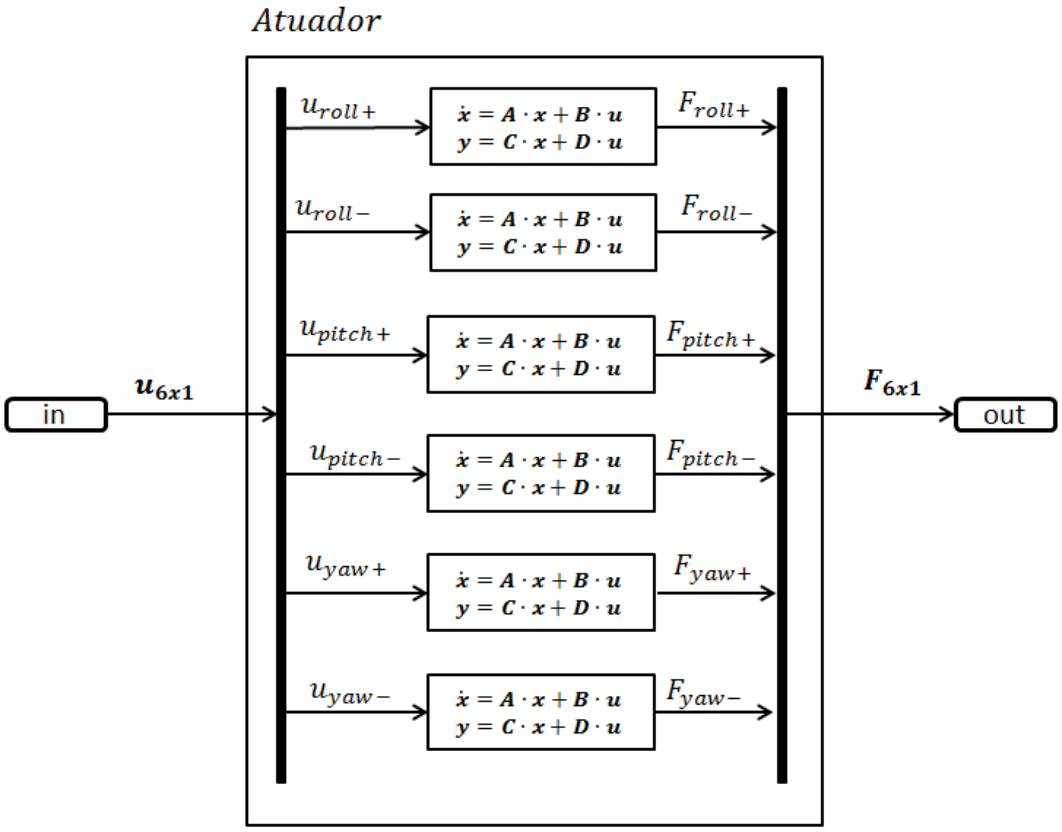
$$y = [K_v \cdot V_e \quad 0 \quad 0] \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \tag{4.67}$$

Cada submodelo está inserido no bloco “atuador”, visto na Figura 4.22.

Com essa representação parcial dos atuadores é possível obter uma entrada $u(t)$ mais fiel a partir do modelo do controlador LQR, inserindo-a como entrada dos modelos do subsistema propulsivo (alimentação / FCV e propulsor), que fornecerão um empuxo, que, por sua vez, irá ser entrada do modelo de dinâmica de atitude.

Dessa forma construiu-se uma representação por fluxos informacionais que pode ser simplificada, com integração parcial dos modelos por fluxo de potência (propulsão), na qual as saídas dos mesmos não podem ser usadas na dinâmica de atitude por GL; ou integrada, na qual estes estarão inseridos na cadeia de simulação e seus resultados (empuxo) poderão ser conectados ao modelo de atitude.

Figura 4.22 – Detalhamento do subsistema “atuador”.



Fonte: Produção do autor.

4.2.5. **Subsistema Propulsivo**

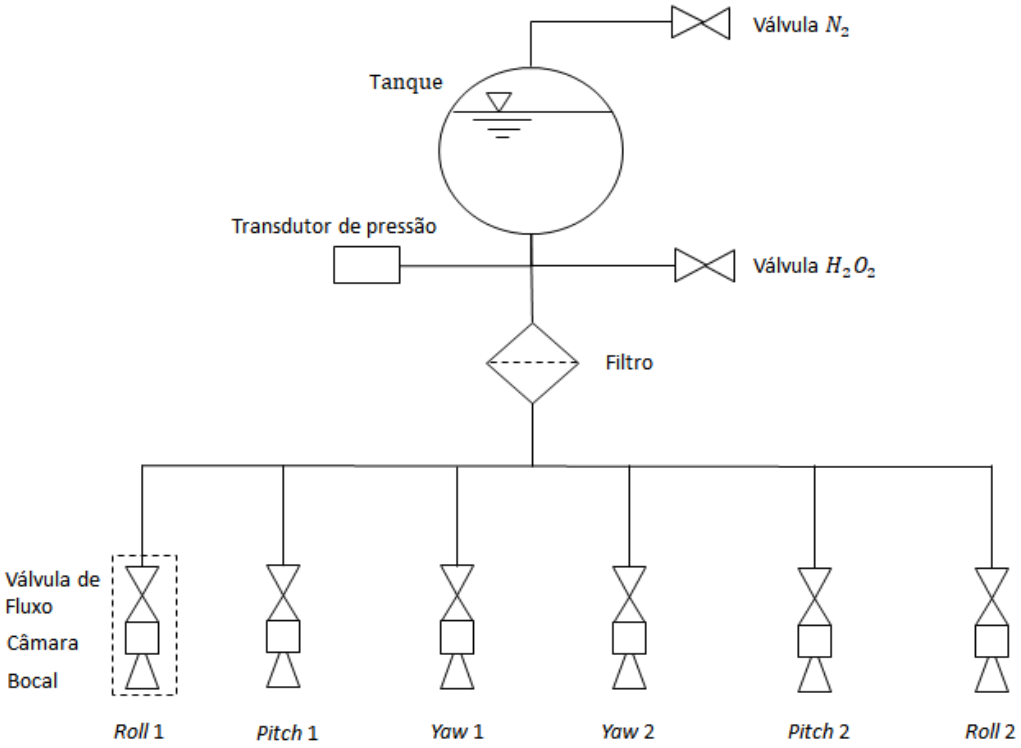
O subsistema propulsivo pode ser subdividido em duas partes: (a) a hidráulica (ou hidrodinâmica), composta de tanque, tubulações, filtro, e válvulas de pressão, que alimentam o propulsor (b), que é subdividido em válvula de controle, câmara catalítica e bocal. A primeira parte (a) pode ser representada por fenômenos hidráulicos – não há reações químicas, transferência de calor nem escoamento compressível. A segunda parte (b) deve ter fenômenos termoquímicos (câmara) e termodinâmicos (bocal) representados em seu modelo.

O subsistema propulsivo que será o atuador do SCA é químico monopropelente, cujo propelente será peróxido de hidrogênio ($\boldsymbol{H_2O_2}$), que é a escolha mais adequada dentre

os tipos de sistemas existentes para micropropulsão – conforme detalhado em seções anteriores.

O controle de atitude deve se dar nos três eixos. Para isso, é necessário que os propulsores sejam posicionados adequadamente em pares – cada empuxo de correção de atitude deve ter uma aceleração num sentido e consequente desaceleração equivalente, para estabilizar o posicionamento no veículo. Parte-se assim da necessidade de seis propulsores. Um diagrama esquemático, na Figura 4.23, revela os componentes do subsistema propulsivo do microssatélite. As válvulas do gás pressurizante e do propelente são acionadas no modo pré-lançamento; o transdutor de pressão mede a pressão a qual o propelente está submetido; o filtro tem a função de impedir a passagem de partículas para o propulsor, que poderiam diminuir sua eficiência e gerar desgaste precoce. Os componentes citados, além da tubulação, pertencem ao domínio hidráulico, caracterizado por escoamento incompressível sem trocas de calor e geração de energia.

Figura 4.23 – Diagrama esquemático do subsistema propulsivo.



Fonte: Produção do autor.

Os propulsores podem ser subdivididos em duas partes: a válvula de controle de fluxo (*Flow Control Valve* – FCV) e o conjunto câmara-bocal. A primeira recebe o sinal de controle, operando a abertura de um orifício pelo qual o fluido deve passar. No caso da PMM, a válvula é solenoide: o sinal gera uma tensão, que faz circular uma corrente através de uma bobina de fio condutor enrolada num carretel com núcleo de material ferromagnético, gerando assim um campo magnético que, por consequência, causa uma força, que deve ser oposta àquela que mantém o orifício fechado – força hidráulica e potencial elástica (mola). Essa etapa ocorre no domínio eletromagnético. A seguir, há a dinâmica de abertura, na qual o carretel se move devido à superação das forças hidráulicas e elásticas pela força magnética, gerando o deslocamento do carretel e consequente passagem de fluido por um orifício. Essa etapa envolve o domínio eletromecânico e hidrodinâmico.

O fluido em estado líquido, ao atingir a câmara catalítica (pré-aquecida), reage, gerando gás oxigênio, água e calor, que fluem através do bocal. Aí ocorrem fenômenos termoquímicos e termodinâmicos (escoamento compressível). Dessa forma, percebe-se que o subsistema propulsivo, em especial seu componente mais complexo (micro propulsor), possui uma multiplicidade de fenômenos físicos e químicos ocorrendo num intervalo de tempo relativamente curto. Convém representá-los através de uma linguagem adequada é útil do ponto de vista de compreensão, reuso e simulação – uma vez que modelos apoiados no paradigma de fluxos físicos (matéria / energia) consideram os fenômenos de causa-efeito, que podem ser muito importantes na validação de sistemas desse tipo, conforme já demonstrado alhures (OLIVA, 2012).

Os GLs partem da abordagem intuitiva de que um sistema dinâmico é composto de subsistemas, componentes ou elementos básicos que trocam fluxos físicos entre si. Essa troca é associada com o intercâmbio de quantidades (ex: massa, momento, energia). Conforme visto anteriormente, nos fluxos de informação não existem restrições a serem impostas. Para fluxos físicos há necessidade de impor leis de conservação do mundo físico. Logo, subsistemas que recebem energia de outro devem: (a) *transferir* essa energia integralmente a outro subsistema, ou; (b) *distribuir* essa energia a outros subsistemas, ou; (c) *transformar* essa energia, ou; (d) *armazenar* essa energia.

A modelagem em GL inicia-se com um processo esquemático de componentes, semelhante ao da Figura 4.23. A partir dele, pode-se iniciar a construção do modelo. Baseado nos conceitos sobre GL apresentados no Capítulo 2, foram esquematizados os modelos simbólicos por fluxos de potência, que constituem a lógica sob a qual a ferramenta de simulação da parte física opera. Foi modelado o sistema de alimentação (tanque, filtro, tubulação e válvula de controle de propulsor), pertencente ao domínio hidráulico; e o propulsor (câmara catalítica e bocal convergente-divergente), pertencente ao domínio termoquímico e termodinâmico.

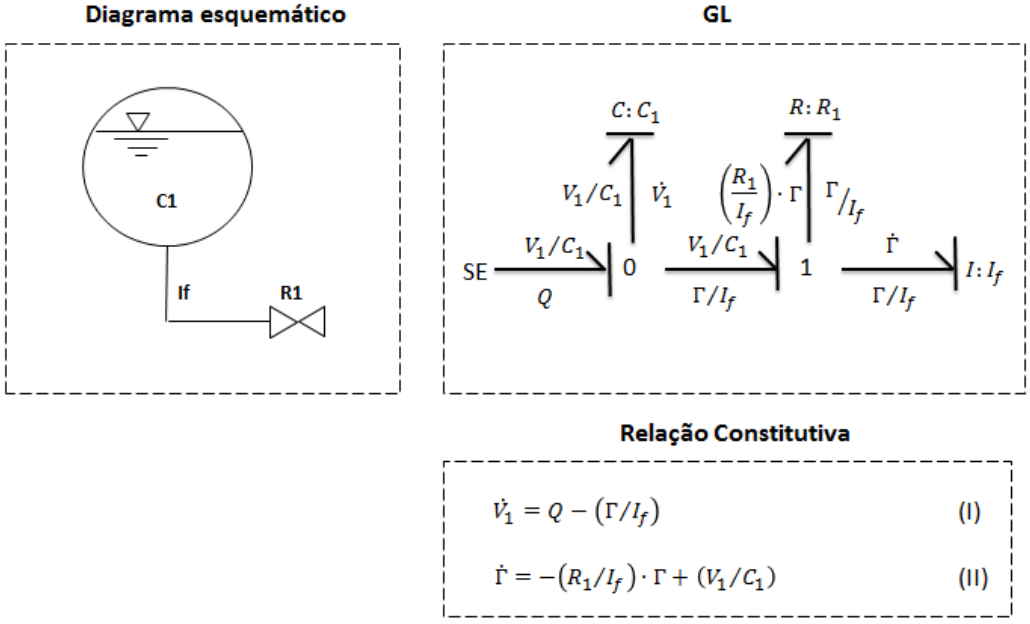
O modelo do tanque, tubulações, filtro, válvulas e propulsor do subsistema propulsivo são, em vários aspectos, próximos a um trabalho anterior do autor (OLIVA, 2012), que serviu de elemento delineador para as construções simbólicas. Logo, o modelo analítico é semelhante. No entanto, aqui será feita uma representação gráfica de partes do subsistema por GL. Iniciou-se com o tanque e tubulação (Figura 4.24).

O primeiro passo consiste em identificar o componente armazenador de energia: o tanque, que é igualmente a fonte de esforço (pressão) do sistema.

A junção “0” representa o tanque com capacitância $\boldsymbol{C_1}$ Nele, a variável esforço é idêntica nas saídas e entradas e equivale à pressão do gás $\boldsymbol{N_2}$, que é o volume do elemento capacitivo (tanque) dividido pela sua capacitância, ou seja, $\boldsymbol{V_1/C_1}$.

Conectado ao tanque, temos a tubulação (junção “1”), elemento com resistência e inertância. Apesar de a primeira ser desprezível em relação à energia transportada, para efeitos de representação o fenômeno foi considerado. A tubulação possui a variável fluxo em comum. Esta é dada pela razão entre o momento hidráulico e a inertância, $\boldsymbol{\Gamma/I_f}$.

Figura 4.24 – GL e relações constitutivas do domínio hidráulico do subsistema propulsivo.



Fonte: Produção do autor.

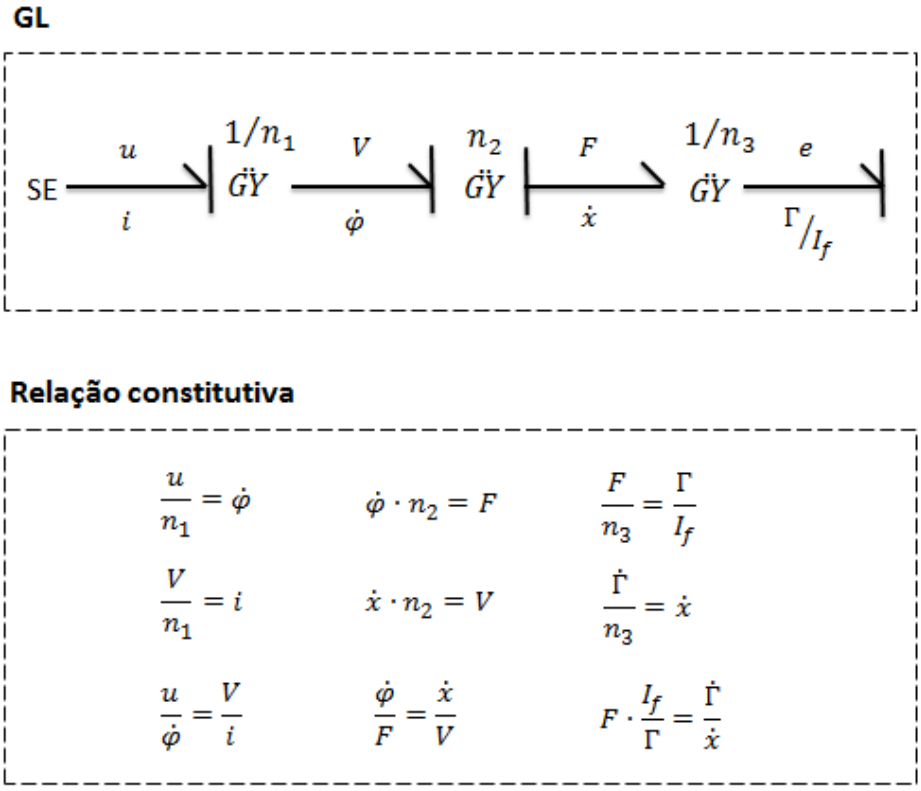
Sabe-se ainda o fluxo que sai do tanque ($\boldsymbol{Q}$). A partir do tetraedro de estados e das simbologias – apresentados no Capítulo 2 – é possível determinar as variáveis restantes.

Denominando $\boldsymbol{\dot{V}_1}$a (variável fluxo) referente à potência armazenada e aplicando a lei referente à junção “0”, chega-se à relação constitutiva (I).

Denominando $\boldsymbol{\dot{\Gamma}}$ a (variável esforço) referente à potência cinética e aplicando a relação linear $\boldsymbol{e = R \cdot f}$ para a dissipação de potência, chega-se a relação constitutiva (II).

O próximo movimento consiste em representar os fluxos de potência da válvula de controle do propulsor. Essa válvula é um elemento transformador do tipo girador (GY), pois converte uma variável de esforço de entrada (tensão elétrica) numa variável fluxo de saída (força magnética), e uma variável fluxo de entrada (corrente elétrica) em uma variável esforço de saída (velocidade de abertura). A Figura 4.25 fornece o GL e as relações constitutivas correspondentes.

Figura 4.25 – GL e Relações constitutivas da válvula do propulsor.



Fonte: Produção do autor.

Neste caso, percebe-se que o transformador, ao contrário da dinâmica de atitude, é um girador (GY), pois transforma uma variável de esforço de entrada numa variável fluxo de saída – e uma variável fluxo de entrada numa variável esforço de saída.

Tomando a 1ª coluna da relação constitutiva para exemplificar o processo, vê-se que a tensão elétrica $\boldsymbol{u}$, um esforço de entrada, se torna fluxo magnético $\boldsymbol{\dot{\varphi}}$ (não confundir com velocidade de *roll*), um fluxo de saída.

O propulsor se divide em câmara catalítica e bocal. As bibliotecas utilizadas para representar os fenômenos físicos foram: pneumática e a térmica. Por fugir do escopo deste trabalho, decidiu-se não elaborar a notação em GL da parte termodinâmica (propulsor). No entanto, seu modelo computacional foi elaborado – baseado nos componentes da biblioteca pneumática e térmica, e em trabalho muito detalhado sobre o tema desenvolvido alhures (OLIVA, 2012).

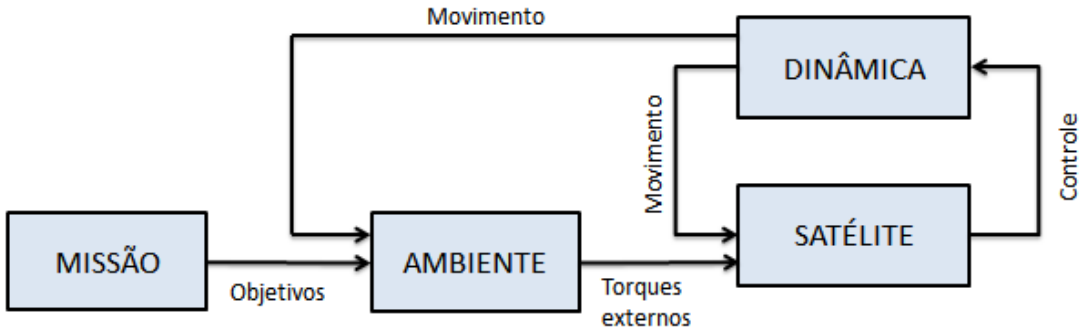
4.3. **Modelos Computacionais**

As ferramentas computacionais podem ser ou não modelos. O que diferencia as primeiras destas últimas é sua função: enquanto modelos, representam (e executam) algo a ser materializado ou já existente; já as ferramentas executam operações que podem ser não-relacionadas a representações. A natureza das ferramentas foi vista anteriormente, no Capítulo 3. Nesse tópico, será feita uma breve descrição da natureza dos modelos construídos para o estudo.

O modelo computacional é a implementação dos modelos simbólicos e métodos, através de ferramentas adequadas, além de comunicar-se com os requisitos – que estabelecem a missão que, por sua vez, define o ambiente espacial. A partir das definições iniciais da missão, serão selecionados modelos adequados à mesma, fazendo-se uso de um método de busca. A busca e escolha são feitas por humanos – mas pode vir a ser construído um mecanismo de busca, que localiza determinado modelo numa biblioteca. Essa busca pode se dar não apenas com a inserção de palavras-chave, mas também baseada em domínios, ferramentas e aplicações dos modelos, além do tipo de relação entre os mesmos.

A Figura 4.26 esquematiza de forma geral o processo, destacando as relações.

Figura 4.26 – Modelo computacional relacionando missão, ambiente, satélite e sua dinâmica através de variáveis que (in)fluem no ambiente colaborativo.



Fonte: Adaptado de Schaus et al. (2010).

Independente do tipo de sistema, existem sempre quatro elementos essenciais: (a) a missão, que a partir dos interessados e dos métodos da Engenharia de Requisitos, estabelece os requisitos de sistema e define o ambiente em que o sistema irá operar; (b) o ambiente de operação, com seus gradientes de temperatura, pressão, forças, torques, riscos, impactos, entre outros; (c) o sistema em si, isto é, o satélite, com seus subsistemas de controle de atitude e órbita, controle térmico, de manuseio de dados, propulsivo, de telemetria, sua estrutura (entre outros); e (d) a dinâmica à qual o sistema está sujeito – no caso do estudo os movimentos de rotação causados pela ação de controle e torques externos.

Viu-se que – para construir as cadeias de processo – deve ser feita uma fusão do ambiente e missão, englobando-os numa ferramenta única que deve repassar as informações essenciais ao projeto (ex: órbita, frequência orbital, dimensões, materiais, etc.).

A seguir, serão apresentadas sucintamente as ferramentas computacionais com suas principais características.

4.3.1. **Missão e Ambiente**

A missão define os modelos a serem usados para o desenvolvimento do sistema. A partir dela se delimita qual será a órbita e o tipo de sistema – e, consequentemente, os fenômenos ambientais a serem considerados. Além disso, nela estão contidos os requisitos de sistema de forma geral, que adicionam objetivos e restrições a cada subsistema.

No processo de desenvolvimento, a ferramenta condutora da cadeia de processos deve fornecer as informações necessárias para que todas as outras ferramentas estejam alinhadas e operem de modo harmônico. Isso inclui: tempo de simulação, parâmetros globais, forma de apresentação de dados, entre outros. O arquivo deve preferencialmente ser visível a todos os nós (pessoas, grupos, institutos) da cadeia de processos colaborativos. É a partir da leitura dele (arquivo) que os responsáveis pelas diversas ferramentas (custos, modelos físicos, informacionais, de projeto, cálculos de

parâmetros, análise, otimização, etc) devem organizar os resultados de suas análises, simulações e testes para serem compartilhados.

O ambiente espacial é constituído de vários elementos que podem influenciar a missão do sistema. Neste trabalho, os agentes externos de interesse para o desenvolvimento do modelo são as forças e os torques externos causados pela órbita (LEO) e a natureza do sistema (satélite de dimensões reduzidas). Como os torques agem diretamente no sistema e a relação bidirecional sistema-ambiente pode ser desconsiderada – o ambiente influencia o sistema infinitamente mais do que este altera o meio – a abordagem por fluxos informacionais, unidirecional, pode ser adotada. Foi utilizado então a notação de diagramas de blocos para implementação do modelo em ambiente computacional.

A partir das hipóteses e equações adotadas em (4.2.1) pode-se construir o diagrama de blocos que representa as perturbações (torques) externos no sistema. A ferramenta computacional de missão e ambiente contém as informações fundamentais do projeto: dimensões principais, materiais (densidade), órbita, inclinação dos painéis, frequência orbital e informações atmosféricas. Essas informações são inseridas pela equipe de Engenharia de Sistemas e alimentam o processo, que faz uso dessas informações diretamente para obter parâmetros secundários. Por exemplo: momento de inércia (SMP), torques de ambiente (ACS) e os resultados de simulações diversos (evolução das variáveis ao longo do tempo) das ferramentas que contém modelos dinâmicos (ACS, PROP, ADE).

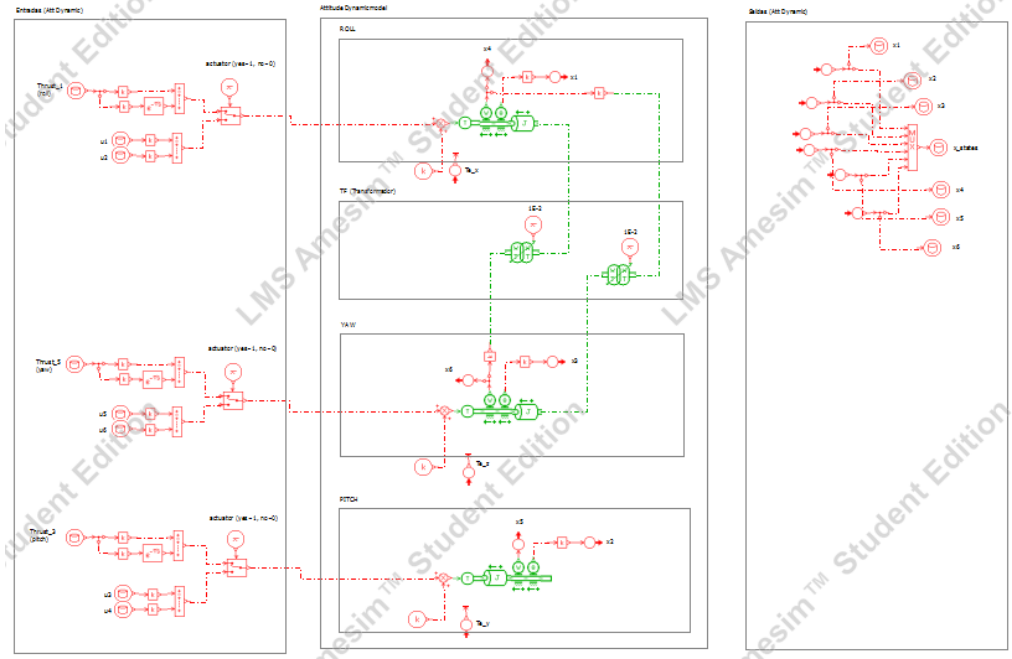
De acordo com o modelo analítico desenvolvido, os torques gravitacionais foram diluídos no modelo de dinâmica de atitude do satélite, ficando implícitos no mesmo, restando apenas na forma explícita os torques magnéticos e aerodinâmicos. Como o objetivo desse trabalho é obter o maior grau de generalidade possível, foi construído um modelo computacional explicitando cada torque, de modo a torná-lo mais apto ao reuso – e assim economizar esforço em novos desenvolvimentos.

Como o ambiente (torques externos) está relacionado diretamente à dinâmica (de atitude), decidiu-se fazer a fusão desses dois modelos – tanto na ferramenta que simula as leis de controle de atitude quanto na ferramenta que reproduz a dinâmica veicular rotacional.

4.3.2. **Dinâmica de Atitude**

O modelo de atitude do satélite é descrito através das equações de movimento. No presente trabalho optou-se pelo uso da representação por fluxos físicos, na Figura 4.27, devido à sua linguagem mais intuitiva, além de seus resultados terem maior grau de fidelidade.

Figura 4.27 – Implementação do modelo GL da dinâmica de atitude e torques externos.



Fonte: Produção do autor.

O modelo foi construído usando o software *AMESim* (*Advanced Modeling Environment for Simulations*), que utiliza GL na modelagem dinâmica de sistemas multidomínio. Ele é eficaz na M&S de sistemas em potência fluida (gases ou líquidos), mecânica, termofluídos, elétricos, magnéticos e mesmo controle. Suas bibliotecas são abertas e permitem personalização, possibilitando a ampliação dos fenômenos considerados (ex: hipóteses particulares). Como essa ferramenta possui interface com o *MatLab*, pode ser usada em co-simulação.

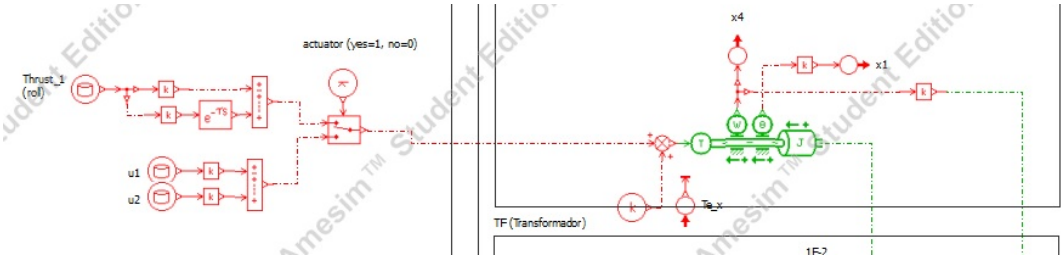
Na Figura 4.27 temos representado as três dimensões de movimento angular, referentes aos ângulos φ (*roll*), θ (*pitch*) ψ (*yaw*). Conforme demonstrado anteriormente, o primeiro e último estão acoplados, sendo para isso necessário um transformador (TF)

que estabeleça a relação entre o torque dado em um e as consequências na variável do outro – e vice-versa. Isso é implementado através do uso de componente específico que executa os princípios do GL.

A dinâmica de *pitch* é desacoplada dos outros movimentos, sendo desnecessário conectar com os componentes $\boldsymbol{I_1}$ e $\boldsymbol{I_3}$.

Como o modelo de atitude pode tanto funcionar como receptor direto do sinal de controle (sem atuador) ou levando em consideração o empuxo (com atuador), foi inserido um *switch* que é acionado ou não a depender do valor de um parâmetro global, repassado pela ferramenta do subsistema propulsivo, através de um arquivo contendo o valor um. Isso pode ser visualizado em detalhes na Figura 4.8. Esse arquivo já é pré-inserido na ferramenta que contém o modelo de atitude com valor zero (não considerar atuador). Caso a cadeia de processos tenha o atuador, o arquivo será automaticamente substituído por outro.

Figura 4.28 – Detalhes da implementação do modelo GL da dinâmica de atitude e torques externos.



Fonte: Produção do autor.

4.3.3. **Satélite**

A literatura classifica o satélite como uma entidade constituída de sete subsistemas interdependentes entre si. Cada qual pode possuir maior ou menor grau de sofisticação relativo às tecnologias existentes. O escopo do presente estudo foi desenvolver três desses sistemas com um grau de sofisticação que oscila do elementar ao intermediário, fornecendo linhas gerais para simulações iniciais de um sistema ainda não projetado, de forma a orientar o desenvolvimento desde as fases iniciais.

A estrutura, que constitui um mundo vasto que pode envolver simulação térmica, aerodinâmica (CFD), de tensões estruturais (CAE) e de projeto (CAD), será vista em seu aspecto de propriedades inerciais, aspectos que estão intimamente relacionados ao controle de atitude e seu atuador.

Dividiu-se o modelo de inércia referente ao satélite em duas partes: sua estrutura básica fixa e o propelente. Enquanto no primeiro o momento de inércia é parâmetro, no segundo ele é variável. Ambos devem ser somados e inseridos no modelo de dinâmica do veículo para permitir resultados de simulação confiáveis.

4.3.3.1. Propriedades de Massa

Os modelos de propriedades de massa foram escritos em código. A partir de requisitos de sistema gerais, o Engenheiro de Sistemas ou de Projeto podem inserir as informações assim que o roteiro é executado. Esse roteiro pode evoluir com o tempo, com adição de novas geometrias pelo Engenheiro de Projeto, aumentando o grau de detalhamento do modelo, e disponibilizando essa informação a todos.

A vantagem de se trabalhar numa rede colaborativa já é evidente: quando um modelo necessário ao Engenheiro de Simulação do SCA (ex. propriedades de massa) é detalhado: o modelo de dinâmica automaticamente terá novos parâmetros, mais precisos, dando resultados mais próximos ao projeto real, sem necessidade de compreender especificidades de projeto.

Os dados de entrada e as saídas do código para a estrutura e o propelente foram tabelados (Tabela 4.5). Maiores detalhes sobre a natureza do código poderão ser visualizados no Apêndice C.

4.3.3.2. Controle de Atitude

Conforme visto nesse trabalho, o SCA engloba o domínio propulsivo (atuador escolhido) e é dependente da estrutura e quantidade de propelente no tanque. Ele é o “cérebro” que deve determinar para onde o veículo deve apontar baseado em referências, posicionamento e leis próprias. A construção dessa lógica parte dos requisitos e é melhor representada através de diagramas de blocos.

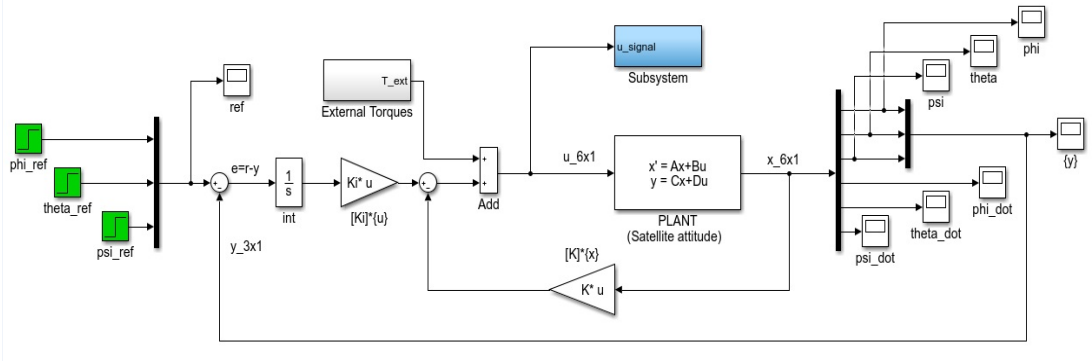
No SCA é feita a fusão de todos os modelos e domínios construídos anteriormente (ambiente espacial, dinâmica de atitude e propriedades inerciais) para obter-se um modelo de controle integrado a domínios de interesse.

O “cérebro” do SCA são as leis de controle. Por trabalhar com informação, seu modelo independe da abordagem física e, portanto, pode ser feito desconsiderando a lei de ação e reação – uma vez que o fluxo de informações é unidirecional. Dessa forma, há uma ampla gama de ferramentas de simulação aptas a implementação do controle, podendo este ser aplicado tanto no *MatLab / Simulink* (fluxos informacionais) quanto no *AMESim* (fluxos físicos) através da simbologia de diagrama de blocos – já feita anteriormente. A primeira delas, por possuir uma biblioteca de sinais mais rica, e componentes com maior grau de detalhamento, além de mais opções quanto a métodos de integração, foi escolhida como ferramenta para representar as leis de controle.

O modelo do controlador LQR é composto de duas partes: um submodelo em código (*Gains_K_Satellite.m*) cujo escopo é calcular os ganhos para o sistema proposto através do método LQR (no Apêndice B, em construção) A saída é a matriz de ganho $\boldsymbol{K_{LQR}} = [\boldsymbol{K} \quad -\boldsymbol{K_i}]$. Esta é usada na implementação (*Control_Satellite.mdl*). A Figura 4.29 revela a implementação da parte informacional do SCA.

O modelo da Figura 4.29 foi usado para modelar a lógica de controle, usando os ganhos calculados via o método LQR. As referências são os ângulos desejados, que podem ocorrer simultaneamente. O modelo de espaço de estados do microssatélite contém os acoplamentos na forma de equações diferenciais. Ele foi posteriormente removido para que o modelo de dinâmica (planta) e propulsivo (atuador) pudessem entrar no processo colaborativo.

Figura 4.29 – Implementação do controle LQR em DB em MatLab/Simulink.

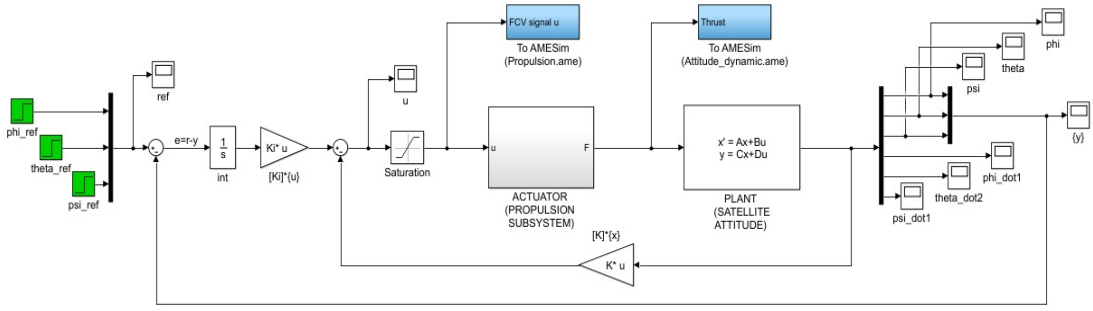


Fonte: Produção do autor.

As interfaces do SCA com o ambiente externo são: o vetor de torques externos $\{\boldsymbol{T_{ext}(t)}\}$; o vetor de estados $\{\boldsymbol{x(t)}\}$; e o vetor de controle $\{\boldsymbol{u(t)}\}$. Os dois primeiros são entradas obtidas de outros modelos. O último é a saída dada pelo SCA.

O modelo do controlador mais completo considera a dinâmica do atuador, linearizada (espaço de estados). Trata-se de uma extensão do modelo do controle de atitude original (SCA) – que, por sua vez, é uma simplificação do modelo completo (SCAO). Sua implementação serve para a execução da cadeia em que o modelo propulsivo é inserido no ciclo integrado controle-atuador-planta, visto na Figura 4.30.

Figura 4.30 – Implementação do controle LQR em DB em MatLab/Simulink considerando o modelo linearizado do atuador.



Fonte: Produção do autor.

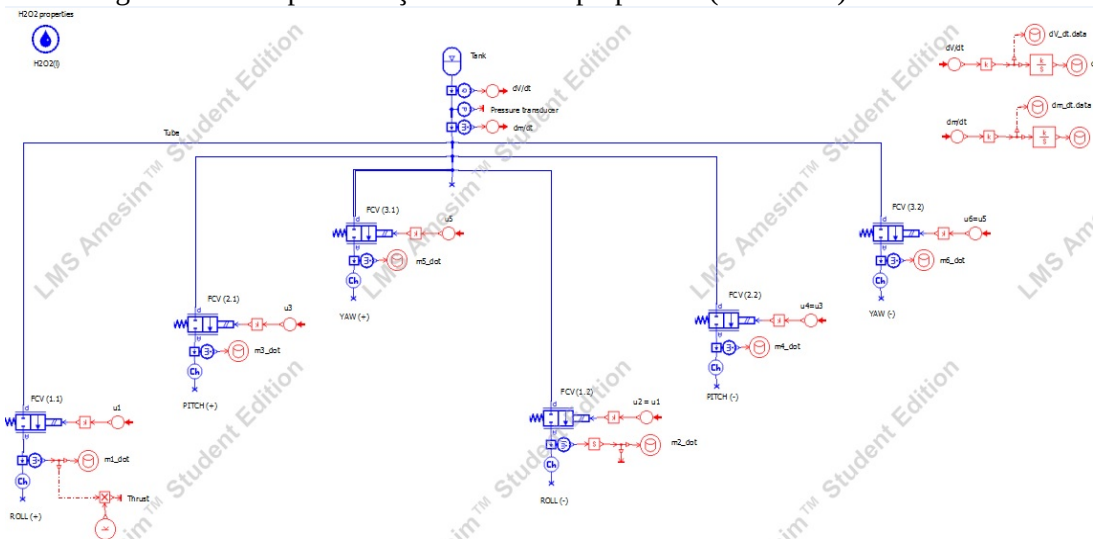
4.3.3.3. Propulsão

O subsistema propulsivo foi representado com dois modelos: um hidráulico, do subsistema em geral, com tanque, tubulações, filtros e válvulas injetoras, isotérmico e incompressível, usando a biblioteca hidráulica. Assim construiu-se o modelo físico do subsistema propulsivo, – sendo a parte hidráulica do subsistema, exibida na Figura 4.31, com componentes de cor azul; e a termodinâmica, exibida na Figura 4.32, cujos componentes possuem cor roxa e marrom –com seus fenômenos característicos, com trocas térmicas e escoamento compressível.

O fenômeno químico é destacado do processo físico por ser de natureza diversa. No entanto, como se está interessado apenas na variável temperatura e pressão de entrada no bocal, é possível criar uma interface entre o domínio hidráulico (FCV) e termo-pneumático (bocal), buscando apenas as variáveis comuns entre os processos físico e químico.

Apesar de desenvolvimento distinto, os modelos propulsivos foram inseridos no mesmo diretório por três motivos: (i) fazem parte do mesmo subsistema; (ii) suas variáveis e parâmetros de projeto estão intimamente relacionadas; (iii) seus domínios são da mesma família (termo fluidodinâmica).

Figura 4.31 – Implementação do modelo propulsivo (hidráulico) em *AMESim*.



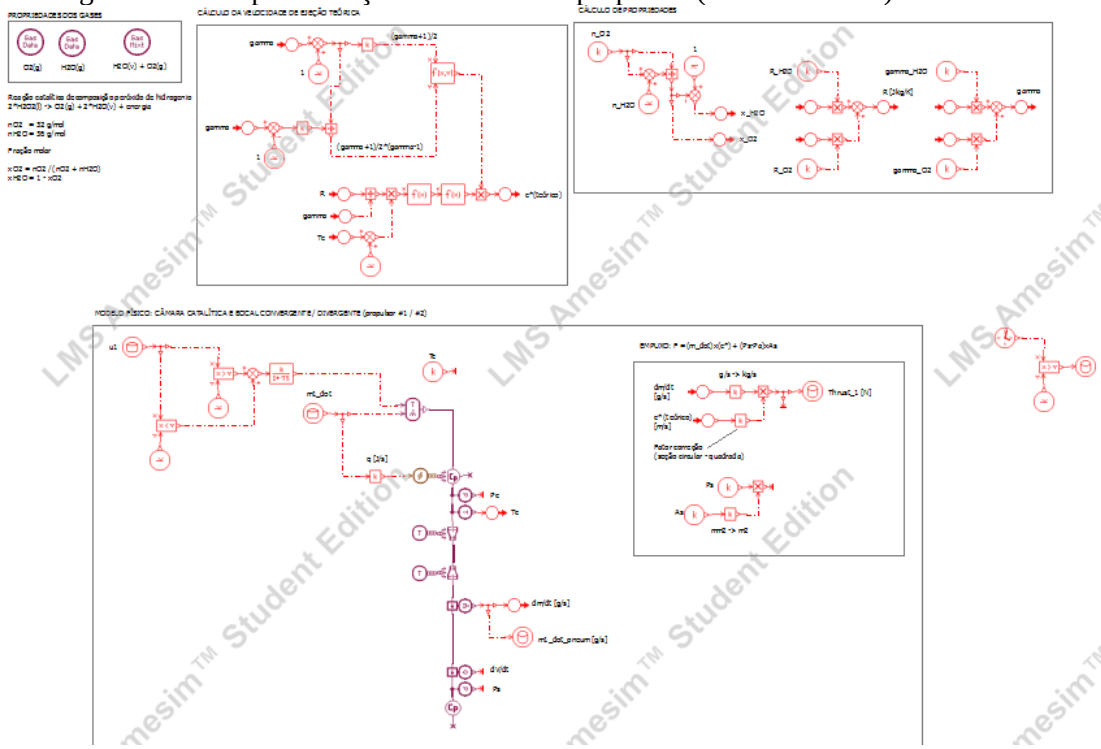
Fonte: Produção do autor.

A Figura 4.31 representa a parte hidráulica do subsistema propulsivo, considerando todos os propulsores. Os parâmetros de cada FCV e da tubulação, em geral, são idênticos e, portanto, podem ser inseridos como globais (*globalparameters*). Dessa forma evita-se um processo laborioso de repetições quando se altera o projeto – o que aumenta a probabilidade de uma transcrição errônea.

O ambiente espacial é modelado como uma câmara com volume muitas vezes superior àquele do tanque (~10^6 vezes maior) com pressão interna próxima a zero.

Os sinais de entrada que determinam o acionamento das válvulas são arquivos fornecidos pela lei de controle. A vazão mássica captada na saída irá alimentar o modelo propulsivo.

Figura 4.32 – Implementação do modelo do propulsor (termodinâmico) em *AMESim*.



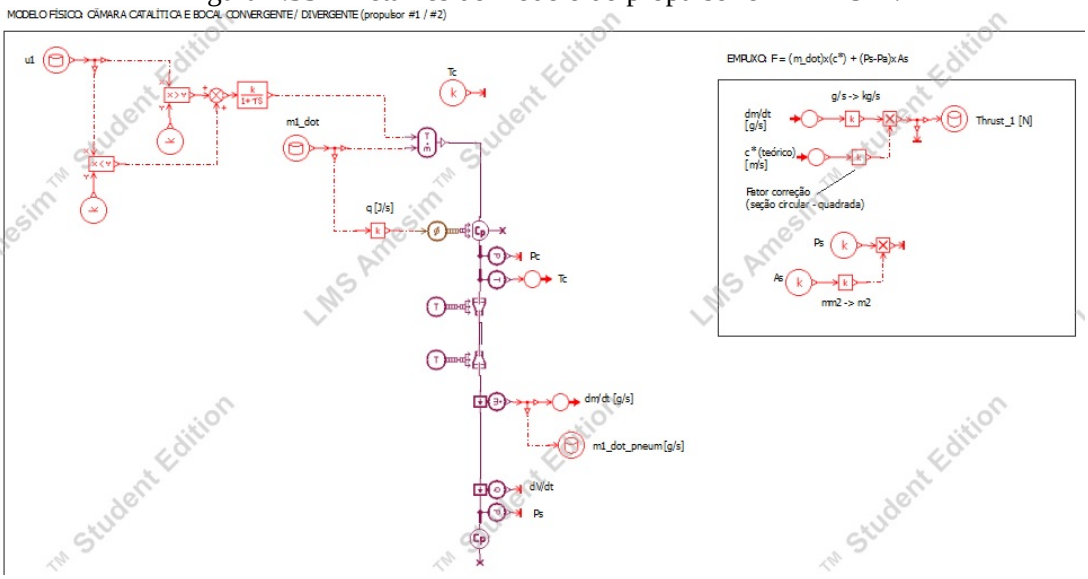
Fonte: Produção do autor.

Conforme dito anteriormente, a reação de decomposição do Peróxido de Hidrogênio é considerada completa e os únicos produtos são vapor de água e Nitrogênio gasoso.

Essas propriedades devem ser inseridas como parâmetros globais, que servirão de base para determinar a velocidade de ejeção teórica, a proporção entre os dois produtos, o coeficiente de calor específico e constante universal da mistura – os três blocos superiores na Figura 4.32.

O bloco inferior, na Figura 4.33, representa a câmara catalítica e o propulsor, cujas entradas são o sinal de controle e a vazão mássica de H_2O_2. O calor liberado na câmara é função da vazão e da entalpia específica da reação (OLIVA, 2012). A temperatura de entrada na câmara se relaciona com a vazão (ou sinal) por uma função de 1ª ordem (OLIVA, 2012). Os arquivos de saída são a variável vazão mássica – ligeiramente diferente da hidráulica devido à dinâmica mais lenta da reação, comparada ao acionamento da FCV – e a pressão de saída, que irão servir para determinar o empuxo, que pode ser visto no bloco à direita, na Figura 4.33.

Figura 4.33 – Detalhes do modelo do propulsor em *AMESim*.



Fonte: Produção do autor.

É importante destacar que, devido à ausência de um componente aeroespacial específico (bocal), existe a necessidade de se determinar o empuxo da forma convencional (paradigma informacional). Daí o bloco de sinais ao lado do submodelo do propulsor. No entanto, pode-se considerar que certas partes do modelo por fluxos de potência não é

integralmente baseado no paradigma, mas possui partes que operam sob o regime da lógica informacional, sendo portanto híbrido – sendo esta uma das possibilidades elencadas anteriormente, no Capítulo 3, no Diagrama de Ferramentas.

5 RESULTADOS

Os roteiros em RCE são escritos em linguagem *Python*. Cada versão deve ser compatível com o ambiente colaborativo. A depender do tipo de ferramenta (abordagem física/informacional, modelo estático/dinâmico), plataforma de simulação, saídas e entradas, o código difere ligeiramente. Os códigos para cada componente se encontram no Apêndice C. Aqui será descrito o fio condutor que irmana os diferentes roteiros, destacando os aspectos mais importantes.

Conforme descrito na Seção 2.4.1, existem três etapas que toda ferramenta deve cumprir. Essas etapas se dão em sequência. Se há erro em uma, a ferramenta cessa as atividades e todas as atividadesda cadeia cessam. Daí a importância de ter claramente definidas as entradas e saídas (pré e pós-execução), os tipos de informação fluindo e quais são as priorizações (R/RIC/NR).

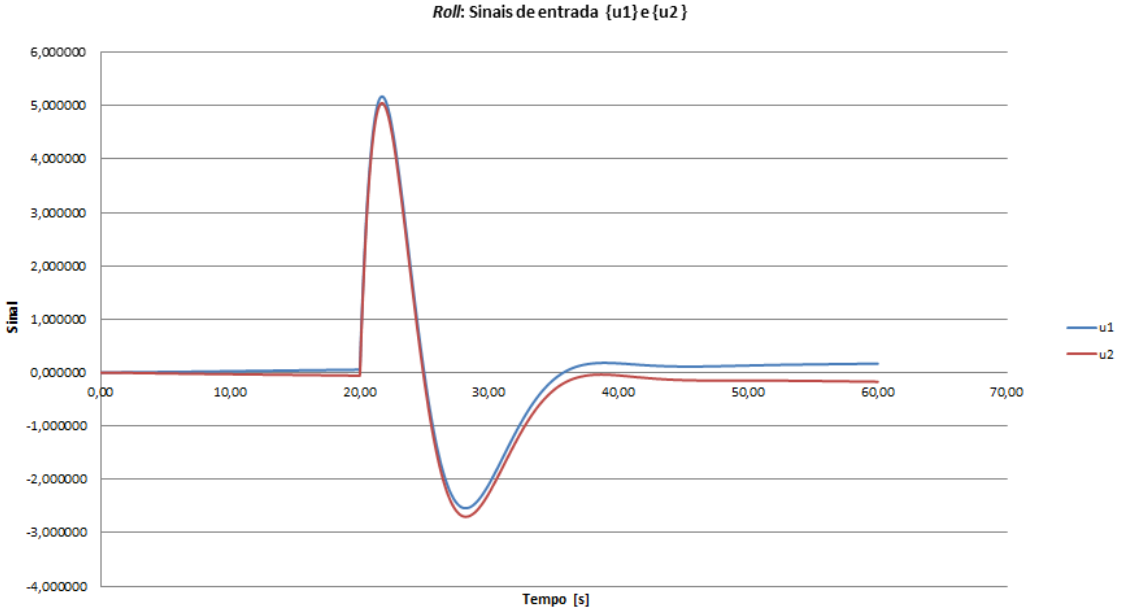
Uma vez iniciado o processo, tudo ocorre de forma autônoma. Não há necessidade de interferência de nenhum nó da rede. Cada parte envolvida no processo pode realizar a execução, com todos os resultados disponíveis a todos. Por outro lado, modificações das ferramentas só podem ser feitas por nós que integraram as mesmas. Aí se vê claramente a distinção entre o respeito ao desenvolvimento individual (base técnica, conhecimento especializado) e a colaboração (base sistêmica, coordenação do processo).

5.1. **Processo 1**

Levando em conta um conjunto de informações iniciais de projeto, foi executada uma cadeia de processos. Na cadeia 1 o objetivo foi observar a coerência de resultados entre as diversas ferramentas, como: (i) checar se os dados de saída de uma ferramenta (SMP) eram os mesmos na entrada de outra dependente da mesma (ACS_v1); (ii) traçar os resultados das variáveis principais (estados e sinais de controle); (iii) mensurar o tempo de execução da cadeia: (iv) verificar possibilidades de otimização do tempo de processo.

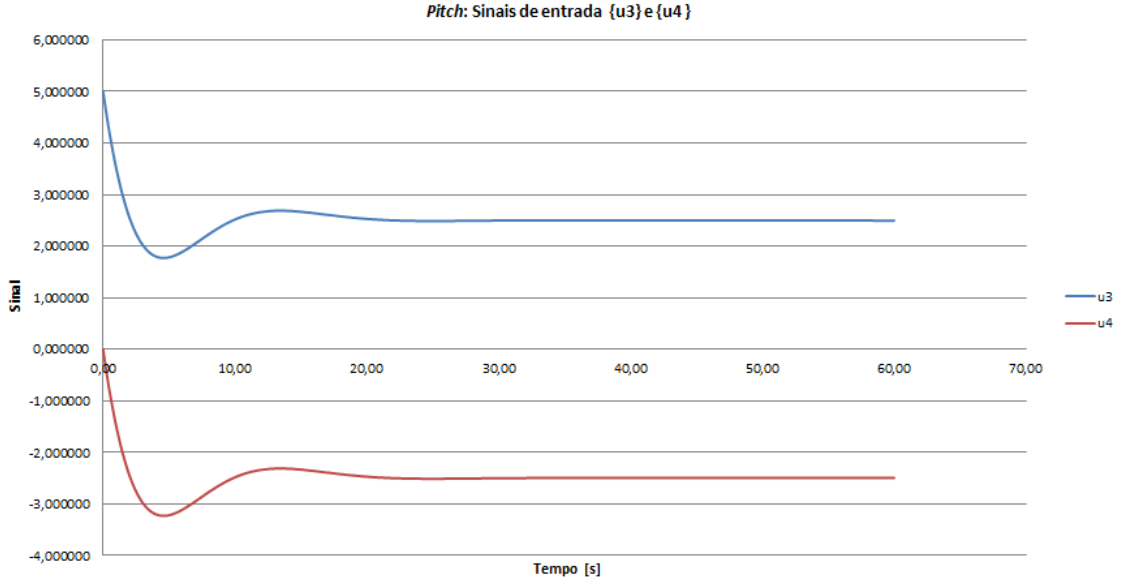
No processo 1 foi executada **uma manobra de *roll* de 10°**, com a observação dos sinais e estados disponibilizados pelas ferramentas. Os resultados foram capturados em forma gráfica manualmente e podem ser visualizados nas Figuras 5.1 a 5.3. Nesse caso, observou-se a relação entre os sinais de controle de ACS_v1 e as saídas (estados) da ferramenta ADE.

Figura 5.1 – Variáveis de saída u1 (azul) e u2 (vermelho) da ferramenta ACS_v1 – *roll*.



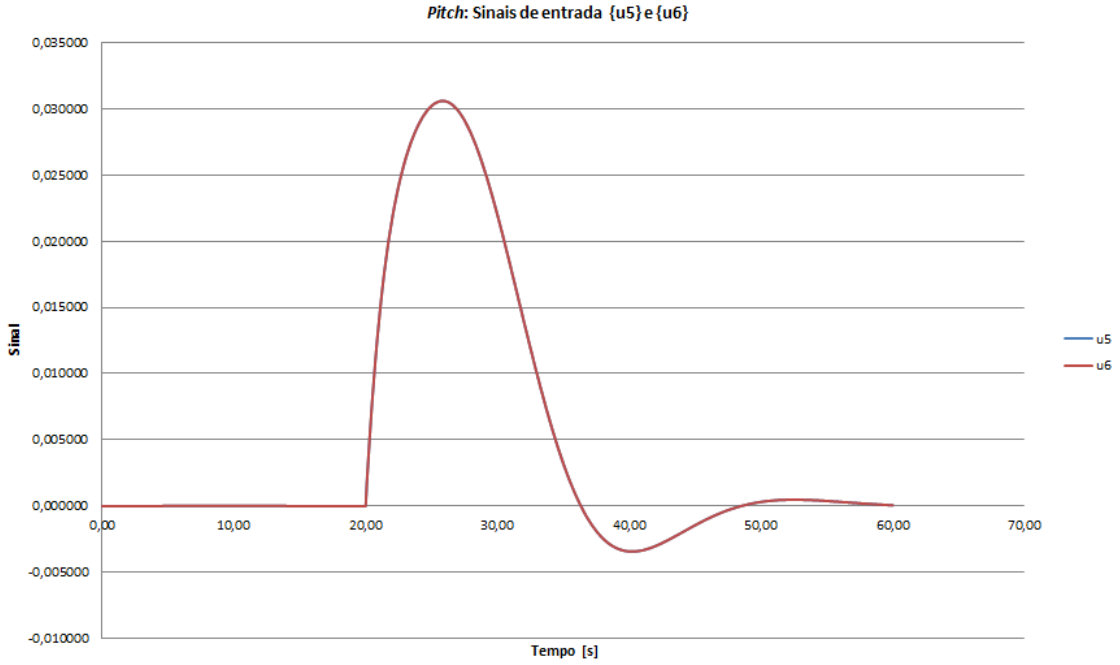
Fonte: Produção do autor.

Figura 5.2– Variáveis de saída u3 (azul) e u4 (vermelho) da ferramenta ACS_v1 – *pitch*.



Fonte: Produção do autor.

Figura 5.3– Variáveis de saída u5 (azul) e u6 (vermelho) da ferramenta ACS_v1 – *yaw*.

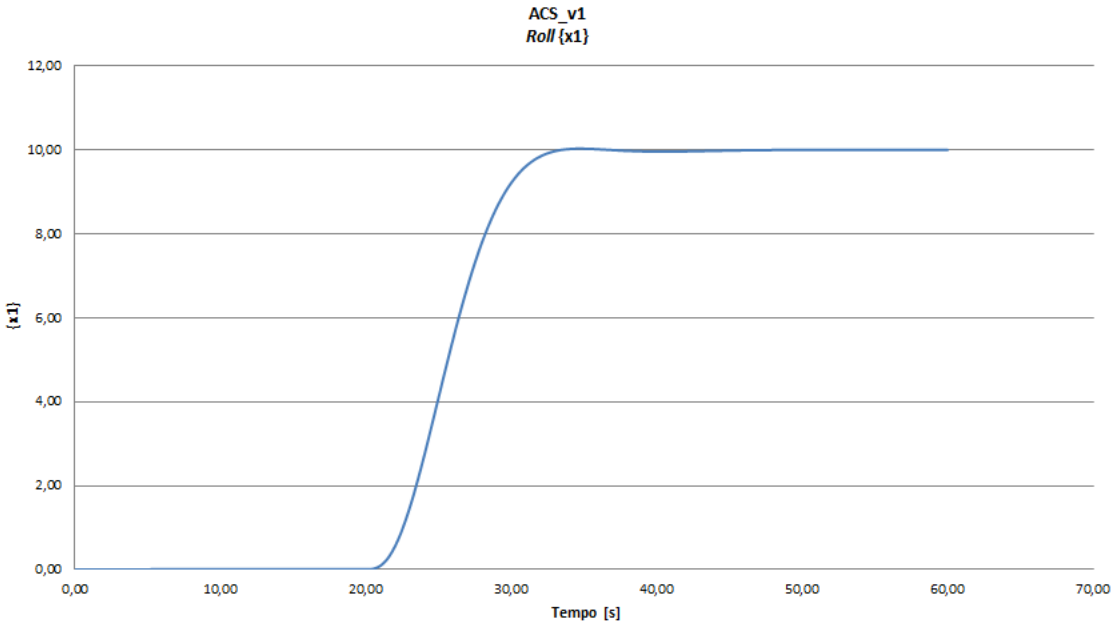


Fonte: Produção do autor.

As variáveis de estado fornecidas pela ferramenta ADE foram muito próximas àquelas da ferramenta ACS_v1 (as curvas coincidem na Figura 5.3), revelando não apenas que o repasse de informações foi coerente, mas que, nesse caso particular, a diferença entre as abordagens informacional e física é pequena. A única exceção foi a constatação de uma diferença entre as curvas de *yaw* na Figura 5.3. Isso se deve provavelmente a um modelamento inadequado das condições ambientais numa ou noutra ferramenta.

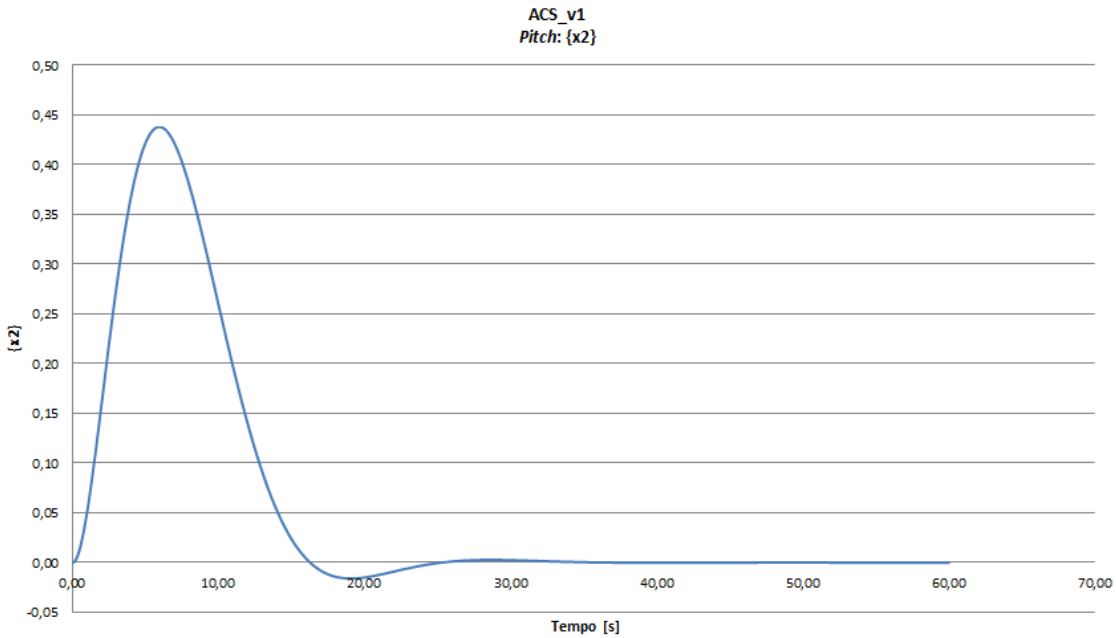
Essa comparação foi possível porque as leis de controle possuíam um modelo de planta em espaço de estados que também fornecia o valor dos estados. Dessa forma, foi estabelecida uma relação entre os estados determinados pelas ferramentas apoiadas nos paradigmas informacional e físico.

Figura 5.4– Diferença entre curvas de *roll* (x1) entre o modelo informacional (azul) e físico (vermelho).



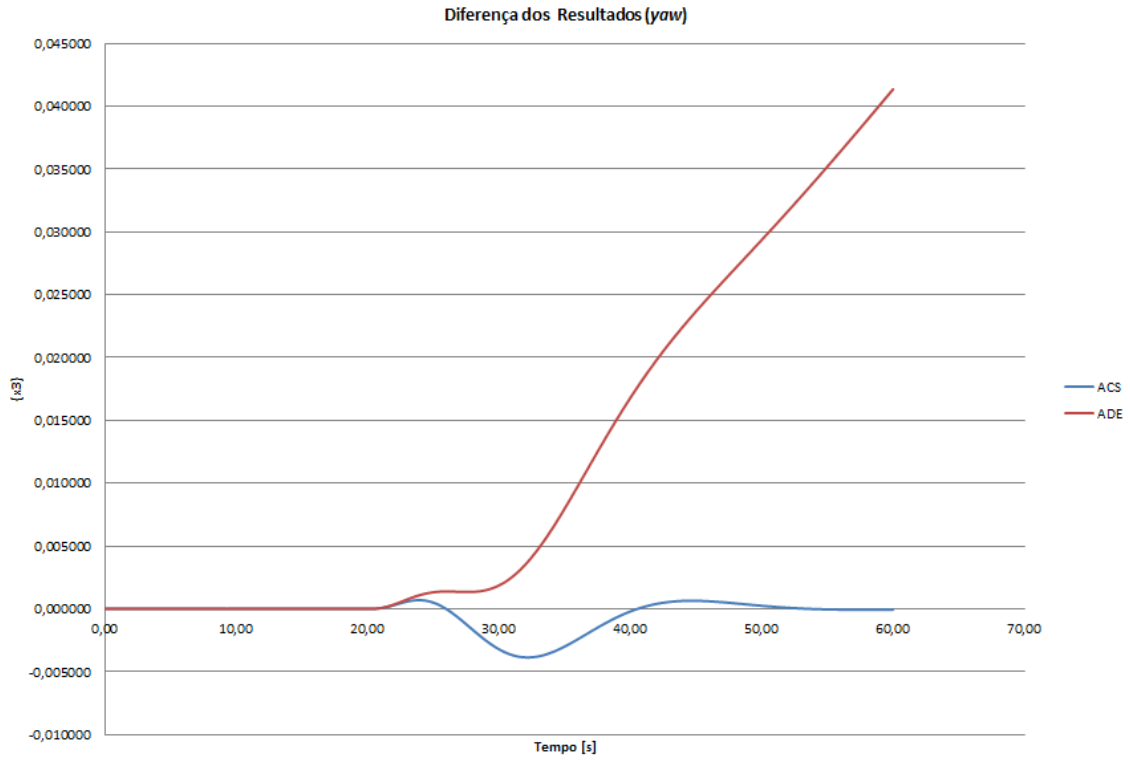
Fonte: Produção do autor.

Figura 5.5– Diferença entre curvas de *pitch* (x2) entre o modelo informacional (azul) e físico (vermelho).



Fonte: Produção do autor.

Figura 5.6– Diferença entre curvas de *yaw* (x3) entre o modelo informacional (azul) e físico (vermelho).



Fonte: Produção do autor.

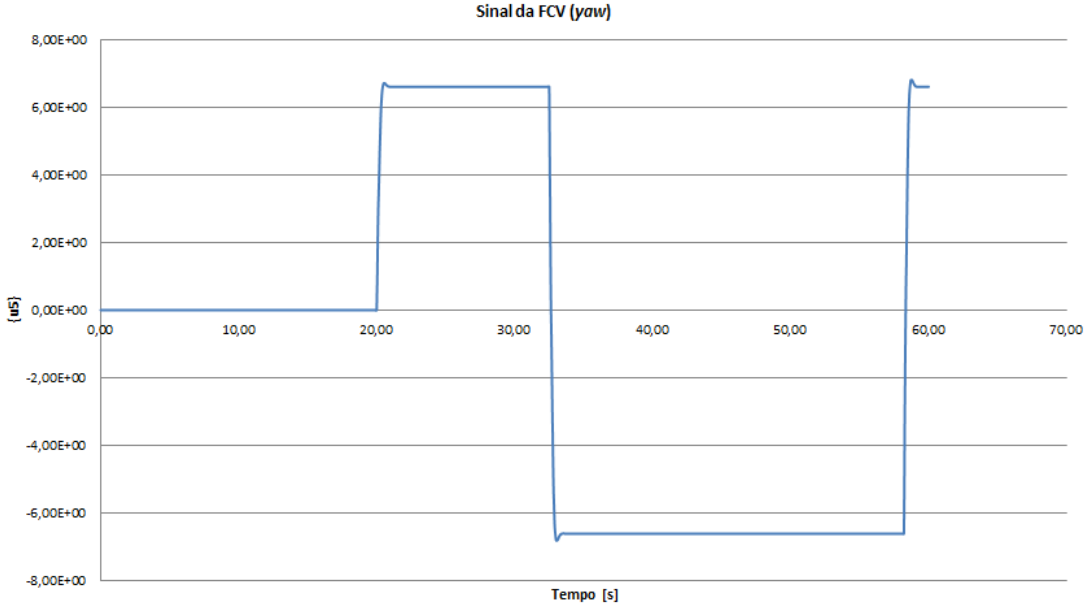
5.2. **Processo 2**

Os mesmos procedimentos foram adotados para a segunda cadeia. Como essa envolve mais ferramentas, os itens (i), (ii) e (iv) possuem mais observações. Nesse caso, puderam ser avaliadas e comparadas outras variáveis (empuxo, vazão, temperaturas, alteração do momento de inércia devido à diminuição de propelente, entre outras), já que ele é mais completo.

Decidiu-se plotar os sinais $\{\boldsymbol{u}\}$ (agora equivalente à força magnética), a vazão de propelente, $\dot{\boldsymbol{m}}_{\boldsymbol{p}}$, o empuxo, $\{\boldsymbol{F}\}$, o impulso específico $\{\boldsymbol{I}_{\boldsymbol{sp}}\}$ e os estados do veículo $\{\boldsymbol{x}\}$.

A execução da 2ª cadeia considerou **uma manobra de *yaw* de 20º**.

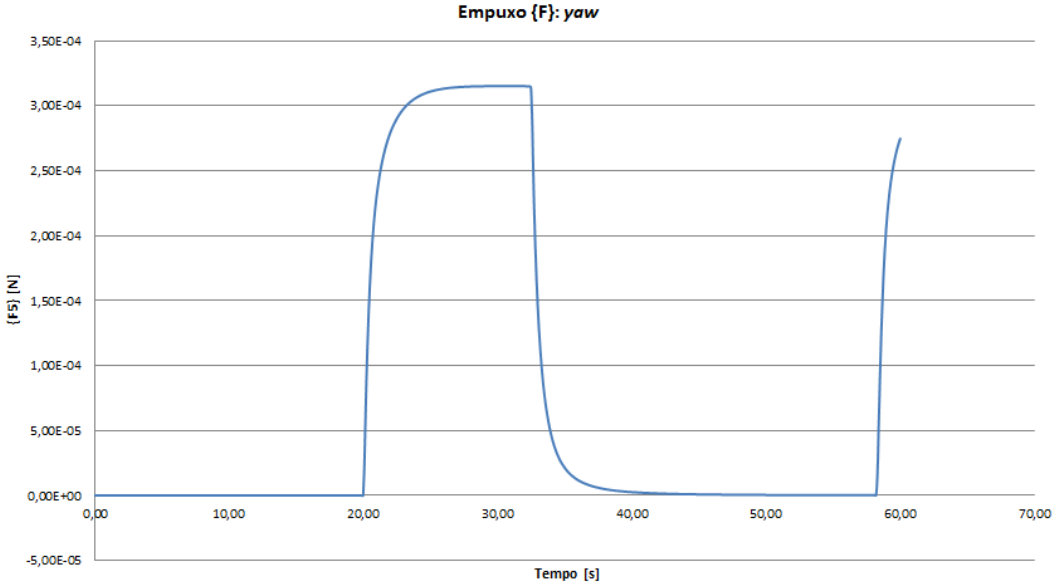
Figura 5.7 – Sinal de controle u5 (FCV) para uma manobra de *yaw* em função do tempo (s).



Fonte: Produção do autor.

O sinal de controle foi obtido da ferramenta ACS_v2, que possui um modelo linearizado do atuador (FCV). Os parâmetros desse modelo são os mesmos daqueles inseridos no modelo hidráulico (subsistema propulsivo). Esses dados podem ser visualizados nos códigos (Apêndice C) e tabelas de parâmetros globais (Apêndice F). O sinal é capturado e enviado às outras ferramentas após a rodada de simulação.

Figura 5.8 – Empuxo (em N) para o sinal de controle u5 em função do tempo (s).

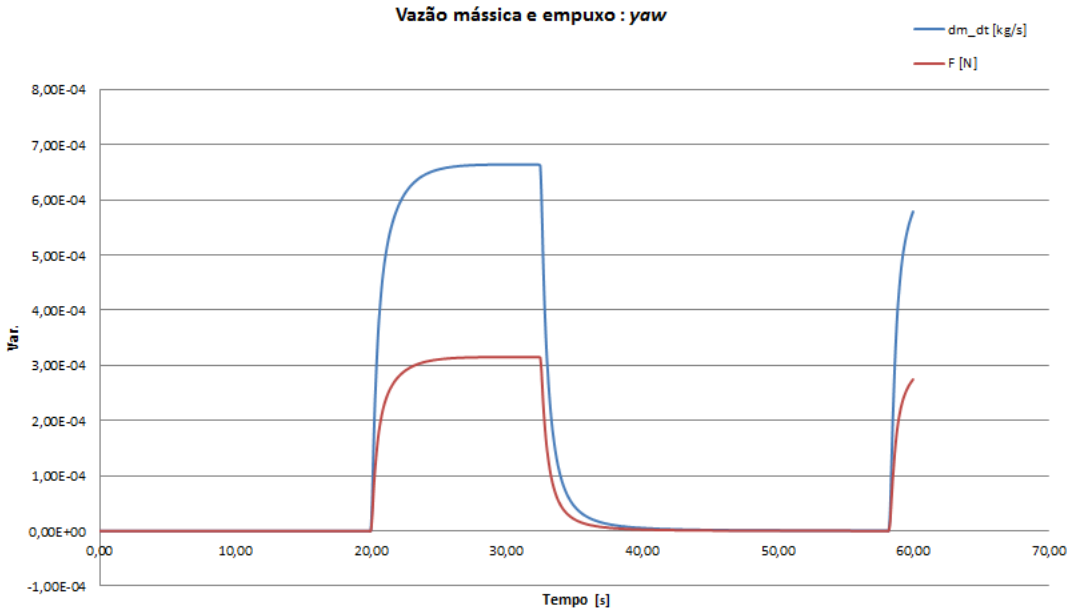


Fonte: Produção do autor.

A magnitude do empuxo atinge um valor em regime permanente próximo àquele de subsistemas de dimensões e natureza semelhante (ver Tabela 4.2). O valor de 310 $\boldsymbol{\mu N}$ está dentro da faixa de operação para sistemas do tipo, revelando que tanto as representações, quanto os parâmetros e a sintonia entre as ferramentas estão satisfatórias.

A vazão mássica, também obtida da ferramenta PROP_2 (que executa e armazena os arquivos do modelo *Thruster.ame*), atinge valor em regime permanente de aproximadamente 660 $\boldsymbol{\mu g/s}$. A literatura especializada (HITT et al., 2001) revela que a magnitude está dentro da faixa de sistemas do tipo.

Figura 5.9– Curvas de empuxo (curva vermelha), em N, e de vazão mássica (curva azul), em kg/s, para a manobra de *yaw*.



Fonte: Produção do autor.

Constatado o funcionamento das cadeias de processos e a ordenação de alguns resultados, partiu-se para o aprofundamento das cadeias de processos com o intuito de torná-las aptas a análises de diversos tipos - comuns nas fases iniciais de desenvolvimento, como o estudo paramétrico.

5.3. **Análise Paramétrica Automatizada**

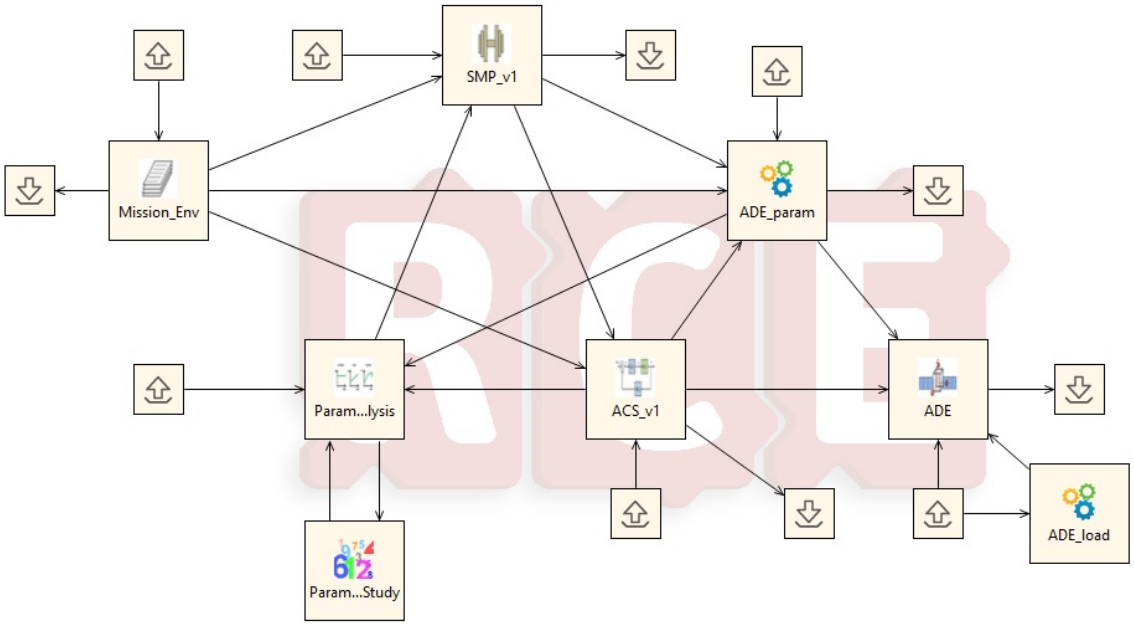
O processo de automação de processos consiste em tornar uma rede autônoma em termos de execuções. Após integrar cada elemento (ferramenta) na rede (processo de desenvolvimento), enriquece-se o repertório de ferramentas as quais todos os nós – da rede – tem acesso. Isso realizado com a metodologia descrita na Seção 3.8. Mas essas integrações visam à construção de um processo. Logo, os elementos devem "conversar entre si", trocando informações adequadas, em sequência definida e fluxos pré-estabelecidos. Isso foi feito e está descrito também na Seção 3.8. Se a execução do processo ocorre e apresenta um fluxo coerente, com troca de informações conforme esperado, tem-se a primeira etapa cumprida. Pode-se então iniciar uma automação desse processo.

A função de automatizar um processo, seja ele produtivo ou de desenvolvimento, é realizar análises para melhorar o projeto. No primeiro caso, visa-se reduzir custos e prazos. No segundo, o escopo é melhorar o modo como a informação flui – e, consequentemente, como o conhecimento é construído.

Serão destacadas as realimentações e os detalhes da execução automática das ferramentas.

A partir do processo 1, foi realizada uma adaptação que possibilitou a análise paramétrica, culminando na cadeia 3, visualizada na Figura 5.10.

Figura 5.10 – Cadeia de processo 3.

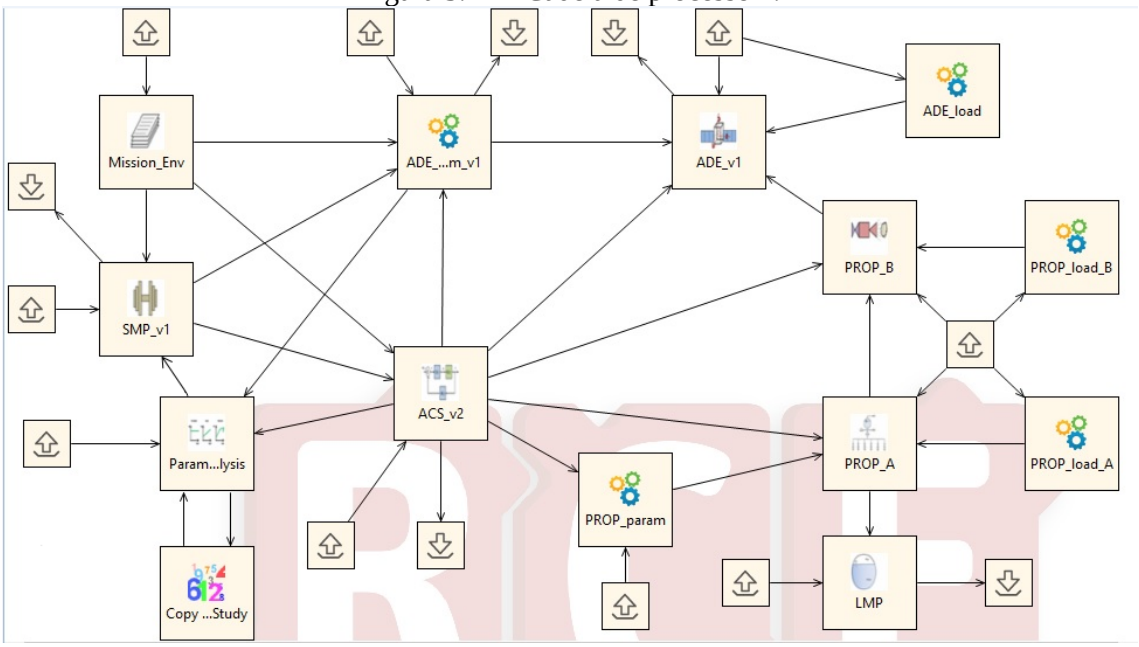


Fonte: Produção do autor.

Duas ferramentas novas foram integradas ao processo: (a) um roteiro (*script*), **Param_Analysis**, com a finalidade de fornecer um arquivo com parâmetros fundamentais – a ser usado pela ferramenta SMP_v1; (b) e uma ferramenta de avaliação (do próprio RCE), **Param_Study**, que visa estabelecer a quantidade de ciclos a serem executados. Esta interage diretamente com o roteiro, de modo a termos um “ciclo aninhado” (*nested loop*).

Escolhe-se a quantidade de iterações do processo no componente de estudo paramétrico. O primeiro passo consiste em estabelecer um valor inicial, um valor final e o passo de cada rodada. O roteiro é acionado cada vez que as entradas estão disponíveis. É importante lembrar os requisitos de automação (Capítulo. 3) referentes ao consumo das variáveis do circuito.

Figura 5.11 – Cadeia de processo 4.



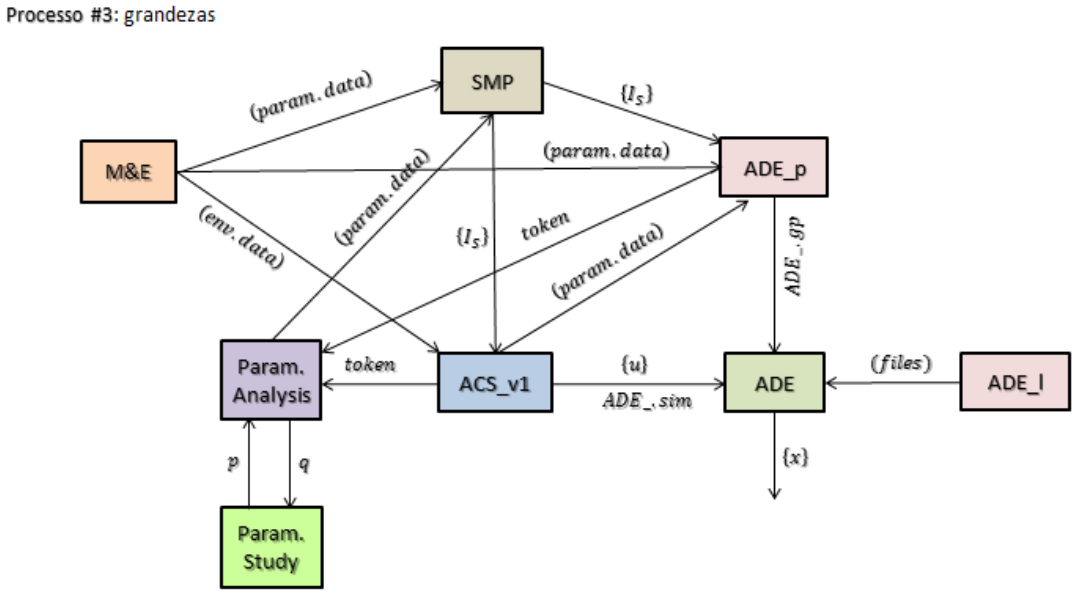
Fonte: Produção do autor.

Assim como os processos 1 e 2, construiu-se um diagrama simplificado para os processos 3 e 4, estabelecendo relações de grandezas e conexões (Figuras 5.12 a 5.15). Percebe-se que a diferença entre estes e os processos fundamentais consiste na inserção das duas ferramentas citadas e nas relações provenientes. É importante lembrar que, como se trata de um processo a ser executado várias vezes – com alteração dos arquivos de parâmetros de entrada a cada rodada –, é necessário definir bem os tipos de conexões da perspectiva de modo de lidar com os dados (*input handling*). Nesse caso, os arquivos que contém os parâmetros fundamentais de projeto, dados pela ferramenta de missão e ambiente (Mission_Env), devem ter seus receptores configurados para tratar esses dados como consumíveis (*consumed*). Ou seja, assim que chegar um arquivo do tipo, após o uso desse pela ferramenta, ele deverá ser descartado e na próxima rodada a ferramenta exige um novo, diverso.

A diferença entre tratar os dados consumidos como fila (*queue*) ou únicos (*single*) está no acúmulo do histórico de arquivos no primeiro ou na eliminação do último arquivo de dados recebido. Esse tipo de escolha dependerá da necessidade de cada parte (nó) no projeto.

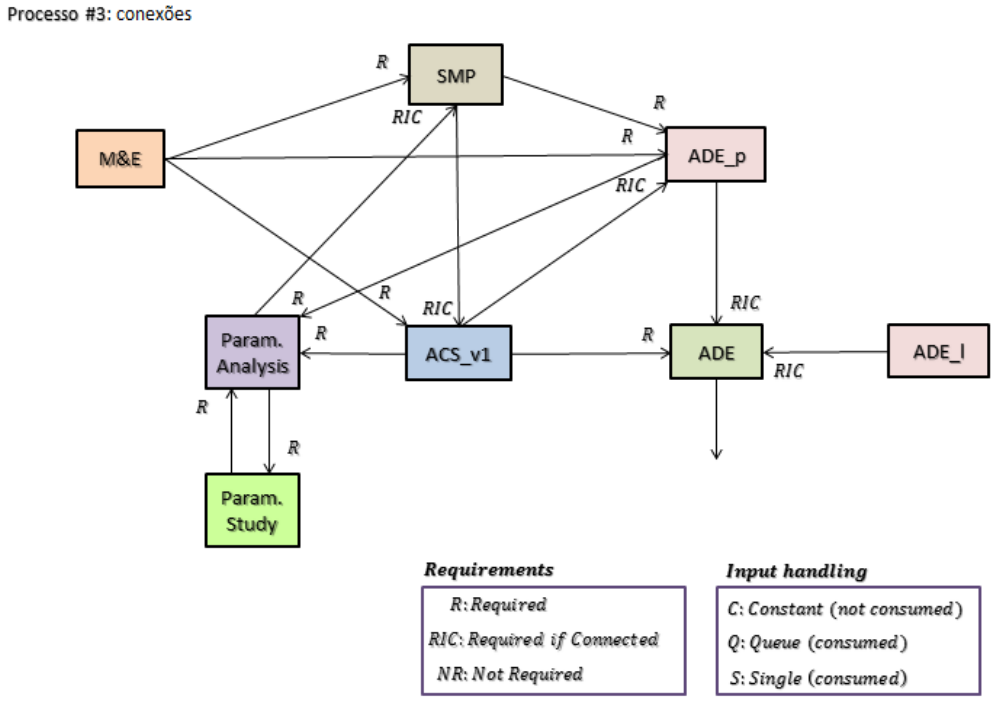
Os diagramas simplificados referentes ao processo 3, das Figuras 8.28 e 8.29, e ao processo 4, dasFiguras 5.12 e 5.13, explicitam as especificidades sistêmicas das respectivas cadeias.

Figura 5.12 – Esquema simplificado do processo 3 (grandezas).



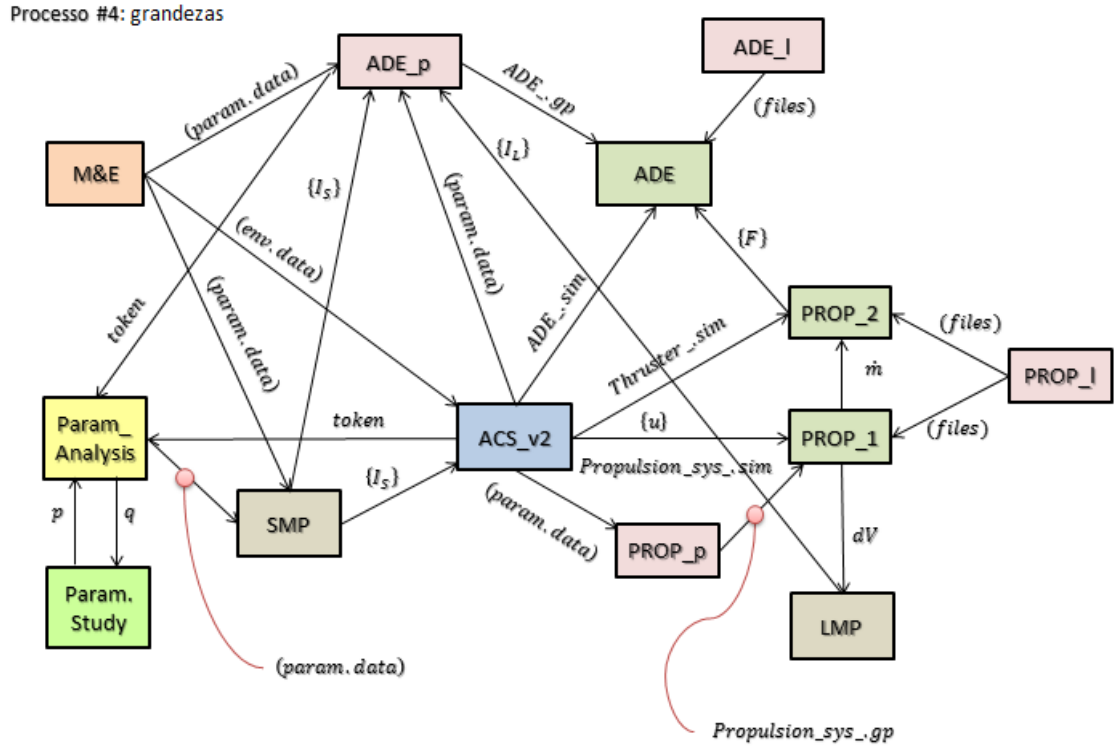
Fonte: Produção do autor.

Figura 5.13 – Esquema simplificado do processo 3 (conexões).



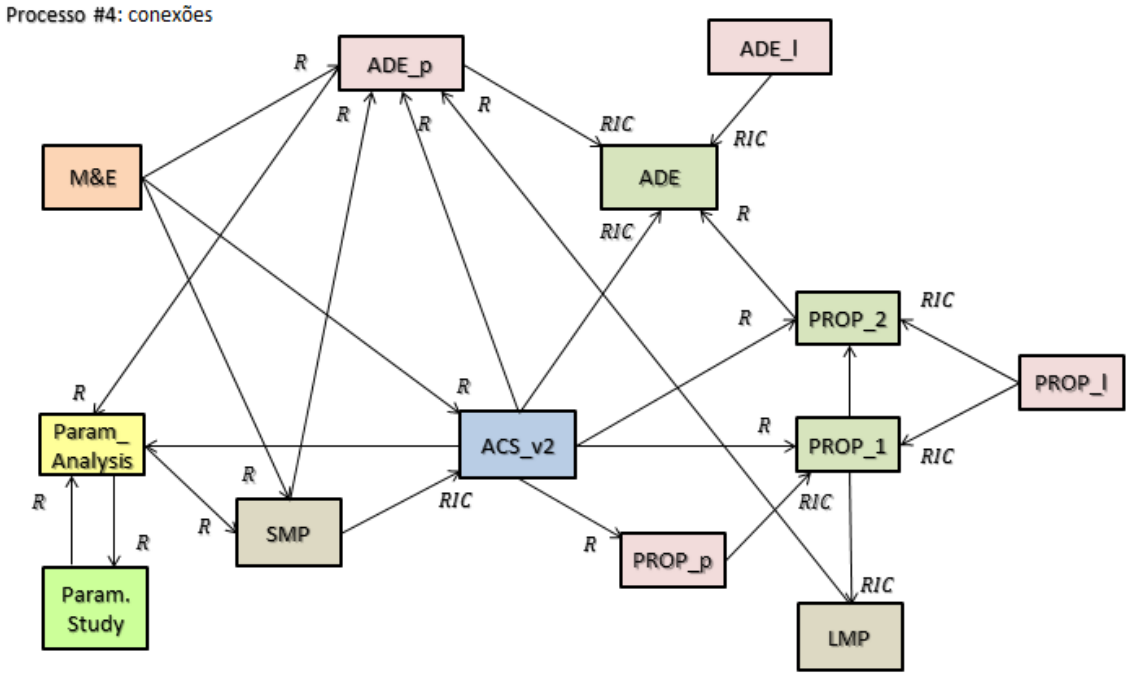
Fonte: Produção do autor.

Figura 5.14 – Esquema simplificado do processo 4 (grandezas).



Fonte: Produção do autor.

Figura 5.15 – Esquema simplificado do processo 4 (conexões).



Fonte: Produção do autor.

Aqui interessa definir um arquivo ou conjunto de arquivos, aliado a um roteiro de execução (a ferramenta *script*), responsável por uma variação de um ou mais parâmetros, para a obtenção de resultados que permitam estabelecer relações entre parâmetros (análise paramétrica global) ou entre diversas variáveis (análise paramétrica local).

O objetivo desse tópico é fazer uma análise paramétrica que permita estabelecer algumas relações entre parâmetros de entrada.

Os parâmetros a serem alterados num estudo dessa natureza são (sempre) os primários (dimensões e materiais (densidade), entre outros). Consequentemente, mudam-se parâmetros secundários (momentos de inércia, centro de massa, etc.) e as variáveis das corridas de simulação. É importante hierarquizar constantes universais, parâmetros e variáveis.

Constantes universais:

São grandezas que não variam em hipótese alguma, sendo constantes para todos os sistemas e fenômenos físicos, químicos, biológicos e humanos do universo conhecido. A determinação das mesmas é objeto de estudo das ciências “duras”.

Parâmetros:

São grandezas consideradas invariáveis num determinado contexto (conjunto de experiências ou simulações). Eles podem ser classificados como primários ou fundamentais quando servem de base para a determinação de outros parâmetros (secundários). São tratados como constante numa corrida de simulação.

Variáveis:

São grandezas que variam ou não ao longo do tempo (ex: posição de um veículo no espaço, consumo de combustível, temperatura de um componente) ou do espaço (ex: deflexão de uma viga ao longo de uma dimensão). Elas podem ser independentes ou dependentes. Por exemplo, se A depende de B, então A = A(B), ou seja, A é influenciada pela variação de B – mas o inverso não ocorre.

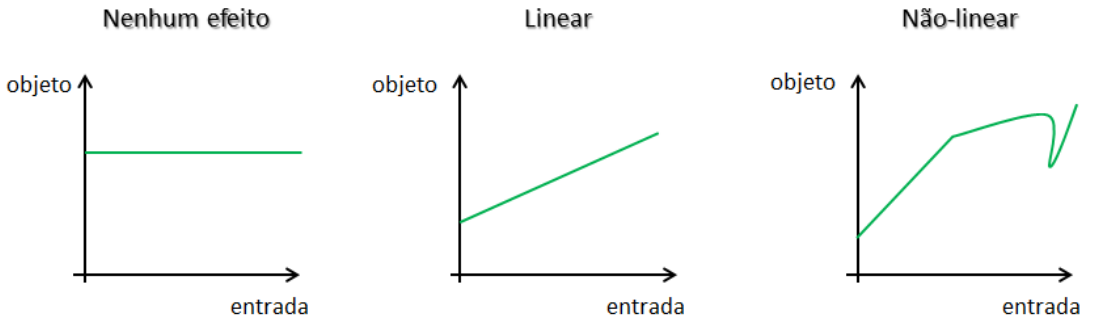
No trabalho anterior do autor (OLIVA, 2012), o foco residia na simulação em si, que trata da relação entre as variáveis. No presente estudo, desloca-se esse foco para os parâmetros – adentrando-se no mundo do engenheiro de projeto. Apesar disso, percebe-

se a relação entre ambos os profissionais (projeto e simulação). É justamente na fase inicial de projeto que coexistem esses mundos (projeto e simulação), cujos delineamentos gerais são estabelecidos pela Engenharia de Requisitos e Sistemas (requisitos e especificações).

Decidiu-se variar de 1 a 3 parâmetros simultaneamente (por rodada). Foram definidos três valores para cada parâmetro (baixo, médio e alto). A partir daí, foram tabelados alguns resultados de interesse fornecidos pelas ferramentas. As saídas foram plotadas em gráficos, na qual o eixo das abcissas indicam o valor do parâmetro e o eixo das ordenadas o valor de uma (ou mais) variável de interesse. O objetivo é estabelecer uma relação (se houver) entre as duas.

De modo geral, podemos ter três tipos de comportamento: (i) não há relação causa-efeito entre as grandezas; (ii) existe uma relação linear entre ambas; ou (iii) existe uma relação não linear entre as grandezas. A Figura 5.16 ilustra essas três possibilidades.

Figura 5.16 – Possíveis resultados de uma análise paramétrica.



Fonte: Produção do autor.

O ambiente colaborativo permite não apenas que esse tipo de análise seja feita de modo mais completo – pois geralmente envolve estabelecer relações entre disciplinas muito distintas –, mas também possibilita o compartilhamento de informações entre todos os nós da rede e a automação do processo. Este último permite que toda a coleta, tratamento e disponibilização de informações seja direcionada de modo mais eficaz.

Foram alterados dois parâmetros de entrada simultaneamente, com a coleta de dados de alguns parâmetros secundários.

A partir das cadeias adaptadas puderam ser feitas diversas análises paramétricas sem a necessidade de interferência humana ao longo dos ciclos. Foram variados alguns parâmetros de entrada simultaneamente (mesmo ciclo de execução) e, a partir daí, foi constatado como parâmetros secundários – calculados a partir destes e de outros dados e de entrada – se alteravam a cada ciclo, além de curvas de variáveis importantes.

Foram feitas três análises para um grupo de parâmetros, alterando largura do corpo do satélite, altura do painel solar e densidade média dos painéis (Tabela 5.1); e duas análises para outro grupo de parâmetros, com variação da altura do corpo do satélite e do raio do tanque (Tabela 5.2). É importante destacar que, nesse segundo caso, as duas grandezas estão relacionadas: o raio do tanque não pode exceder a menor dimensão do veículo (largura, comprimento ou altura).

Um valor "original" (inicial) foi adotado e a partir deste feitas três variações, que resultaram nos três casos: um valor alto, outro médio e um último baixo (destaque em vermelho).

Tabela 5.1 – Parâmetros de entrada alterados no 1º grupo.

FERRAMENTA MISSION_ENV / Param_Analysis								
Nº	Parâmetros de entrada [unidade]	Análise paramétrica				1ª análise			
		Valor original	baixo	médio	alto	1ª	2ª	3ª	
1	a: largura sat.[m]	0,400	-0,100	0,000	0,100	0,300	0,400	0,500	GRUPO 1
2	b: comprimento sat. [m]	0,300				0,300	0,300	0,300	
3	c: comprimento painel [m]	0,700	-0,100	0,000	0,100	0,600	0,700	0,800	
4	h1: altura painel [m]	0,400				0,400	0,400	0,400	
5	h2: altura sat. [m]	0,300				0,300	0,300	0,300	
6	e1: espessura corpo satelite [m]	0,010				0,010	0,010	0,010	
7	e2: espessura paineis [m]	0,020				0,020	0,020	0,020	
8	Ri: raio interno do tanque [m]	0,290				0,290	0,290	0,290	
9	Re: raio externo do tanque [m]	0,300				0,300	0,300	0,300	
10	R: raio medio do tanque [m]	0,295				0,295	0,295	0,295	
11	L: disância entre painéis e corpo sat. [m]	0,050				0,050	0,050	0,050	
12	ro1: densidade media do corpo satelite [kg/m3]	2000,000				2000,000	2000,000	2000,000	
13	ro2: densidade media paineis [kg/m3]	2000,000	-100,00	0,00	100,00	1900,000	2000,000	2100,000	
14	ro3: densidade tanque esferico [kg/m3]	2000,000				2000,000	2000,000	2000,000	

Fonte: Produção do autor.

Numa 2ª análise, foi feito um estudo relacionando diferentes propelentes a serem usados para o mesmo subsistema e configurações. Peróxido de Hidrogênio ($\boldsymbol{H_2O_2}$) e Hidrazina ($\boldsymbol{N_2H_4}$) foram comparados em termos de impulso específico ($\boldsymbol{I_{sp}}$). Como cada um tem um conjunto de propriedades, estas podem ser relacionadas diretamente com essa variável de eficiência propulsiva.

Tabela 5.2 – Parâmetros de entrada alterados no 2º grupo.

Nº	Parâmetros de entrada [unidade]	Análise paramétrica				2ª análise			
		Valor original	baixo	médio	alto	1ª	2ª	3ª	
1	a: largura sat.[m]	0,400				0,400	0,400	0,400	GRUPO 2
2	b: comprimento sat. [m]	0,300				0,300	0,300	0,300	
3	c: comprimento painel [m]	0,700				0,700	0,700	0,700	
4	h1: altura painel [m]	0,400				0,400	0,400	0,400	
5	h2: altura sat. [m]	0,300	-0,050	0,000	0,050	0,250	0,300	0,350	
6	e1: espessura corpo satelite [m]	0,010				0,010	0,010	0,010	
7	e2: espessura paineis [m]	0,020				0,020	0,020	0,020	
8	Ri: raio interno do tanque [m]	0,290	-0,050	-0,025	0,000	0,240	0,265	0,290	
9	Re: raio externo do tanque [m]	0,300				0,300	0,300	0,300	
10	R: raio medio do tanque [m]	0,295				0,295	0,295	0,295	
11	L: disância entre painéis e corpo sat. [m]	0,050				0,050	0,050	0,050	
12	ro1: densidade media do corpo satelite [kg/m3]	2000,000				2000,000	2000,000	2000,000	
13	ro2: densidade media paineis [kg/m3]	2000,000				2000,000	2000,000	2000,000	
14	ro3: densidade tanque esferico [kg/m3]	2000,000				2000,000	2000,000	2000,000	

Fonte: Produção do autor.

As propriedades em questão são aquelas dos produtos de reação. Nesse caso ter-se-á água e oxigênio, no caso do propelente $\boldsymbol{H_2O_2}$; e Nitrogênio e Hidrogênio no caso do propelente $\boldsymbol{N_2H_4}$ (Tabela 5.3).

Tabela 5.3 – Propriedades dos produtos de reação catalítica dos propelentes.

ANÁLISE DE PROPELENTES		Caso 1		Caso 2		
		Perox. Hidrogênio (H2O2)		Hidrazina (N2H4)		
Produtos da reação -->		H2O(v)	O2(g)	N2(g)	H2(g)	Unidade
Propriedades	Calor específico: cp	-	-	-	-	[kJ/kgK]
	Calor específico: cv	-	-	-	-	[kJ/kgK]
	Razão cp/cv: gamma	1,327	1,393	1,400	1,409	[ ]
	Cte. Universal do gás: R	461,5	259,8	296,0	4124,0	[kJ/kgK]
	Massa molar: n	18,0	32,0	28,0	2,0	[g/mol]
	Fração molar: x	0,529	0,471	0,875	0,125	[ ]
	Temp. estagnação: To	1200		1200		[K]
	gamma médio	1,358		1,401		[ ]
	R médio	366,6		774,5		[kJ/kgK]

Fonte: Produção do autor.

Foi necessário criar uma ferramenta adicional a título de alterar propriedades dos respectivos gases de exaustão para a execução da ferramenta PROP_2 (que contém o modelo do propulsor, *Thruster.ame*). Essa ferramenta captura informações de propriedades do propelente (ferramenta LMP), que já relaciona o combustível com seus

produtos de reação. E como será gerado um arquivo de parâmetros globais para o modelo do propulsor, a ferramenta que abre o modelo e insere os arquivos do mesmo deve ser alterada, para que ela não mais repasse para a ferramenta PROP_2 os dados de parâmetros globais (*Thruster_.gp*). Dessa forma, foram feitas adaptações de três ferramentas (PROP_2, PROP_2_load e LMP) e criada uma nova (*Thruster_param*). A Figura 8.33 ilustra a conexão dessa ferramenta com as outras.

A entrada de *Thruster_param* são os arquivos contendo informações dos gases de exaustão: cada arquivo se refere e um combustível (no caso do estudo, apenas dois). Sua saída é um arquivo de parâmetros do mesmo formato usado pelo modelo.

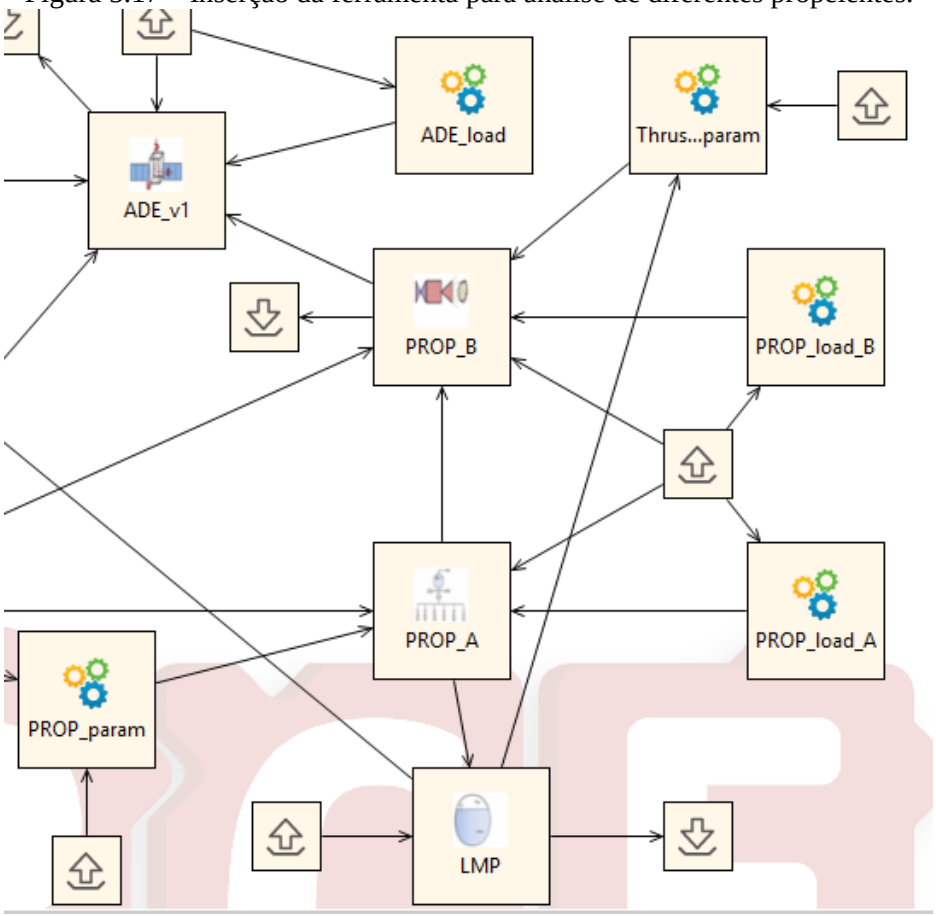
A adição desse ingrediente requer o remodelamento local de algumas ferramentas. Percebe-se a característica sistêmica de um projeto dessa natureza – e a importância de se guiar por requisitos de conexão e sequenciamento. Ao mesmo tempo, esse incremento torna a cadeia mais completa sem necessariamente aumentar a carga dos nós – já que podem haver vários pontos para realizar cada tarefa.

Para estabelecer a comparação entre propelentes, deve-se obter os resultados de vazão mássica e empuxo de cada caso (simulação / experimento) e compará-los. A relação que determina o impulso específico na prática é descrita por:

$$\boldsymbol{I_{sp} = \frac{F(t)}{\dot{m} \cdot g}} \qquad (5.1)$$

Em que $\boldsymbol{F(t)}$ é o empuxo (em N), $\boldsymbol{\dot{m}}$ a vazão mássica de propelente (em kg/s) e $\boldsymbol{g}$ a aceleração da gravidade (em $\boldsymbol{m/s^2}$) – que é função da órbita (altura). Para uma órbita LEO de 400 km, foi calculado que a aceleração da gravidade equivale a 8,75 $\boldsymbol{m/s^2}$ (a fórmula e informações disponíveis podem ser visualizadas nos códigos, Anexo C).

Figura 5.17 – Inserção da ferramenta para análise de diferentes propelentes.



Fonte: Produção do autor.

Adicionalmente, é possível fazer o cálculo do máximo valor teórico, e assim estabelecer umacomparação entre os casos reais – de teste ou simulação, caso seja possível validar o modelo – e o ideal. Essa relação é dada por HITT (2001):

$$\boldsymbol{I_{sp} = \sqrt{\frac{2 \cdot \gamma \cdot R \cdot T_0}{g^2 \cdot (\gamma - 1)}}} \qquad (5.2)$$

Em sequência são apresentados os resultados de execução para cada caso descrito.

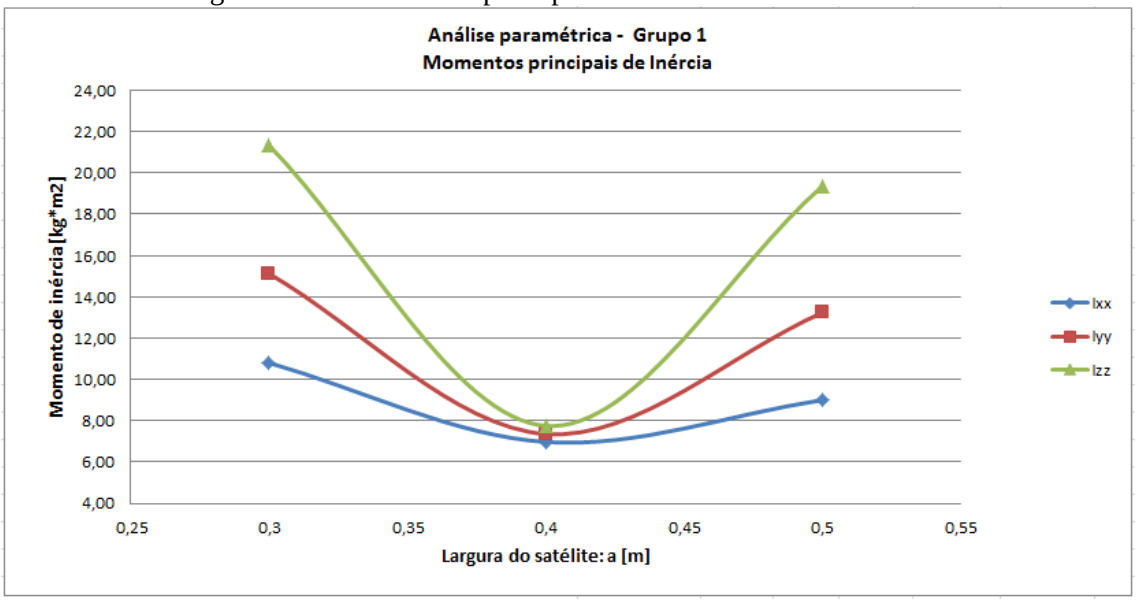
5.3.1. **Execução Distribuída Automatizada**

As execuções em três ciclos iniciais (Tabela 5.1) geraram um grupo de dados que foram ordenados de forma gráfica. As primeiras relações estabelecidas foram entre as

dimensões **a** (largura do satélite) e **c** (largura da placa solar), e os momentos de inércia principais. Elas podem ser visualizadas nas Figuras 5.18 e 5.19. Uma curva ligando os pontos foi feita apenas para ilustrar uma possível forma de relação.

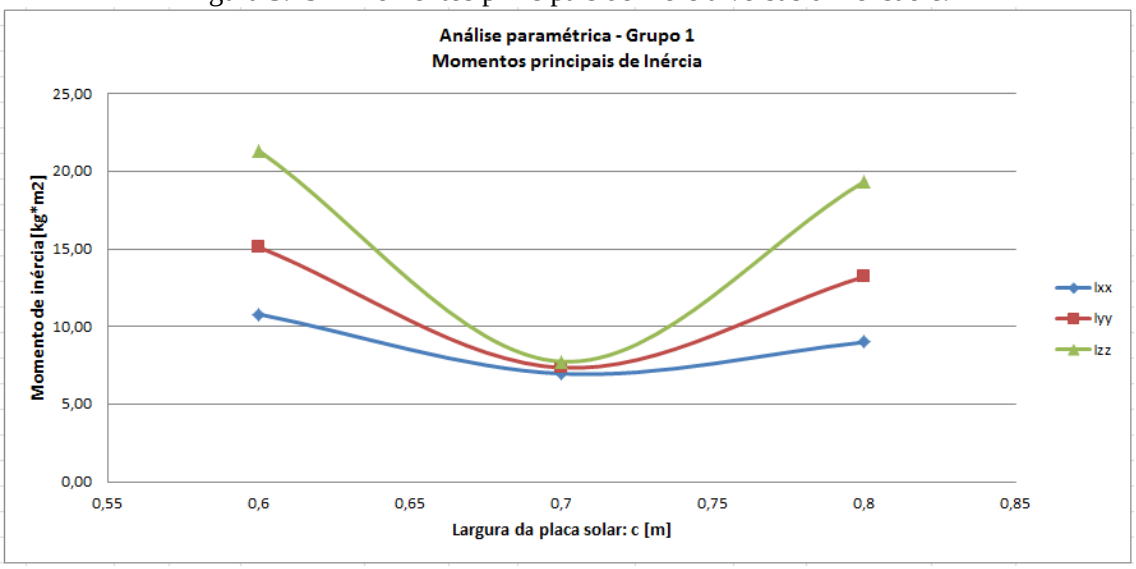
Observa-se que, para os valores intermediários dos parâmetros de entrada, há uma convergência dos parâmetros secundários. Isso tem consequências nos comportamentos das variáveis de simulação, conforme pode ser visto nas figuras subsequentes, que traçam o sinal u1 e o estado x1 (*roll*).

Figura 5.18 – Momentos principais de inércia versus dimensão a.



Fonte: Produção do autor.

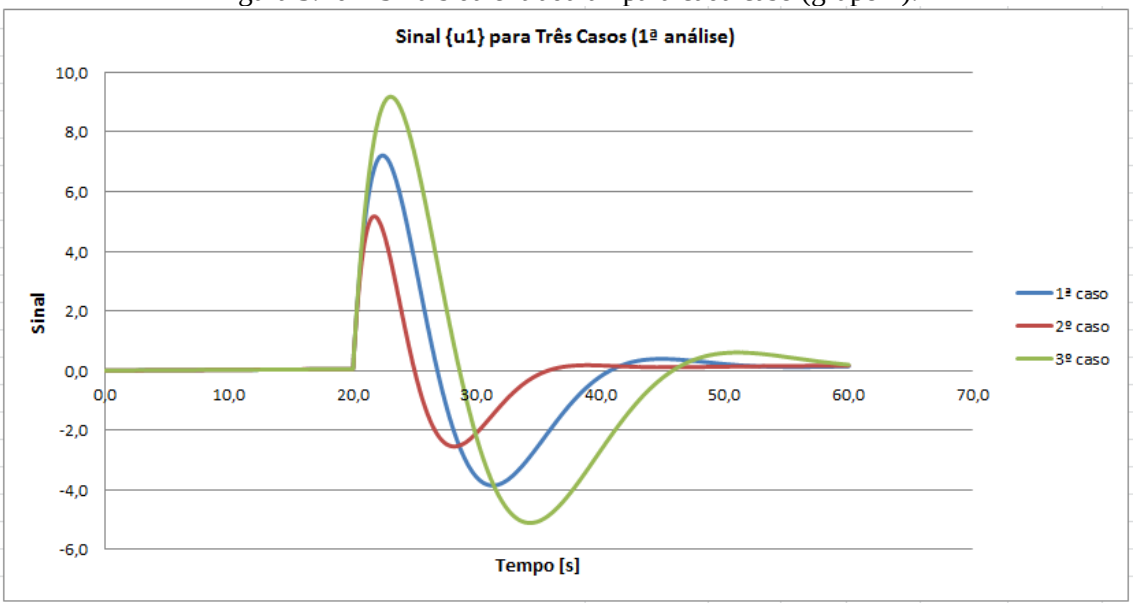
Figura 5.19 – Momentos principais de inércia versus dimensão c.



Fonte: Produção do autor.

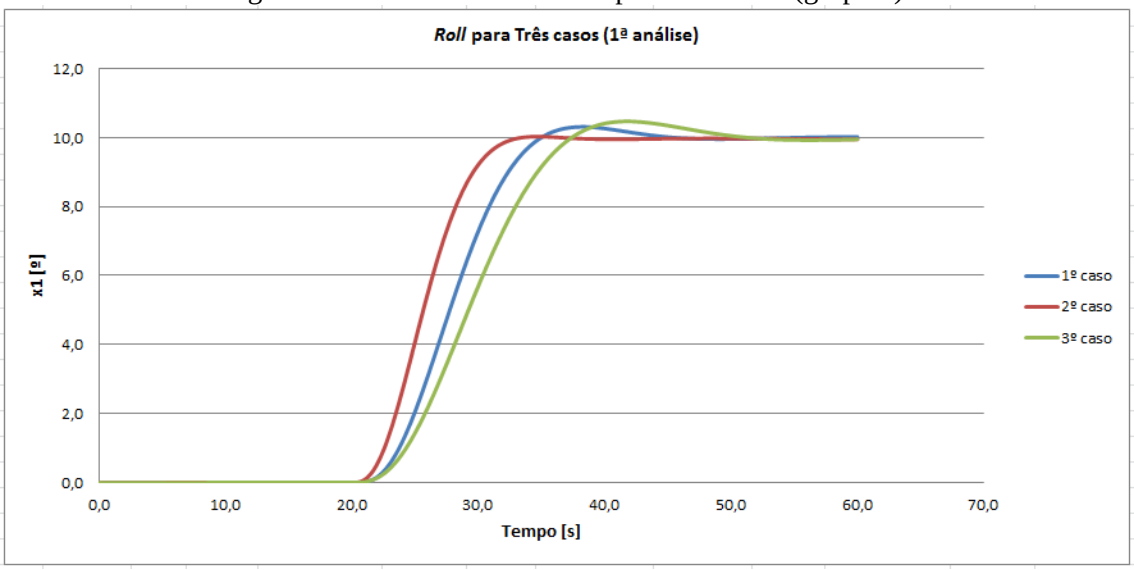
Comparando as Figuras 5.18/19 com os resultados de simulação correspondentes 5.20/21, constata-se que o melhor desempenho é obtido com a configuração do 2º caso – que é justamente aquele no qual os momentos de inércia principais se aproximam ao máximo. Tanto o tempo de estabilização quanto o sobressinal estão mais próximos do desejável. Esse fato não prova uma relação direta entre proximidade desses parâmetros (secundários) e melhoria de resposta do sistema, mas aponta para possíveis relações a serem consideradas no projeto.

Figura 5.20 – Sinais da entrada u1 para cada caso (grupo 1).



Fonte: Produção do autor.

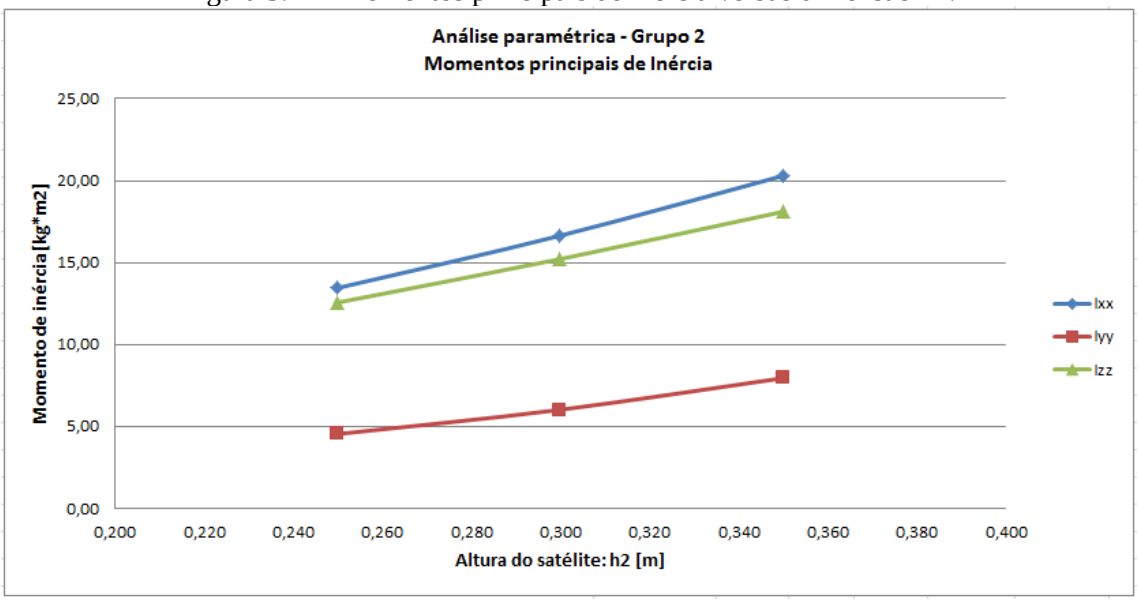
Figura 5.21– Sinais do estado x1 para cada caso (grupo 1).



Fonte: Produção do autor.

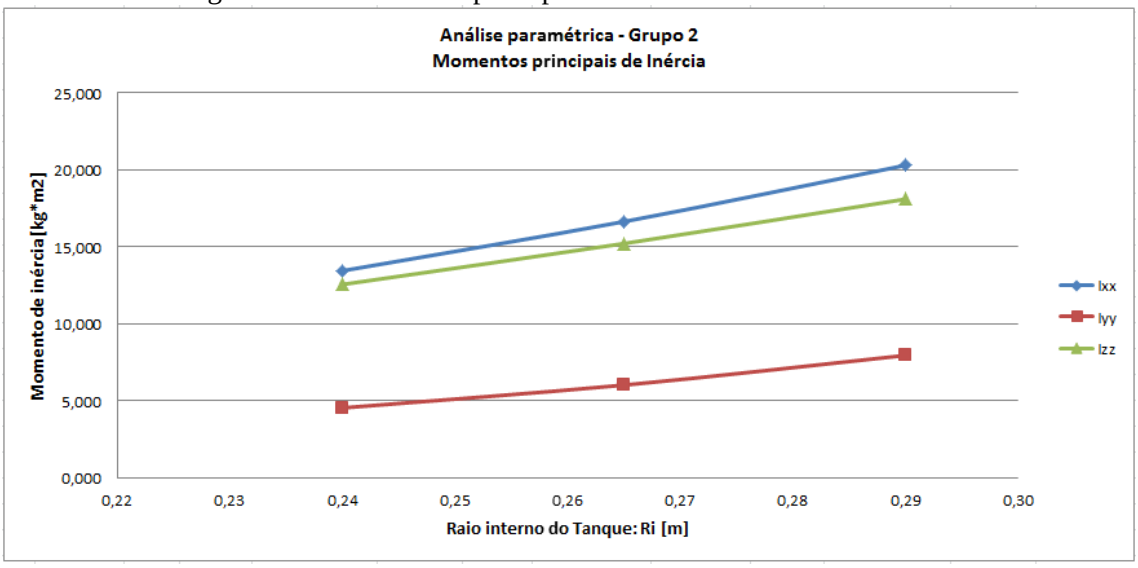
Os três casos do segundo grupo (Tabela 5.2) revelam uma relação entre parâmetros de entrada e secundários diferentes daquela do primeiro grupo: enquanto neste houve uma relação não-linear – difícil de ser descrita analiticamente – no segundo a relação aparenta ser linear – ao menos para a faixa de variação escolhida.

Figura 5.22– Momentos principais de inércia versus dimensão h2.



Fonte: Produção do autor.

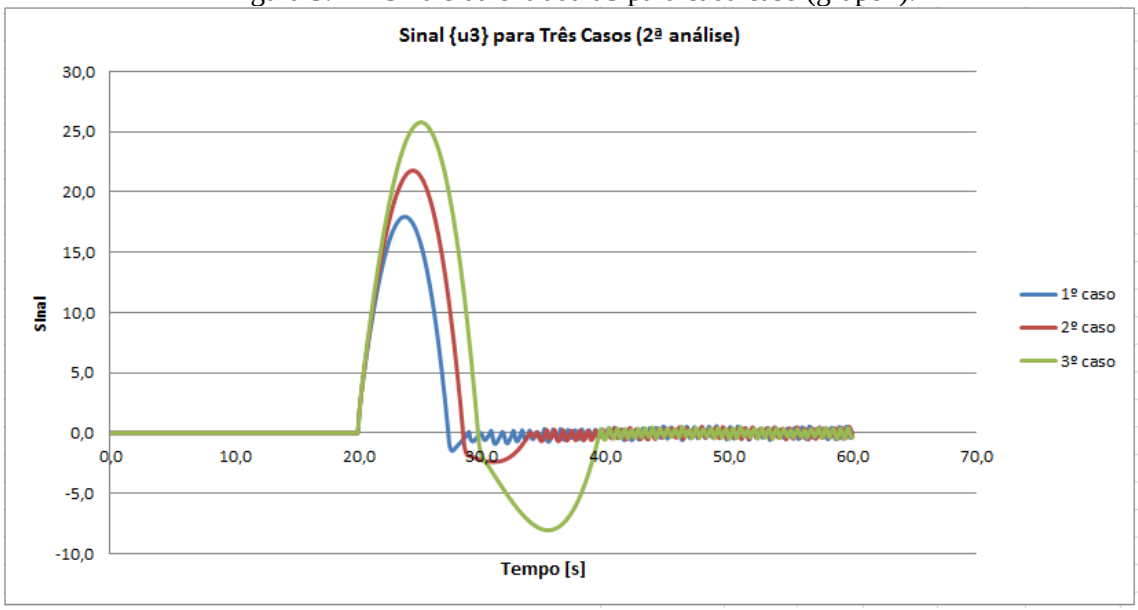
Figura 5.23– Momentos principais de inércia versus dimensão Ri.



Fonte: Produção do autor.

Para esse grupo, foi experimentada uma **manobra de *yaw* de 20º em sentido positivo**.

Figura 5.24– Sinais da entrada u3 para cada caso (grupo2).



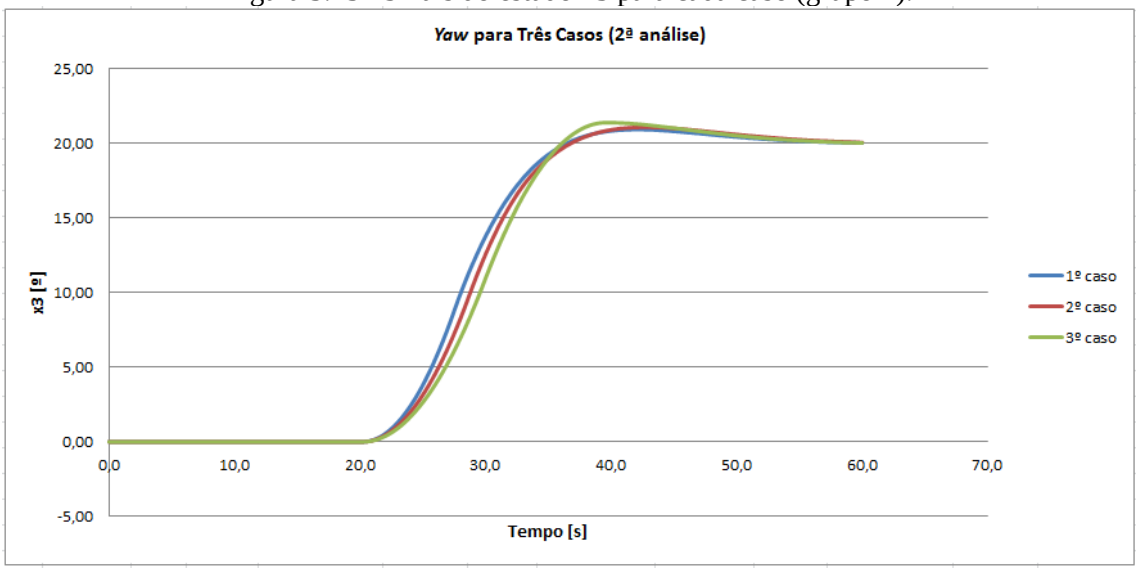
Fonte: Produção do autor.

Percebeu-se que os casos com menor sobressinal são o de número 1 e 2. O tempo de estabilização é aproximadamente igual nos três casos, conforme atesta a Figura 5.25. Quanto ao sinal de entrada – equivalente à força magnética de acionamento da válvula – constatou-se que o 1º caso apresenta menor magnitude máxima, conforme revela a Figura 5.24.

O empuxo – conforme esperado ao ver a Figura 5.24 – tem menor tempo de aplicação no caso 1, o que revela que a configuração de parâmetros nessa situação é mais adequada a obter maior eficiência da manobra (não do propulsor).

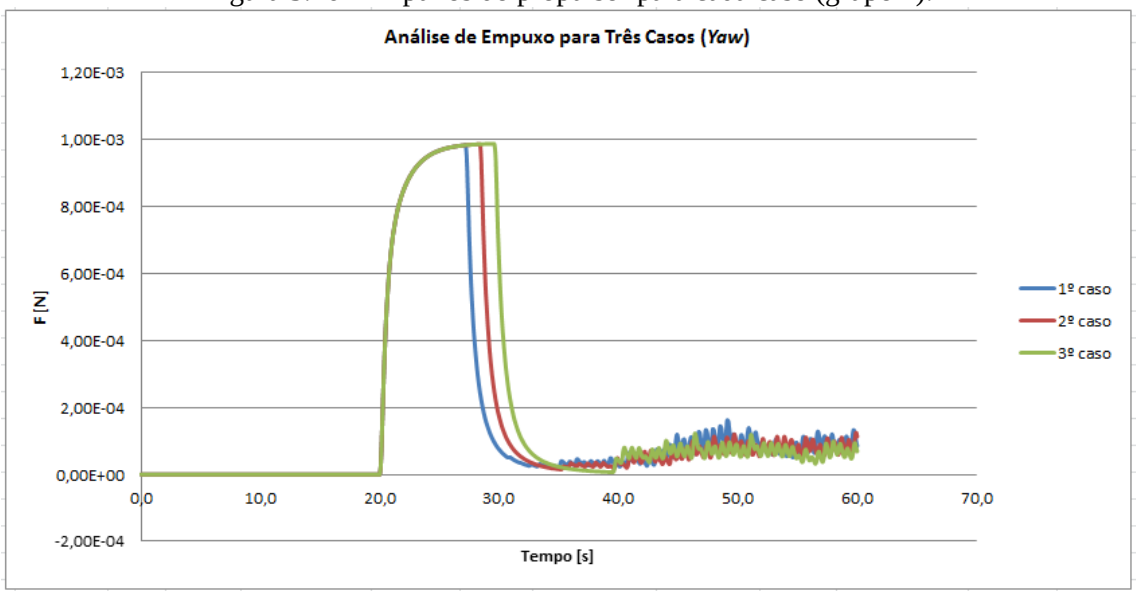
Com essas observações, pode-se perceber a importância da análise paramétrica no desenvolvimento de um sistema. Automatizar e executar remotamente diversos casos de teste pode representar uma agilização e diminuição de custos sem precedentes.

Figura 5.25– Sinais do estado x3 para cada caso (grupo 2).



Fonte: Produção do autor.

Figura 5.26– Empuxos do propulsor para cada caso (grupo 2).

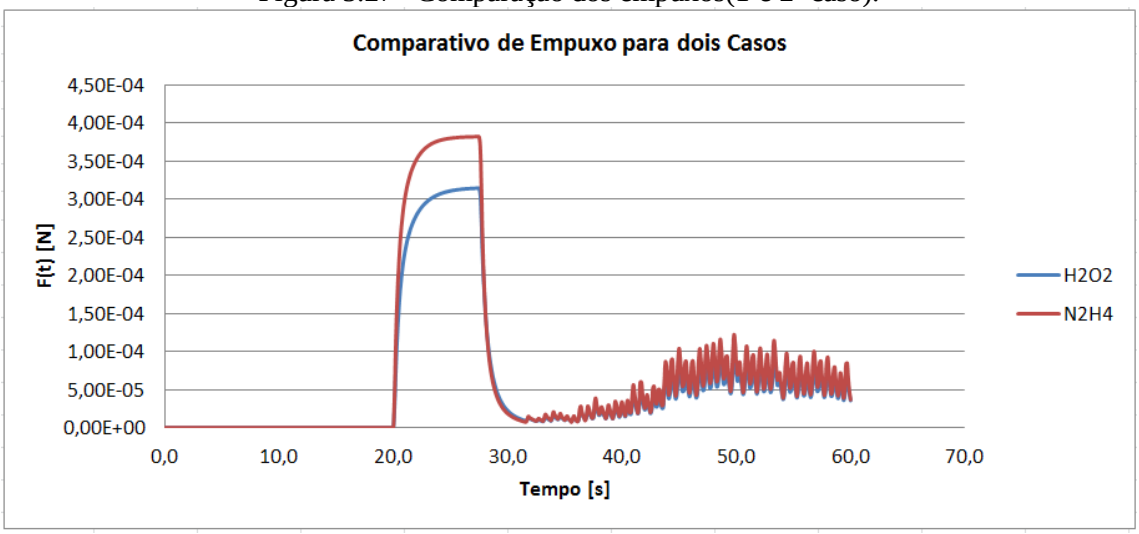


Fonte: Produção do autor.

O próximo estudo consistiu em comparar o desempenho do propulsor a partir de dois combustíveis diferentes. Nesse caso, o interesse era perceber como as propriedades de cada conjunto de produtos do propelente acabavam por afetar o impulso específico – que, apesar de não ser adimensional, é considerado um parâmetro de eficiência no ramo da propulsão.

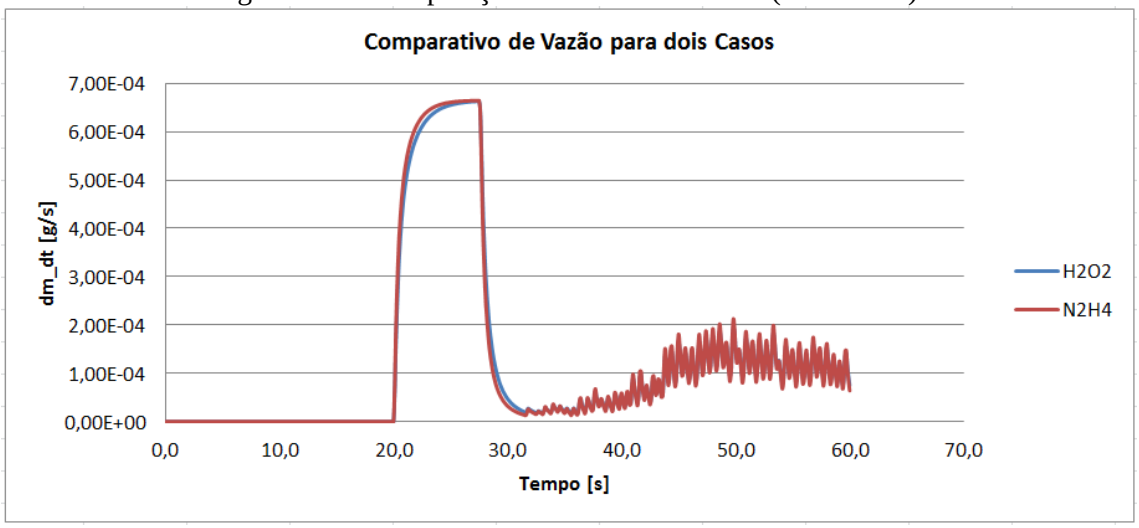
Para uma manobra de *yaw* de 20º, considerando os parâmetros de entrada adotados inicialmente (ver resultados dos processos 1 e 2), foram obtidas duas curvas de empuxo, explicitadas na Figura 5.27.

Figura 5.27– Comparação dos empuxos(1ºe 2º caso).



Fonte: Produção do autor.

Figura 5.28– Comparação das vazões mássicas (1ºe 2º caso).



Fonte: Produção do autor.

Aplicando as relações 5.1 e 5.2 obteve-se o valor do impulso específico de cada caso e seu ideal (Tabela 5.4).

Tabela 5.4 – Comparação entre impulsos específicos de cada caso e entre seus respectivos ideais.

Valores de Isp [s]		
	H2O2	N2H4
Ideal (teórico)	208,8	291,2
Simulado	54,3	65,9
Percentual	26,0%	22,6%

Fonte: Produção do autor.

As informações obtidas apontam para uma maior eficácia do $\boldsymbol{H_2O_2}$, que, com um empuxo menor, é capaz de fornecer a potência necessária para a execução da mesma manobra.

A contribuição desse trabalho consistiu, portanto, em elaborar uma metodologia, requisitos de processo e uma diagramação dinâmica, através da elaboração de duas cadeias de processos. Essa construção permite o desenvolvimento de sistemas complexos de forma mais eficaz, criando uma gramática comum (diagrama iner-ferramentas dinâmico) que cria sinergia a partir do manuseio das interconexões, dentro de transformações possíveis.

6 O caso de estudo visou explicitar a potencialidade dessa construção. Ele pode ser usado em diversos tipos de desenvolvimentos, sendo um elemento que aumenta a eficiência do processo de desenvolvimento.

6 CONCLUSÕES

O presente trabalho buscou **esboçar uma série de requisitos de ambiente** colaborativo, ressaltando aspectos críticos no desenvolvimento de projetos. Esses requisitos balizam o estabelecimento de interconexões em termos visíveis (**requisitos de conexão**) e mais sutis, relacionados: à simulações (**requisitos de temporização**); à ordem de execução (**requisitos de sequenciamento**), sem o qual é possível obter-se resultados confiáveis e coerentes; e de retroalimentações (**requisitos de automação**), que estabelecem o número de iterações.

Viu-se que o modo de conexão, de sequenciamento, de temporização e de automação influencia na eficiência e sucesso da cadeia. Dessa forma é desejável que o desenvolvimento de sistemas através de práticas colaborativas tenha um delineamento claro de requisitos de ambiente. Exercitou-se a capacidade de sistematizar a integração e interconexão de ferramentas através de um diagrama inter-ferramentas dinâmico, o que pode sustentar a adoção de uma política de reuso eficaz nas instituições.

Para desenvolver as cadeias propostas foi necessário superar alguns obstáculos – que conduziram à **criação de ferramentas auxiliares**, dando uma maior maleabilidade aos processos. O escopo das cadeias é **unificar as diversas especialidades nas fases iniciais de um projeto**. Os casos desenvolvidos poderiam ser ampliados, envolvendo outras disciplinas como planilha de custos (ferramenta financeira), gradiente de temperatura (incremento de condições ambientais), análise de casos específicos (colisões, falha de equipamentos), entre outros. Ter-se-á uma rede cada vez mais completa, cuja complexidade de elaboração passa pela necessidade de delinear o que se pode e não se pode; o que se deve e não se deve; o que implica certas conexões; quais as possibilidades e outros problemas (**requisitos de ambiente**).

Seria possível **mensurar ganhos de tempo e recursos** nas Fases 0 e A de um sistema construindo-se uma cadeia de processos colaborativos de um projeto já elaborado. O objetivo nesse caso seria quantificar quantas horas (por exemplo) foram usadas para os funcionários desenvolverem de modo convencional – e quantas foram utilizadas para desenvolvimento colaborativo. Ter-se-ia uma evidência da eficácia de se trabalhar com ambientes colaborativos.

Tabela 6.1 – Eixos centrais e secundários e seus respectivos avanços ao longo da Tese.

GÊNESE DE TÓPICOS DE PESQUISA E AVANÇOS AO LONGO DO TRABALHO			>>> GRAU DE IMERSÃO NO TEMA >>>				
			1	2	3	4	5
EIXOS DA TESE	CENTRAIS	Diagrama de Relações e Transformações através de equivalências			x		
		Criação de ferramentas auxiliares para integração de cadeias de processo				x	
		Listagem e elaboração de esboço de Requisitos de Ambiente (colaborativo)		x			
		Criação de Diagrama de Processos			x		
		Criação de Diagrama de Classificação de Ferramentas		x			
		Execução de Cadeias de Processo multidisciplinares, multiparadigma e multiplataforma				x	
	SECUNDÁRIOS	Implementação da simbologia de Grafos de Ligação a sistemas físicos				x	
		Tabelação de diversos conceitos de micropropulsão		x			
		Possibilidade de realizar validação via análise dimensional e semelhança	x				
		Análise de sensibilidade automatizada: implementação e diferenciais			x		

Fonte: Produção do autor.

A Tabela 6.1 destaca os tópicos centrais e secundários abordados ao longo desta Tese, destacando o nível de aprofundamento de cada um. Foi criado um grau de imersão nos temas de 1 a 5: (1) sugestão; (2) iniciação; (3) desenvolvimento; (4) aprofundamento e; (5) consolidação. A Tabela 6.1 revela que todos os temas centrais da OGU foram ou iniciados, ou desenvolvidos ou aprofundados. Adicionalmente, temas secundários – porém importantes para conduzir o estudo e mostrar sua aplicação – também foram tocados e trabalhados até certo grau. O único eixo central de desenvolvimento colaborativo que ficou à nível de sugestão foi a questão da validação via análise dimensional (não prometida), que será discorrida como tópico a ser iniciado em trabalhos futuros.

A análise paramétrica automatizada, apesar de não ser o eixo central, se relaciona diretamente aos processos remotos, pois estes potenciam a velocidade (geração de informações) de desenvolvimento de sistemas multidomínio, multiparadigma, multiplataformas (MMM). Resta aprofundar esses estudos, quantificando ganhos em relação aos métodos convencionais.

Conforme foi evidenciado no Capítulo 2, a complexidade é um parâmetro cada vez mais relevante no processo de desenvolvimento de sistemas. Sendo assim, é salutar para as

instituições que desenvolvem sistemas de grandes implicações revisarem a forma de organizarem seus processos – especialmente nas etapas inciais de desenvolvimento, em que mudanças são relativamente fáceis e o poder de economia é grande.

7 SUGESTÕES PARA TRABALHOS FUTUROS

Muitas outras ferramentas podem ser fundidas ao processo de desenvolvimento. Um projeto completo e integrado pode requisitar ferramentas que modelem geometria (CAD), manufatura de componentes (CNC), fluidodinâmica (CFD), análise de tensões (CAE), falhas de dispositivos, entre outros. O grande mérito de ambientes como o RCE reside na *simplicidade de integração* e *capacidade de coordenação*. Esta última viabiliza a difusão de informações relevantes, dando liberdade a todos os nós perceberem como os outros subsistemas afetam seus próprios – e vice-versa. Ao mesmo tempo, não há nenhuma interferência nas particularidades de desenvolvimento (metodologia) e na representação (modelamento) dos fenômenos e dispositivos; e nos métodos numéricos (parâmetros de simulação) dos componentes. Trata-se de uma rede descentralizada cuja coordenação é realizada por uma máquina.

O **diagrama de relações dinâmico inter-modelos** é uma simbologia de grande utilidade que permite mapear ferramentas e assim construir da melhor forma uma cadeia de processo. Quanto maior o número e diversidade de ferramentas, e mais completo o mapeamento das mesmas, passa-se a ter menos esforço para desenvolver a partir do zero. Somando a isso a disponibilidade de ambientes colaborativos, ter-se-á condições plenas de desenvolver sistemas sem recorrer à construção de modelos. Pois na colaboração, não é necessário a um nó (grupo / instituto) possuir todas informações, saberes e ferramentas – é preciso que os participantes, em conjunto, possuam o necessário (diagramas e licenças), se comuniquem e trabalhem (ambiente colaborativo) da forma mais proveitosa possível.

Há também muito a ser feito em termos de análise paramétrica. Com a automação de processos multidisciplinares, é possível realizar diversos estudos, com geração de grande quantidade de dados em pouco tempo, observando possibilidades de projeto dentro dos limites (requisitos) estabelecidos e estabelecendo relações entre parâmetros de entrada e calculados – ou entre aqueles e variáveis.

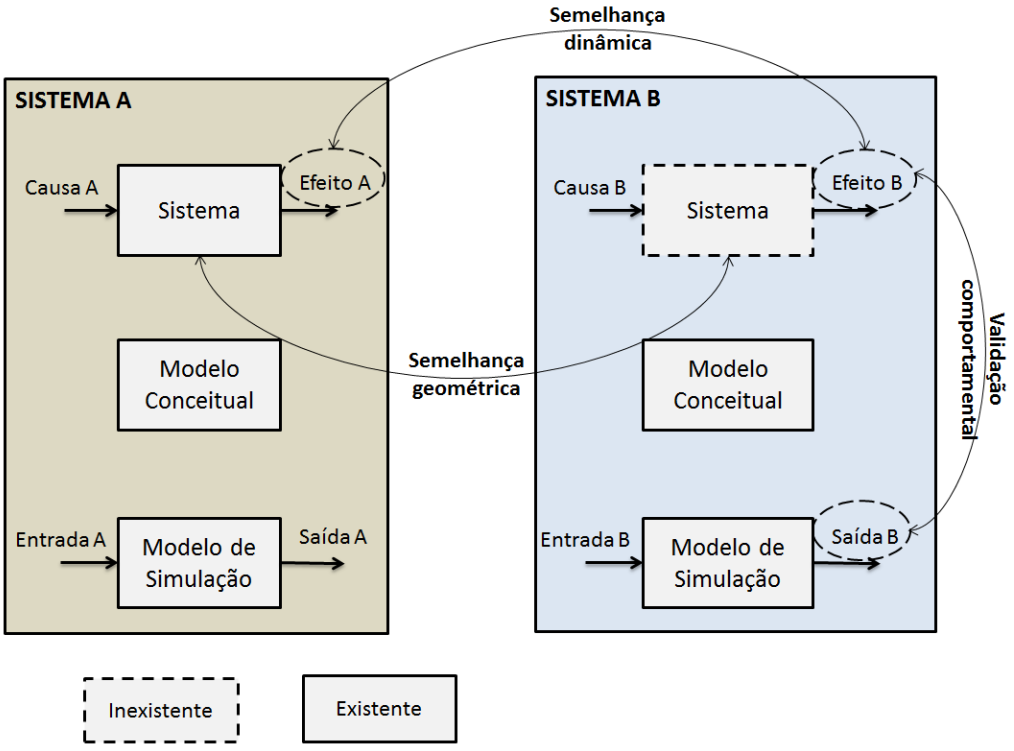
Outra ideia a ser desenvolvida futuramente é a “validação indireta”, através do uso da Análise Dimensional e da Teoria da Similaridade. A primeira é um método que visa reduzir o número e a complexidade das variáveis físicas envolvidas na análise de um

determinado fenômeno físico. A segunda visa estabelecer uma relação (estática=geométrica, cinemática=temporal, ou dinâmica) entre um modelo físico (escala reduzida) e um protótipo (replica do sistema real).

O objetivo de resgatar os conceitos da Análise Dimensional e da Teoria da Similaridade é sistematizar uma possibilidade de validação sem a existência do sistema real para serem executados testes, baseado em conceitos de Similaridade entre componentes e partes de subsistemas.

A Figura 7.1 revela uma ideia que poderia ser desenvolvida futuramente. Nele estabelece-se a possibilidade de realizar a validação comportamental de um sistema ainda inexistente.

Figura 7.1 – Conceito de validação via Análise Dimensional e Teoria da Similaridade.



Fonte: Produção do autor.

O Sistema B está sendo projetado. Existem modelos conceituais e de simulação do mesmo. Deseja-se validar os mesmos antes do projeto. Isso não é possível usando os conhecimentos de validação – que transitam entre as fronteiras modelo conceitual,

modelo computacional e sistema real de um mesmo sistema. No entanto, caso exista um sistema (Sistema A) com características fundamentais parecidas ao que se pretende construir, pode-se estabelecer uma semelhança estática, cinemática e dinâmica entre as estruturas dos sistemas (A, existente e B, inexistente) e suas saídas.

Em suma, as possibilidades de aprofundar o tema são variadas, e com isso fomentar a criação de metodologias de desenvolvimento diferentes, que visem focar em etapas mais adiantadas do desenvolvimento. Assim ter-se-á condições de reorientar os recursos humanos das instituições para questões mais centrais, que demandam criatividade.

REFERÊNCIAS BIBLIOGRÁFICAS

ALEXANDER, I. S.; STEVENS, R**. Writing better requirements**. [S.l.]: Pearson Education, 2002.

BANDECCHI, M.; MELTON, B.; ONGARO, F. Concurrent engineering applied to space mission assessment and design. **ESA Bulletin**, v.99, Sept. 1999.

BAJAJ, M.; ZWEMER, D.; YNTEMA, R. MBSE++ foundations for extendedmodel-based systems engineering acrosssystem lifecycle. In: ANNUAL INCOSE INTERNATIONAL SYMPOSIUM, 26., 2016, Edinburgh, Scotland. **Proceedings…** 2016.

BEARDEN, D. A. Small satellite costs. **Crosslink**, p.33-44, Winter 2001.

BHAVEL, A.; GARLAN, D.; KROGH, B. Augmenting software architectures with physical components. In: EMBEDDED REAL TIME SOFTWAREAND SYSTEMS CONFERENCE, 2010, Toulouse, France. **Proceedings…** 2010.

BEHERE, S. **A systems engineering approach to design of complex systems.**Kungliga Tekniska Hogskolan, Estocolmo, 22 June 2015.Disponível em: https://www.kth.se/polopoly_fs/1.576015!/6.22_2_Systems_Engineering_SagarBehere%2BMartinTorngren.pdf

BOOCH, G.; BROWN, A.W. **Collaborative development environments**. Rational Software Corporation, 28 October, 2002. Disponível em: https://www.researchgate.net/publication/220662520_Collaborative_Development_Environments.

BRITTO, R.M.; SOUZA, M.L.O.Management of modeling and simulation processes and data for space system applications. In: WORKSHOPEM ENGENHARIAE TECNOLOGIAS ESPACIAIS, 8., São José dos Campos. **Anais...** São José dos Campos: INPE, 2017.

BUCHEN, E.; DEPASQUALE, D. **Nano / microssatellite market assessment**. Atlanta, GA: SpaceWorks Enterprises, 2014. Disponível em: http://www.sei.aero/eng/papers/uploads/archive/SpaceWorks_Nano_Microsatellite_Market_Assessment_January_2014.pdf. Acesso em: 18 jul. 2018.

CAPRA, F. **O ponto de mutação:** a ciência, a sociedade e a cultura emergente.[S.l.]: Cultrix, 1982.

CAPRA, F.; LUISI, P.L. **The systems view of life:** aunifying vision.Cambridge: Cambridge University Press, 2014.

CHAN, S. Complex adaptative systems. In: REASEARCH SEMINARIN ENGINEERING SYSTEMS, 83., 2001. **Proceedings…** 2001.

CHEN, A.; DI, L..; WEI, Y.; BAI, Y.; LIU, Y.An optimized grid-based, OGC standars compliant collaborative software system for servind NASA geospatial data.In: ANNUAL IEEE/NASA SOFTWARE ENGINEERING WORKSHOP, 30., 2006. **Proceedings…** 2006.

DESHMUKH, M.; SCHARTZ, R.; BRAUKHANE, A.; LÓPEZ, R. P.; GERNDT, A. Model linking to improve visibility and reusability of models during space system development**.** In: IEEE AEROSPACE CONFERENCE, 2014, Big Sky, MT. **Proceedings…** IEEE, 2014. Disponível em: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6836287.

ERICHSEN, H. **Spacecraft propulsion systems impulsive performance analysis** London: Computadorized Educational Platform, 2006.

FERREIRA, F. M. **Modelagem de sistemas mecânicos utilizando procedimentos modulares.**156p. Dissertação (Mestrado em Engenharia Mecânica) - Instituto Militar de Engenharia, Rio de Janeiro, 2006.

GOÉS, L.C.S. **Modelagem de sistemas dinâmicos:** transdutores: sensores e atuadores. São José dos Campos: Instituto Tecnológico de Aeronáutica, [S.d.].

HASKINS, C. et al.**Systems engineering handbook**:a guide for system life cycle processes and activities.[S.l.]: INCOSE, 2007.

HITT, D.; ZAKRZWSKI, C.; THOMAS, M. **MEMS-based satellite micropropulsion via catalyzed hydrogen peroxide decomposition.** Institute of PhysicsPublishing, 2001. Disponível em: http://stacks.iop.org/SMS/10/1163.

HULL, E.; KEN, J.; DICK, J. **Requirements engineering.** 2.ed. London: Springer, 2005. ISBN 1-85233-879-2.

MAIA, F.F. **Novo catalisador para decomposição de peróxido de hidrogênio em micropropulsores de satélites**. 2012. 123p. Disssertação (Mestrado em Combustão e Propulsão) - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2012. Disponível em: http://mtc-m16d.sid.inpe.br/col/sid.inpe.br/mtc-m19/2012/04.04.21.02/doc/publicacao.pdf.

MANELLI NETO, H. **Estudo de requisitos e especificações para a tolerância a falha simples do sistema de controle de atitude da plataforma multimissão**. 209p. Dissertação (Mestrado em Engenharia e Tecnologia Espaciais) - Instituto Nacional de Pesquiasas Espaciais, São José dos Campos, 2011.

MARIA, R. B. Management of modeling and simulation processes and data for space system applications. In: WORKSHOP DE ENGENHARIA E TECNOLOGIA ESPACIAIS, 8., 2017, São José dos Campos, SP. **Anais...** São José dos Campos: INPE, 2017.

MARTINS, A. V. B. **O uso da técnica dos grafos de ligação para a simulação de centrais hidrelétricas em regime transitório.** 160p. Dissertação (Mestrado em Engenharia da Energia) - Universidade Federal de Itajubá, Itajubá, 2004.

MASI, D.D. **A sociedade pós-industrial**. 4.ed. São Paulo: Senac, 2003.

MEADOWS, D. **Thinking in sytems:** a primer. [S.l.]: Chelsea Green Publishing, 2001.

NAGEL, B. et al. Communication in aircraft design: can we establish a common language? In: INTERNATIONAL CONGRESSOF THE AERONAUTICAL SCIENCES, 28., 2012. **Proceedings…** 2012.

NEELAMKAVIL, F. **Computer simulation and modelling.** Dublin, Ireland: John Wiley and Sons, 1987.

NERI, J.A.C.F. Microssatélites do INPE e o Programa Espacial Brasileiro. **Parcerias Estratégicas,** n.7, p.227-233, Out.1999.

OGATA, K**. Engenharia de controle moderno**. 5.ed. Rio de Janeiro: LTC, 2010.

OLIVA, L.L; SOUZA, M.L.O. An overview of multi-paradigm modelling and simulation. In: SAE INTERNATIONAL, 2015. São Paulo. **Anais...** 2015.

OLIVA, L. L. **Comparação da modelagem e simulação do subsistema propulsivo da PMM orientada por fluxos físicos e fluxos de informação.** Dissertação (Mestrado em Engenharia e Gerenciamento de Sistemas Espaciais). - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2012.

PINHEIRO, E.R; SOUZA, L.C.G. Projeto do sistema de controle de atitude de um microsatélite usando o LQG. In: WORKSHOPON SPACE ENGINEERINGAND TECHNOLOGY, 2012. **Proceedings…** 2012.

PROCHNOW, S. L. **Subsistemas de controle de atitude para miniaturização de satélites.** Santa Maria: Instituto Nacional de Pesquisas Espaciais, 2007. Relatório final de projeto de Iniciação Científica.

RAMOS, A.L. et al. Model-based systems engineering: an emerging approach for modern systems. **IEEE Transactionson Systems, Manand Cybernetics, Part C: Applicationsand Reviews**, v.42, n.1, p.101-111, 2012.

SCHAUS, V. et al. **Simulation and software technology collabortive development of a space system simulation model.** Disponível em: https://core.ac.uk/download/pdf/30995894.pdf.

SCHAUS, V. et al. **Automated sensitivity analysis in early space mission design**. In: INTERNATIONAL WORKSHOPON SYSTEMS& CONCURRENT ENGINEERINGFOR SPACE APPLICATIONS, 5., 2012. **Proceedings…** 2012.

SCHAUS, V. et. al. **Concurrent engineering software development at German Aerospace Center:** status and outlook. Berlin: German Aerospace Center (DLR), 2010.

SEIDER, D. et al.Ad hoc collaborative design with focus on iterative multidisciplinary process chain development applied to thermal management of spacecraft. In: CEAS AIRAND SPACE, 4., 2013. **Proceedings…** 2013.

SEIFERT, C.; STALLBAUM, S. **The impact of complexity costs on operations planning**. APICS, 2014. Disponível em: https://pt.slideshare.net/Wilson_Perumal/the-impact-of-complexity-costs-on-operations-planning-v5.

SIAHPUSH, A. **A brief survey of attitude control systems for small satellites using momentum concepts**. Glendale, Arizona: Satellite Systems Division, 1988.

SOUZA, P.N. **Curso introdutório em tecnologia de satélites:** CSE-200, unidade 2, parte 2.1, versão 3.0. São José dos Campos: INPE, 2009. Notas de aula.

SOUZA, A.C.C.; SILVA NETO, L.P.; OLIVA, L.L.; LOUREIRO, G. System concurrent engineering for the conception of an attitude and orbit control system. In: WORKSHOPDE ENGENHARIAE TECNOLOGIAS ESPACIAIS, 2., 2010, São José dos Campos. **Anais...** São José dos Campos: INPE, 2010.

VANGHELUE, H. **Multi-formalism modelling and simulation.** Gent, Belgica: Faculteit Wetenschappen, 2001.

VASCONCELLOS, M. J. E. **Pensamento sistêmico:** o novo paradigma da ciência. Campinas: Papirus, 2003.

WERTZ, J.R.; LARSON, W.J. **Space mission analysis and design**. 3.ed. [S.l.]: Microcosm, 1999.

YANG, C.-D.; SUN, Y.-P. Mixed H_2/H_∞ state-feedback design for microsatellite attitude control. **Control Engineering Practice**, v.10, n.9, p;951-970, Sept. 2002.

ZUR, S.; TROLTZSCH, A. Optimization of the DLR Space Liner inside the integration environment RCE. In: RODRIGUES, I. et al. (Ed). **Engineering optimization IV**. London: Taylor and Francis, 2015. p.757-761.

APÊNDICE A –PENSAMENTO SISTÊMICO

O pensamento sistêmico permeia a Engenharia de Sistemas (ES). Aqui tratar-se-á da dimensão conceptual, que forma a pedra basilar sobre a qual toda Teoria Geral de Sistemas (BERTALANFFY, 1937)se apoia para se auto elaborar, e que a ES usou (e usa) de base para sustentar e criar seus conceitos, métodos e ferramentas.

As principais características do pensamento sistêmico, segundo (CAPRA; LUISI, 2014), são:

- Se constrói das partes para o todo;
- Vai dos objetos aos relacionamentos;
- Da medição para o mapeamento;
- Migra do conhecimento objetivo para o contextual;
- Muda o foco da estrutura para o processo.

Conforme será visto adiante, podemos fazer uma divisão do mundo econômico-tecnológico em três fases: pré-industrial, industrial e pós-industrial (De MASI, 1999). De modo geral, também podemos destacar os três paradigmas dominantes na mentalidade humana ao longo de sua História – lembrando que estes não estão necessariamente em fase com os três anteriores – em termos cronológicos.

Da Antiguidade até o final da Idade Média (Séc. XV) predominou no imaginário das massas o paradigma religioso, caracterizado por crenças calcadas nos mitos, no senso comum e nos dogmas religiosos. Independente da cultura ser politeísta ou monoteísta, a essência era idêntica em termos de explicar os fenômenos naturais. Essa visão de mundo considera a divindade como o centro de tudo, e o conhecimento seria acessível a poucos (sacerdotes), responsáveis pela intermediação entre a(s) divindade(s) e o mundo. No entanto, é importante frisar que os avanços na ciência e na técnica foram significativas nessa época.Como exemplo pode-se citar a cúpula da Catedral de Florença, *Santa Maria del Fiore*, projetada por Filippo Brunelleschi (1377-1446) em 1419, continua sendo a maior cúpula de alvenaria já construída.

É amplamente aceito que a ciência foi o principal motor humano responsável pelo fim do domínio da mentalidade místico-religiosa e que ela, através de proeminentes

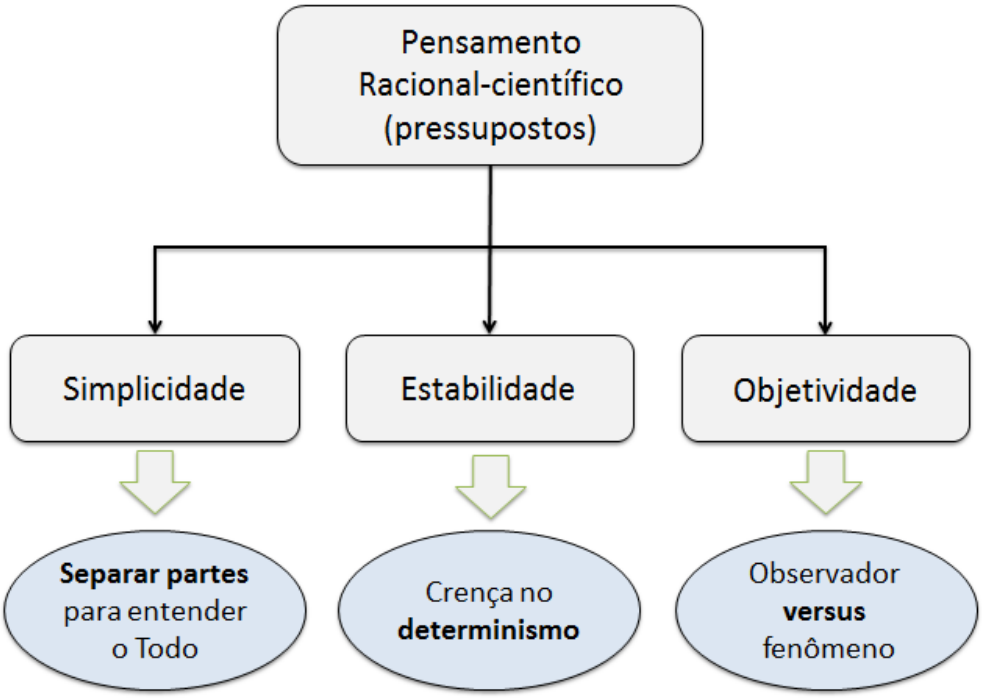
personalidades, iniciou um processo de pleno desenovimento a partir do século XVI e XVII. A respeito dessa transformação o proeminente físico norte-americanoFritjof Capra afirmou que:

> Entre 1500 e 1700 houve uma mudança drástica na maneira como as pessoas descreviam o mundo e em todo o seu modo de pensar. A nova mentalidade e a nova percepção do cosmo propiciaram à nossa civilização ocidental aqueles aspectos que são característicos da era moderna. Eles tornaram-se a base do paradigma que dominou a nossa cultura nos últimos trezentos anos e está agora prestes a mudar. (CAPRA, 1982, p. 27)

Dessa forma o pensamento racional-científico começa a predominar no Século XVI, início da Idade Moderna, época em que o Teocentrismo dá lugar ao Antropocentrismo. Passa-se da idade da fé para a idade da razão. Como consequência, a Ciência começa a se desenvolver de forma metódica e crescente, desde o domínio da Física e da Química, até o da Biologia, da Sociologia, da Economia e da Psicologia. É nessa época (Séc. XV – Séc. XIX) que surgem as maiores descobertas científicas, como a Teoria da Gravitação Universal, A Lei da Conservação de Massa, a Teoria da Evolução, as Leis da Termodinâmica, entre muitas outras.

Para atingir conquistas tão grandes, o paradigma racional-científico se apoia em três pressupostos: a simplicidade, uma visão estática e a objetividade (Figura A.1).

Figura A.1 – Pressupostos do pensamento racional-científico.



Fonte: Adaptado de Vasconcellos (2003).

O pressuposto da simplicidade parte do princípio de que para entender o todo (de qualquer entidade, seja um animal, um vegetal, um objeto, um veículo, uma cidade, o corpo humano, entre outros) é necessário separar as partes e estudá-las isoladamente. Isso provou ser de grande serventia para as ciências, em particular a Física. Mas apenas até certo ponto, pois a depender do sistema em estudo, tirar um de seus componentes do contexto para compreendê-lo – ou tentar – não trará nenhuma informação útil, e consequentemente nenhum conhecimento novo. Esse pressuposto impede que um mesmo objeto (ou uma entidade viva, ou uma ideia) pertença a duas categorias diferentes, o que, observando o mundo, sabe-se não ser verdade.

O pressuposto da visão estática parte do princípio de que o mundo, tal qual é, já está formado em todos seus níveis (físico, orgânico, psíquico, conceptual), o que colima na mentalidade de que o mundo não está em transformação substancial – apenas em aspectos superficiais, da forma. Isso também conduz ao pressuposto da previsibilidade, pois o que não pode ser previsto com segurança é associado a um conhecimento imperfeito, levando a um estreitamento ainda maior do conhecimento.

O pressuposto da objetividade leva o observador a “separar-se” do fenômeno para analisa-lo. Procura discriminar o objetivo do ilusório (suas próprias opiniões ou subjetividade). Daí advém a crença de que o universo e o que ocorre nele independe de quem o observa. Assim, as observações conjuntas e reproduzíveis que coincidem com outras observações independentes se reforçam, adquirindo caráter de confiáveis e objetivas.

Os três pressupostos serviram de base para o desenvolvimento da ciência moderna, permitindo o desenvolvimento de métodos e ferramentas para descobrir fenômenos, compilar conhecimentos, estabelecer leis e construir aparatos baseado nessas leis. No entanto, com o advento da Teoria da Relatividade e da Mecânica Quântica (Séc. XX) essa trindade basilar começa a não dar conta de explicar satisfatoriamente novos conceitos e descobertas, sendo incapaz de trabalhar com uma realidade nascente. Por exemplo, o Princípio da Incerteza de Heisenberg, que impõe restrições à precisão com que se podem efetuar medições simultâneas de uma classe de pares de observáveis em nível subatômico – contrariando o pressuposto da objetividade; ou o conceito de espaço-tempo curvo de Einstein, que estabelece vínculos entre essas dimensões e a força gravitacional.

Tendo em vista essa incompletude do paradigma racional-científico, (VASCONCELLOS, 2003) elaborou uma classificação (Tabela A.1) que define as limitações do paradigma hodierno, exemplificando-as.

Tabela A.1 – Limitações dos pressupostos do pensamento racional-científico.

DIFICULDADE DE CADA PRESSUPOSTO EM LIDAR COM AS CIÊNCIAS		Ciências		
		Físicas	**Biológicas**	**Humanas**
Pressupostos	**Simplicidade**	Fácil	Difícil	Difícil
	Estabilidade	Fácil	Muito difícil	Difícil
	Objetividade	Fácil	Fácil	Muito difícil

Fonte: Vasconcellos (2003).

Usando os pressupostos do pensamento racional-científico, podemos abordar sem dificuldades o mundo físico e químico, com suas grandezas fundamentais, matéria e energia, e suas relações. Nesse campo a formalidade matemática se destaca, sendo uma ferramenta de grande importância para descobrir e explicar fenômenos naturais, além de possibilitar o desenvolvimento de novas tecnologias.

Quando adentramos no mundo das ciências biológicas surge o conceito de vida. E com isso aparecem as dificuldades em simplificar fenômenos e (especialmente) em tratar ela – a vida – como algo estável. Não é verdade que todos os sistemas naturais busquem o equilíbrio a todo custo (homeostase). Sabe-se que a principal causa da evolução biológica das espécies é justamente essa capacidade de atingir estados de desequilíbrio, que possibilita a emergência de novas características. Caso a homeostase fosse a regra da vida, não haveria no mundo nada além de amebas – seres unicelulares que existem há bilhões de anos, justamente por serem os melhores adaptados a mudanças bruscas do ambiente.

Quando ascendemos para a área de ciências humanas torna-se evidente a incapacidade dos pressupostos clássicos da ciência para lidar com o aspecto psicológico do ser humano e seu comportamento coletivo (sociedade), com suas instituições (política, justiça, economia, entre outras). A objetividade raramente se aplica a uma avaliação que envolva fatores psíquicos, sociais ou políticos. Domina a subjetividade – pautada por opiniões individuais, às vezes muito distintas, do universo interno do indivíduo em relação ao mundo externo. Os métodos e ferramentas dos quais as ciências naturais fizeram farto uso – e tiveram enorme sucesso – começam a revelar suas limitações.

Para lidar com as questões apresentadas surge o pensamento sistêmico, apoiado no paradigma da complexidade, que contrapõe os três pressupostos com outros três (Figura A.2).

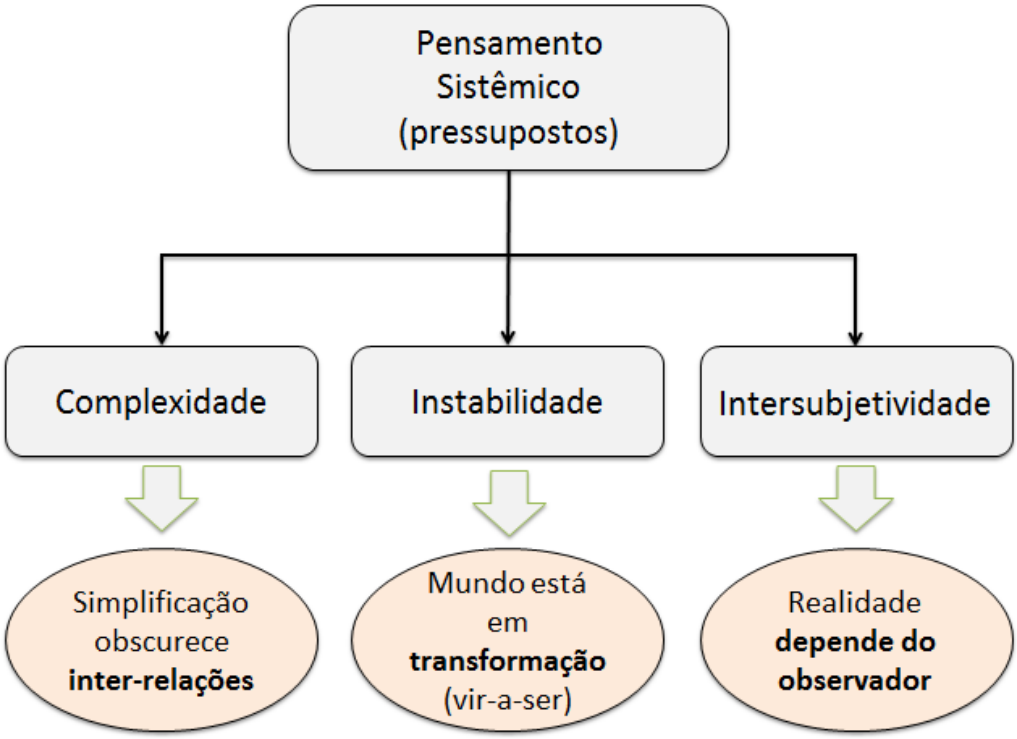
O primeiro deles afirma a **complexidade** do universo, indicando que a simplificação obscurece as inter-relações – a compreensão e construção delas. A sincronicidade é um exemplo de uma descrição baseada nesse pressuposto. Trata-se de um conceito desenvolvido por (JUNG, C.G. 1929) para definir acontecimentos que se relacional por

relação de significado, não causal. O termo sincronicidade é referido por Jung como "coincidência significativa"

Pode-se observar a relevância desse conceito nas leis naturais, que só se sustentam para magnitudes macro físicas, sendo, portanto verdades estatísticas. No mundo microfísico essas leis não mais se aplicam. Isso implica num princípio de justificação diverso do causal, levando à abertura do pensamento da acausalidade – podendo ser tudo que não é definido pela causalidade.

O segundo deles reconhece que o mundo está num processo de transformação (vir-a-ser), levando ao enfoque baseado na indeterminação, irreversibilidade e incontrolabilidade dos fenômenos – ao menos em sua totalidade. Ele afirma a **instabilidade** no (e do) Universo. Exemplo desse pressuposto está na homeostase psíquica e na abordagem orgânica.

Figura A.2 – Pressupostos do pensamento sistêmico.



Fonte: Adaptado de Vasconcellos (2003).

Finalmente, para superar a objetividade, insere-se a intersubjetividade, que afirma que observador e fenômeno estão fundidos, sendo um dependente do outro. O conhecimento científico é uma construção social, em espaços consensuais, guiada pela liberdade (opiniões, história do grupo, circunstâncias históricas). O cientista trabalha com múltiplas versões da realidade, fazendo uso de diversos domínios.

É importante introduzir os principais conceitos de Ciência de Sistemas (CS), uma vez que a Engenharia de Sistemas pode aprender com os sistemas naturais, cujas características impressionam o homem desde tempos imemoriais. Seja pela sua resiliência (ex: gramíneas do bioma savânico do cerrado), pela sua capacidade de auto-organização (ex: flocos de neve de Koch) ou pela sua estrutura hierárquica (ex: abelhas e formigas), os sistemas naturais servem de inspiração para a criação de novas tecnologias e processos.

Antes de avançar é preciso definir o que é um Sistema e quais seus principais constituintes. Segundo (MEADOWS, 2001),“Um Sistema é um conjunto de coisas – pessoas, células, moléculas ou o que quer que seja – interconectadas de tal modo que essas coisas produzem seu próprio padrão de comportamento ao longo do tempo.”

“Ou de modo mais sintético: um Sistema é um conjunto de elementos interconectados que possuem um objetivo comum.” Essa definição tem a vantagem de destacar justamente os três constituintes que caracterizam qualquer sistema: elementos, interconexões e propósito.

Podemos relacionar elementos com partes, sejam seres vivos ou não-vivos; as interconexões com fluxos físicos ou de informação que transitam entre e dentro dos elementos; e o propósito com a razão de existir do sistema – tenha ele consciência ou não disso.

Para esses conceitos ficarem mais claros foi elaborada uma lista (Tabela A.2) explicitando essas três entidades (elementos, interconexões e propósito) para três sistemas de natureza distinta.

Tabela A.2 – Exemplificação de elementos, interconexões e propósitos de três sistemas distintos: social (escola), biológico (árvore) e físico-tecnológico (satélite).

EXEMPLO DE SISTEMAS		ATRIBUTOS DE SISTEMAS		
		ELEMENTOS (UNIDADES)	INTERCONEXÕES (RELAÇÕES)	PROPÓSITO (TELEFINALISMO)
Base --->		Técnica	Sistêmica	Místico-Filosófica
SISTEMAS DE DIFERENTES NATUREZAS	Escola	Alunos, professores, instalações, carteiras, mesas, livros, cadernos, laboratórios.	Requisitos para admissão, notas mínimas, fluxos monetários, fofoca, troca de conhecimento, reuniões, palestras.	Formar cidadãos conscientes e instruídos.
	Árvore	Folhas, galhos, raízes, frutos.	Metabolismo da árvore (reações químicas e fluxos físicos). Ex: perda de água em dias quentes ou solo seco.	Sobrevivência. Reprodução. Evolução.
	Satélite	Painéis solares, propulsores, estrutura, hardware, baterias, cabos.	Comandos para subsistemas (fluxos de informação), trocas de calor (fluxos físicos).	Mapear a Terra, transmitir informações, observar os astros.

Fonte: Produção do autor.

A partir dessa definição exemplificada pode-se começar a formar uma ideia dos desafios futuros para a engenharia. Desenvolvemos de modo intenso uma inteligência em nível de componente (elementos), desenvolvendo-os. Mas as interconexões, que foram surgindo à medida que o progresso foi se desdobrando no século XX, trouxeram em seu bojo novos problemas, que exigem muito mais uma compreensão e adequada representação das interconexões do que da melhoria dos elementos isoladamente. Podemos exemplificar essa questão com um aeroporto. Percebe-se o quanto a tecnologia aeronáutica se desenvolveu no último século (aerodinâmica, materiais, controle, aviônica, propulsão, dentre outras). Por outro lado, a organização dos horários de voos, alocação de passageiros e cargas, ordenamento eficiente das pessoas, entre outros, não acompanhou o progresso tecnológico em nível de elemento (avião). Trata-se de um problema sistêmico, em que as interconexões são os elementos decisivos que alavancam a melhoria dos sistemas.

Num nível mais profundo temos o propósito de um sistema. Trata-se do objetivo que este deve cumprir. Das diretrizes que irão nortear a dinâmica de suas interconexões e da evolução de seus elementos. Nesse ponto chega-se a um nível filosófico-religioso,

pautado pela ética. Esse aspecto – mesmo que não trabalhado de forma profissional – pode enriquecer o estudo de sistemas (ciência) e o desenvolvimento dos mesmos (engenharia).

APÊNDICE B – MODELOS, ABORDAGENS E DOMÍNIOS FÍSICOS

MODELOS

A modelagem é um processo tão antigo quanto a própria civilização, justamente porque ela nasce da (e com a) escrita – modelos linguísticos. A partir desse ponto o ato de representar o mundo, as ideias, os desejos e os projetos vão se sofisticando, até chegar ao ponto em que o grau de sofisticação e consequências possíveis se elevou a tamanho grau, que se iniciaram estudos para sistematizar o ato de representar.

A modelagem é o ato de representar um sistema (real ou virtual) apoiado num paradigma, com um determinado método e/ou ferramenta. Um modelo jamais irá representar um sistema com todas suas nuanças. Isso já indica que essa atividade estará limitada (fidelidade), e que esta limitação pode ser grande (modelo muito simples) ou pequena (modelo elaborado) em relação ao objeto de estudo / projeto.

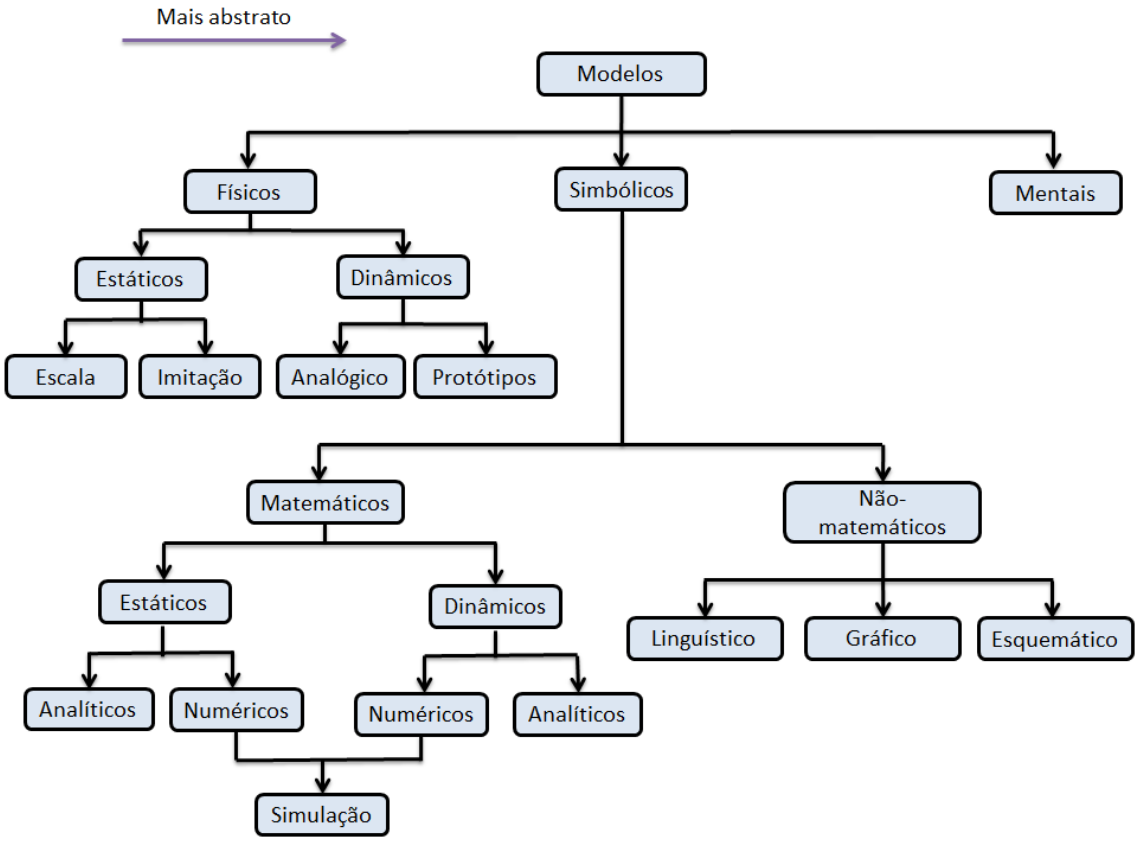
A Figura B.1 revela todos os tipos de modelos existentes (NEELAMKAVIL, 1987).

A primeira subdivisão é entre físicos e abstratos – que se subdividem em simbólicos e mentais.

Modelos físicos só podem representar sistemas físicos (mecânicos, elétricos, térmicos, hidráulicos, entre outros). Logo, eles devem possuir componentes tangíveis. Esses modelos são descritos por variáveis mensuráveis como pressão, voltagem, velocidade, posição, temperatura, fluxo, força, entre outras. Eles podem representar entidades que não evoluem no tempo (estáticos) ou que evoluem no tempo (dinâmicos).

Modelos físicos estáticos se subdividem em dois tipos: de escala, cujo exemplo clássico são maquetes de construções (estátuas de cera, de pontes, de navios, de casas, de navios,...); e de imitação, que podem ser exemplificados por bonecos, estrutura molecular, desenhos animados, divisões geológicas e políticas num mapa, etc. A diferença entre ambos reside no fato de que os primeiros visam representar da forma mais fiel possível a entidade em questão, ao passo que os últimos são aproximações pobres, que pode ser caricatural, feitas com esse propósito.

Figura B.1 – Classificação dos modelos.



Fonte: Adaptado de Neelamkavil (1987).

Modelos físicos dinâmicos também se subdividem em dois tipos: analógico, na qual um sistema físico é usado para compreender outro, com fenômenos análogos entre si (ex: circuito indutivo-capacitivo-resistivo, RLC, para modelar o sistema de suspensão de um carro, termômetro de mercúrio, fluxo hídrico para estudar fluxo de dinheiro, de carros); e protótipos, no qual monta-se um sistema físico tal como é o sistema a ser projetado e/ou estudado (ex: circuito RLC para estudar oscilações elétricas, associação massa-mola-amortecedor para compreender amplitudes de vibração e forças de veículos).

Quase todos os modelos simbólicos são mais econômicos e rápidos de serem construídos comparados a modelos físicos – as exceções são para sistemas ou componentes extremamente simples. Seu uso é grande, e vem crescendo cada vez mais no mundo devido à sua praticidade e poder de abstração. Nessa categoria podemos fazer

uma divisão entre modelos matemáticos e não-matemáticos. Vejamos o que caracteriza cada um deles.

Modelo matemático é um conjunto de relações lógicas e equacionadas entre elementos de um sistema – e entre sistemas. Uma teoria expressa na linguagem matemática é um modelo matemático. Apesar de poderoso em descrever fenômenos físicos de todos os domínios, químicos e mesmo alguns biológicos, esse tipo de abordagem não é capaz de descrever fielmente atitudes, valores, etc – tão importantes para as pessoas. Nesse ponto podemos perceber o limite representativo desse tipo de modelo e estabelecer limitações para seus métodos e ferramentas. A partir da esfera orgânica, passando pelo humano e suas instituições, sente-se a limitação matemática em explicar satisfatoriamente o universo vivo e consciente circunjacente.

Da mesma forma que no caso anterior, podemos subdividir os modelos matemáticos em estáticos (descritos por equações algébricas) e dinâmicos (descritos por equações diferenciais, no mundo contínuo, ou de diferenças, no mundo discreto). Ambos por sua vez se subdividem de forma igual em dois tipos: analíticos e numéricos.

Modelos analíticos são expressões definidas, sem aproximações, ao passo que modelos numéricos são – como o nome indica – aproximações numéricas de expressões definidas por modelos analíticos.

A existência de modelos numéricos revela a necessidade de reproduzir modelos analíticos no tempo, de forma rápida e precisa, de forma a gerar um ou múltiplos gráficos, relacionando variáveis diversas, para estudos e melhoramentos posteriores. Quando inserimos essas aproximações numéricas numa determinada linguagem computacional, temos um modelo de simulação.

Dentro da representação simbólica, modelos não matemáticos são todos aqueles que não fazem uso de expressões lógicas ou equações para descrever algo. Nesse caso podem ser elencados três grandes grupos: modelos linguísticos, gráficos e esquemáticos. Os modelos linguísticos tratam da descrição de uma entidade ou narrativa em forma de linguagem escrita; os modelos gráficos funcionam na base da percepção visual – pinturas, desenhos artísticos – sem recorrer à uma norma específica ou à escrita;

modelos esquemáticos são desenhos que visam mostrar um processo (diagrama de produção), linha de raciocínio (algoritmo), fluxo de entidades num sistema, entre outros.

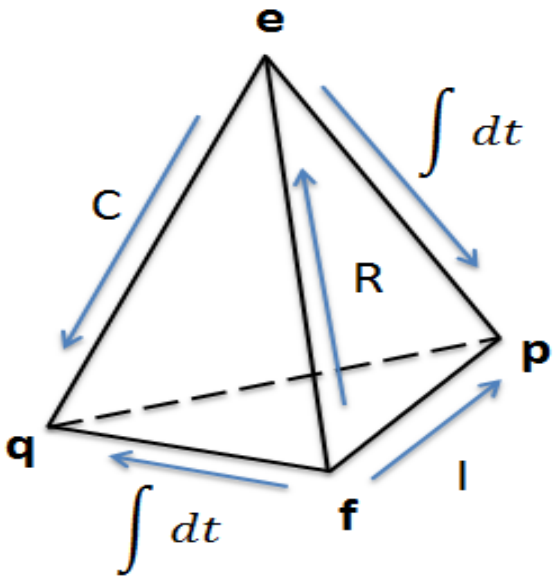
Os modelos mentais são a categoria mais abstrata. Eles são concebidos apenas mentalmente, e não são expressos fisicamente nem simbolicamente – o que torna o uso deles para a ESBM inviável.

ABORDAGENS E DOMÍNIOS FÍSICOS

Quanto ao aspecto de paradigma, pode-se observar outra questão importante. Dentro dos modelos (matemático-dinâmicos) de simulação, existe um detalhe importante a ser levado em conta quando se está interessado em executá-los numa ferramenta: os tipos de fluxos (abordagem), que tanto podem ser de informação (grandeza imaterial) quanto de massa, energia, ou grandezas relacionadas (grandezas materiais) determinarão os resultados de saída, que podem variar significativamente dependendo do tipo de abordagem adotado (OLIVA, 2012). Portanto torna-se importante a compreensão da representação dos fluxos.

O princípio que rege a abordagem física é baseado em fluxos físicos, o que se traduz por matéria e/ou energia em suas diversas formas. Podem-se exprimir esses fluxos através da notação de Grafos de Ligação (BG's - *Bond Graphs*). Esse tipo de notação é capaz de sintetizar todos os domínios físicos – mecânico, térmico, hidráulico, pneumático, elétrico, magnético – fazendo uso de analogias entre grandezas. Assim exclui-se a necessidade de uma notação específica para cada fenômeno. Tomemos como exemplo os amortecedores (fenômeno mecânico) e os resistores (fenômeno elétrico), cuja função é idêntica: dissipar energia. Podemos sintetizar esses dois fenômenos – substancialmente idênticos – usando apenas um símbolo para representar as variáveis, o que torna o trabalho de representação mais fácil, e consequentemente menos sujeito a ambiguidades. Essa transformação pode ser obtida através do tetraedro de estados generalizado (Figura B.2).

Figura B.2 – Tetraedro de estados generalizado.



Fonte: Adaptado de Painter(1959).

A Figura B.2 revela as quatro variáveis gerais, representadas pelos vértices, que representam variáveis físicas de qualquer domínio; e a relação entre elas, dadas pelas arestas. As variáveis são:

- Variável de esforço (e):

  Relacionam-se a estoque. São variáveis que mensuram uma grandeza circunscrita às fronteiras do estoque de um sistema.

- Variável de fluxo (f):

  Relaciona-se a fluxos. São variáveis que fluem através dos estoques de um sistema. Estão relacionadas diretamente à variável tempo, uma vez que fluir através implica em passagem de tempo.

- Variável de momento (p):

  É a integração da variável esforço (e) num dado intervalo de tempo.

- Variável de deslocamento (q):

É a integração da variável fluxo (f) num dado intervalo de tempo, o que resulta num acúmulo de fluxo no intervalo em questão – a variação dessa grandeza para mais ou para menos, no estoque.

A Tabela B.1 exemplifica as quatro variáveis de estado generalizadas para os diversos domínios físicos existentes.

Escolhendo um domínio específico para a análise fica fácil relacionar as grandezas físicas do mesmo com o conceito apresentado. Imagine a representação de um acumulador (domínio hidráulico). Trata-se de um componente que contém um líquido sobre certa pressão. Definindo o estoque como a pressão que o gás exerce sobre o fluído dentro do acumulador, chega-se: à variável de fluxo (apenas de saída), a vazão de saída do líquido; a variável de deslocamento, o volume de líquido (que saiu do estoque); e a variável de momento, a pressão atuando ao longo do tempo no líquido. A partir daí pode-se montar um modelo físico do tipo analógico (ver diagrama na Figura B.3) para estudar o sistema hidráulico em questão. Isso pode ser útil por diversos motivos, dentre os quais:

- Não há instrumentos para medição de pressão ou vazão;
- Não há um laboratório adequado para realizar experimentos nesse domínio;
- Os técnicos que devem preparar, executar e coletar os dados do experimento não possuem conhecimento do domínio.

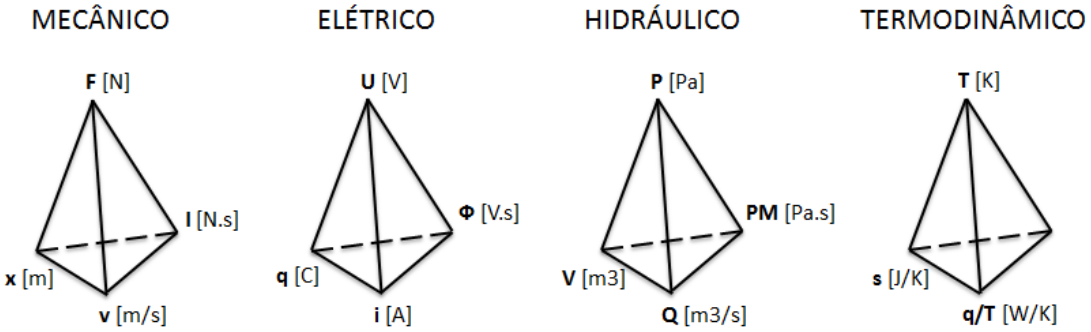
Tabela B.1 - Variáveis de potência e energia para diferentes domínios físicos.

		Variáveis de potência		Variáveis de energia	
		Esforço (e)	**Fluxo (f)**	**Momento (p)**	**Deslocamento (q)**
Domínio Físico	**Mecânico (translação)**	Força [N]	Velocidade [m/s]	Impulso [N.s]	Posição [m]
	Mecânico (rotação)	Momento [N.m]	Velocidade angular [rad/s]	Momento angular [N.m.s]	Ângulo [rad]
	Elétrico	Tensão [V]	Corrente [A]	Fluxo [V.s]	Carga [C]
	Magnético	Força magneto-motriz [A]	Taxa de fluxo [Wb/s]	---	Fluxo [Wb]
	Hidráulico	Pressão [Pa]	Vazão [m3/s]	Momento de pressão [Pa.s]	Volume [m3]
	Termodinâmico	Temperatura [K]	Fluxo de entropia [W/K]	---	Entropia [J/K]
	Químico	Afinidade [J/mole]	Fluxo molar [mole/s]	----	Progresso da reação [mole]

Fonte: Adaptado de Goés (2004).

Pode-se ainda inserir outros domínios na Tabela B.1, como o químico, da difusão e das reações, com suas variáveis características. Logo, chega-se a uma primeira síntese no domínio da ciência física e química, em que é possível representar diversas variáveis, de fenômenos diversos, usando um único símbolo, contanto que esta seja substancialmente idêntica em todos eles. Ou seja, é necessário apenas que haja um fio condutor – uma identidade intrínseca – entre as variáveis para que o processo de modelagem e simulação seja simplificado através da redução do número de variáveis envolvida no trabalho de representação de um sistema multi-domínio. Dessa forma é possível construir e trabalhar com modelos de domínios diferentes usando uma mesma simbologia, caso não haja ferramentas computacionais para representar determinado domínio, ou o profissional deseje fazer uma simulação breve – mas não esteja familiarizado com o fenômeno. Tem-se assim diversos tetraedros de estados (Figura B.3).

Figura B.3 – Exemplos de tetraedros de estados em domínios físicos distintos.

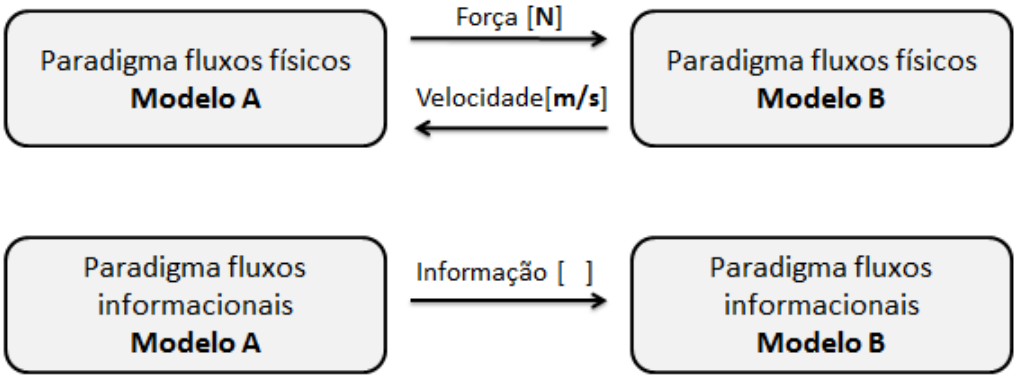


Fonte: Produção do autor.

As implicações da representação generalizada são importantes. Basta pensar no seguinte caso: se existe um projeto de engenharia no qual um sistema (ou às vezes componente) que sofre influências eletromecânicas, eletromagnéticas, hidráulicas e termodinâmicas (ex: propulsor espacial bipropolente), ao invés de trabalharmos com 16 variáveis, trabalha-se com 4, que são justamente as variáveis generais (esforço, fluxo, momento, deslocamento). Dessa forma qualquer problema que envolva n domínios físicos ou químicos pode ser reduzido a um problema de 1 domínio geral, substancial, que sintetiza a multiplicidade fenomênica numa variável única. Num conceito único.

Quanto a abordagem informacional, percebe-se que ela é de natureza diferente, pois não considera a causalidade. O seja, o que transita nesse tipo de abordagem é informação, que em sistemas físicos / de controle não obedece à lei causa-efeito. Nesse paradigma o fluxo no sentido Modelo A → Modelo B não gera uma reação Modelo B → Modelo A (Figura B.4). A informação flui num único sentido. No caso da abordagem por fluxos físicos, a lei causa-efeito é representada. Porque o que está fluindo entre elementos é uma grandeza física, seja matéria ou energia nas suas diversas formas. Isso implica em reações.

Figura B.4 – Princípios da abordagem física e informacional.



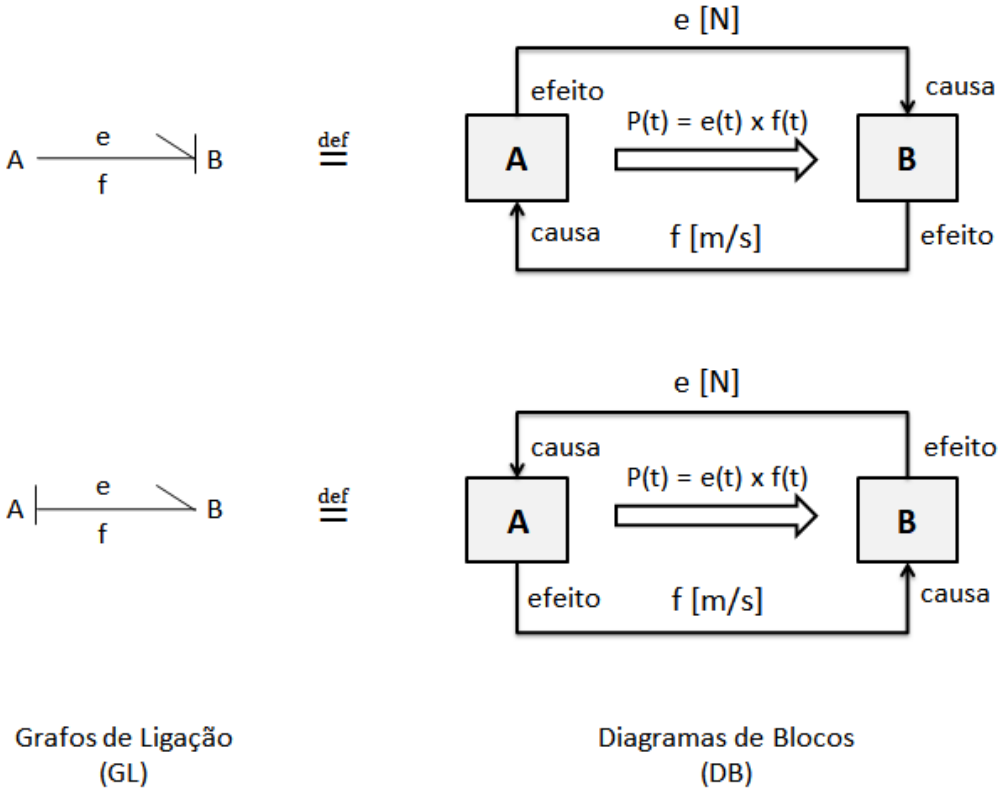
Fonte: Produção do autor.

Pode-se tomar como exemplo uma força impressa num dado corpo, representado por um modelo B. Esta irá causar uma aceleração nesse corpo, fazendo com que seu vetor velocidade se altere, levando a uma alteração de velocidade relativa de B em relação a A. Essa diferença entre velocidades afeta diretamente a força que A imprime em B, criando uma lógica natural de realimentação. Se se deseja manter a mesma força aplicada em B, deve-se imperiosamente alterar o vetor força. Esse fenômeno não seria representado instantaneamente na abordagem informacional – seria necessário um ciclo completo para termos uma nova configuração entre A e B.

Um diagrama mais pormenorizado (Figura B.5) permite estabelecer a relação entre a notação de fluxos físicos (BG) e a de fluxos informacionais, em Diagrama de Blocos (DB). Na notação BG usa-se meia seta para designar o fluxo de potência entre dois elementos/sistemas. A barra na parte esquerda indica o sentido das variáveis de esforço e fluxo: quando se encontra ao lado de B o esforço é causa em B; quando se encontra ao lado de A o esforço é causa em A.

As variáveis de potência são esforço e fluxo. O modelo A causa um esforço e (força) no modelo B, que por sua vez, como efeito, causa um fluxo f (velocidade) no modelo A, fechando um ciclo de retroalimentação.

Figura B.5 – Equivalência entre modelos esquemáticos GL / DB.



Fonte: Adaptado de Martins (2004).

A potência que flui instantaneamente entre os elementos A e B é dada pelo produto das variáveis $\boldsymbol{e(t)}$ e $\boldsymbol{f(t)}$ – daí o nome "variáveis de potência".

As variáveis de energia são, por definição, a integral das variáveis de potência, e são denominadas de momento ($\boldsymbol{p}$) e deslocamento ($\boldsymbol{q}$).

$$\boldsymbol{p(t)} = \int \boldsymbol{e(t)dt}$$

$$\boldsymbol{q(t)} = \int \boldsymbol{f(t)dt}$$

As ligações entre as variáveis gerais (arestas do tetraedro) são dadas por relações matemáticas. Essas relações possuem um equivalente físico, caracterizando um tipo de elemento, podendo ser um componente ou fenômeno, conforme mostrado abaixo.

- Elementos de armazenamento: inertância ($\boldsymbol{tipo - I}$)

  Armazenam energia, relacionando a variável de fluxo com a integral da variável momento. Isso ocorre quando o elemento tem capacidade de armazenar energia na forma de energia cinética. Logo, $\boldsymbol{f = f(p)}$.

- Elementos de armazenamento: capcitância ($\boldsymbol{tipo - C}$)

  Armazenam energia, relacionando a variável de esforço com a integral da variável deslocamento. Isso se dá quando o elemento tem capacidade de acumular energia em sua forma potencial. Logo, $\boldsymbol{e = e(q)}$. Um exemplo clássico é a mola, cuja deformação (variável q) se relaciona com a força se tração/compressão (variável e) a qual está submetida; e o capacitor, cuja carga (q) se relaciona com a tensão entre suas placas (e).

- Elementos dissipativos ($\boldsymbol{tipo - R}$)

  São elementos que dissipam energia, gerando calor (ex: resistência de chuveiro elétrico; atrito viscoso entre superfícies de rolamentos, mancais; atrito mecânico entre superfícies; geração de entropia num sistema). É dada pela relação entre as variáveis de potência na seguinte forma:

$$\boldsymbol{e(t) = \varphi_R \cdot f(t)}$$

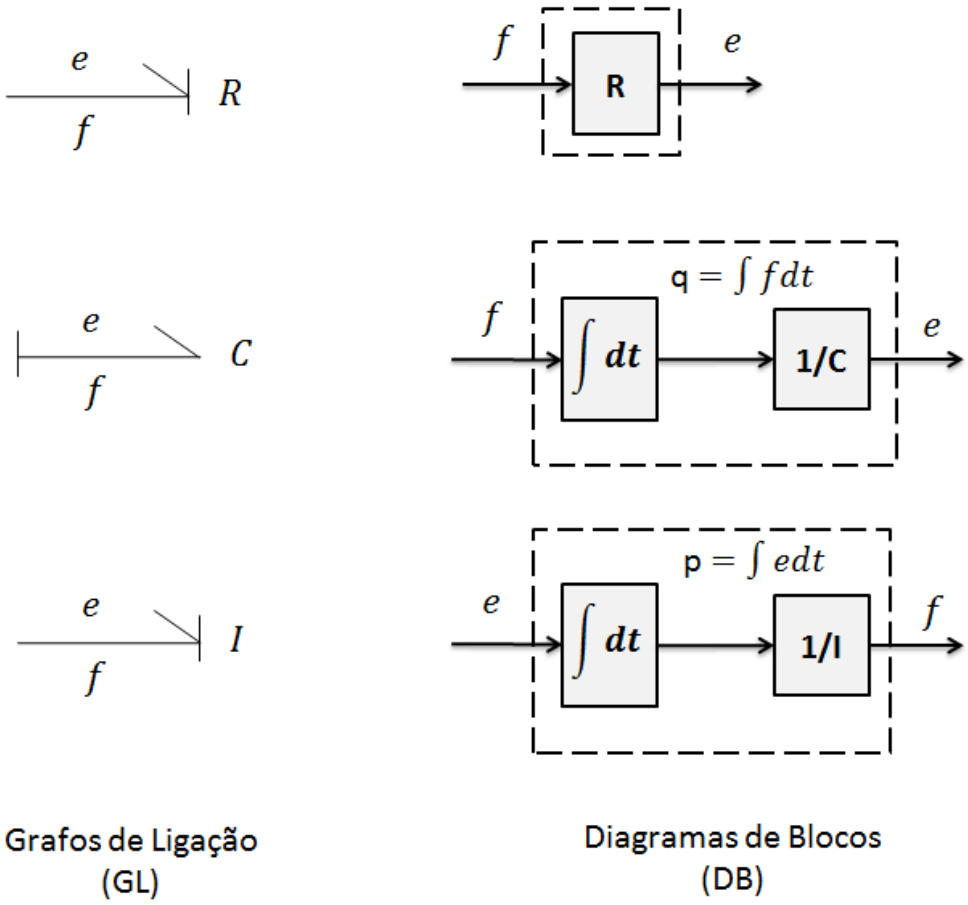
  Em que $\boldsymbol{\varphi_R}$ é uma função .

  Caso a relação seja linear, $\boldsymbol{\varphi_R = R}$ (constante) e a relação entre as variáveis de potência passa a ser linear.

$$\boldsymbol{e(t) = R \cdot f(t)}$$

Os elementos armazenadores (I,C) e dissipativo (R) podem ser simbolizados por DB ou GL. A relação entre as simbologias (Figura B.6) revela a equivalência entre a representação por fluxos físicos (potência) e informacional.

Figura B.6 – Relação entre simbologia DB e GL para elementos armazenadores (de energia potencial ou cinética) e dissipadores (de energia).



Fonte: Adaptado de Martins (2004).

Existem ainda elementos que operam outros tipos de fenômenos físicos. São eles os transdutores, que podem ser transformadores (TF) ou giradores (GY).

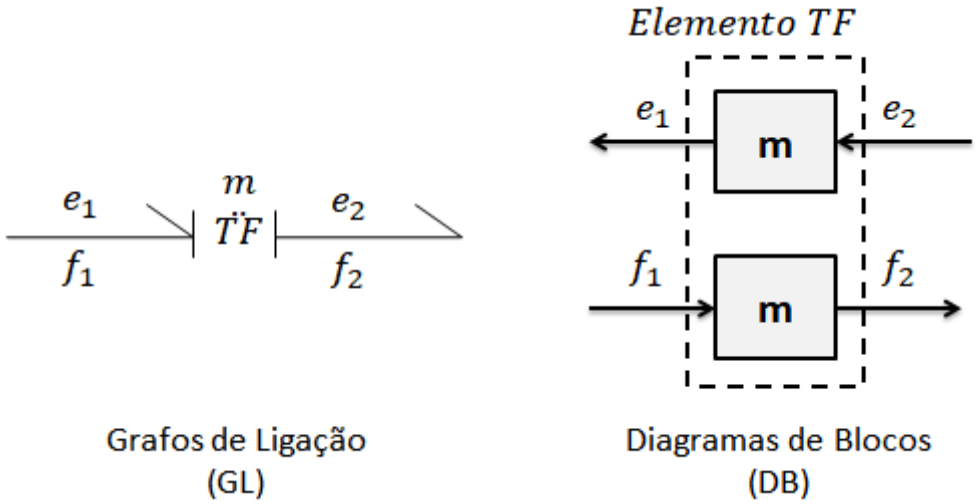
- Transformadores (***TF***)

  Realizam a conversão de energia entre diferentes domínios físicos. Sua lei constitutiva é dada pela seguinte relação:

$$\boldsymbol{e_1 = m \cdot e_2}$$
$$\boldsymbol{m \cdot f_2 = f_1}$$

A relação entre o modelo simbólico GL e DB das equações acima pode ser visualizado na Figura B.7.

Figura B.7 – Relação entre notação de transformadores (GL e DB).



Fonte: Adaptado de Martins (2004).

Exemplos de transformadores diretos: alavancas de comando, caixas de engrenagens, pistão hidráulico, transformadores de potência.

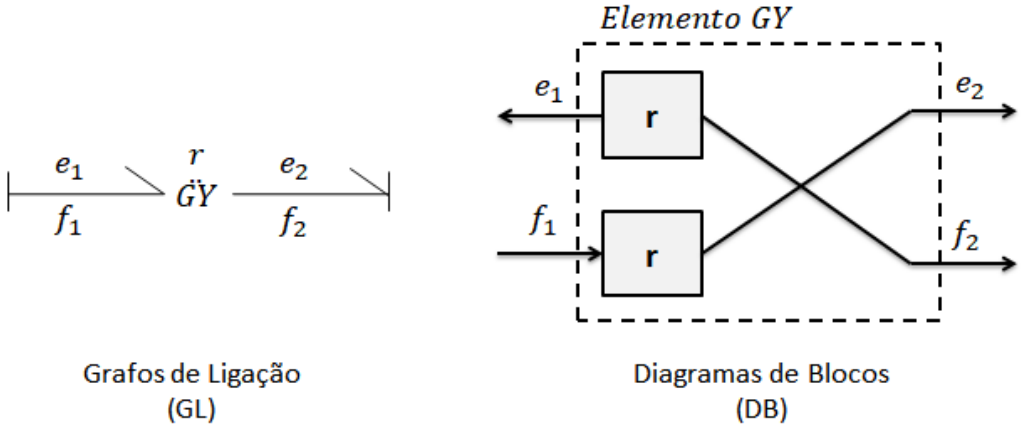
- Giradores (***GY***)

  Funcionam como um transformador inverso. Converte as variáveis esforço de entrada nas variáveis fluxo de saída; e as variáveis fluxo de entrada nas variáveis esforço de saída (Figura B.8).

  A relação constitutiva entre as variáveis é dada por

$$\boldsymbol{e_1 = r \cdot f_2}$$
$$\boldsymbol{r \cdot f_1 = e_2}$$

Figura B.8 – Relação entre notação de giradores (GL e DB).



Fonte: Adaptado de Martins (2004).

Todas as relações descritas podem ser organizadas conforme a Tabela B.2.

Tabela B.2 – Elementos físicos que relacionam variáveis generalizadas.

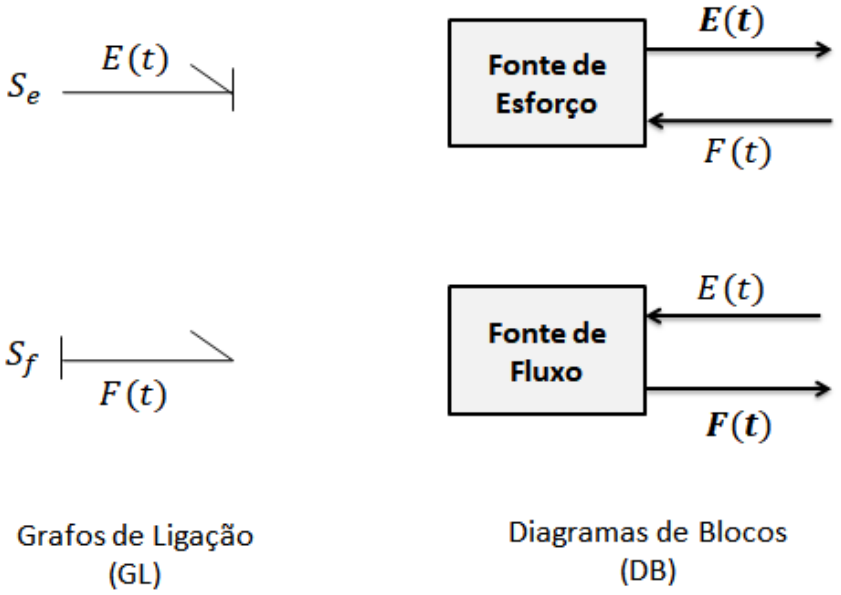
Elemento	**Símbolo**	**Descrição**	**Exemplos**
Armazenamento (tipo-I ou tipo-C)	I ou C	Armazenam energia, funcionando como estoques. Se subdividem em dois: (a) armazenam o fluxo (energia cinética); (b) armazenam o estoque (energia potencial).	Tipo-C: capacitores e molas. Tipo-I: indutores e massas.
Dissipativo (tipo-R)	R	Dissipam energia na forma de calor. Relação entre as variáveis de potência.	Tipo-R: amortecedores, resistores, atrito entre objetos.
Transdutor (TF ou GY)	TF ou GY	Realizam a conversão de energia entre diferentes domínios físicos.	TF: caixa de engrenagens, pistão hidráulico, alavancas de comando. GY: válvulas solenóides.

Fonte: Adaptado de Oliva (2015).

Há ainda as fontes de esforço ($\boldsymbol{S_e}$) e de fluxo ($\boldsymbol{S_f}$), que suprem o sistema com um esforço e um fluxo, respectivamente (Figura B.9). Na figura o negrito simboliza que a variável E ou F é a de interesse – pois uma fonte é agente exclusivamente causador, sendo os efeitos do sistema sobre a fonte considerada desprezível. Essas fontes

estabelecem as fronteiras do sistema modelado, representando a interface entre este e o meio externo.

Figura B.9 – Relação entre notação GL e DB: fonte de esforço e de fluxo.



Fonte: Adaptado de Martins (2004).

Para compor a topologia do sistema há a necessidade de interligar todos os seus elementos. Na abordagem GL isso é feito através dos elementos de junção, em que ocorre acúmulo ou dissipação de energia – ou seja, eles estarão ligados a elementos do tipo C-I-R. Trata-se de elementos que distribuem potência entre vários elementos do sistema.

Existem dois tipos de junções: tipo-0 e tipo-1.

- Junção tipo-0

  É caracterizada pelo esforço comum. Todos os elementos ligados por essa junção estão sujeitos ao mesmo esforço. Neste caso, para garantir a lei da conservação de energia é necessário que o somatório das variáveis de fluxo seja nulo. Trata-se de uma generalização da Lei de Kirchoff para correntes (lei dos nós). Logo, tem-se que:

$$\boldsymbol{e_1 = e_2 = \cdots = e_n}$$

$$\boldsymbol{\sum_i^n f_i = f_1 + f_2 + \cdots + f_n = 0}$$

- Junção tipo-1

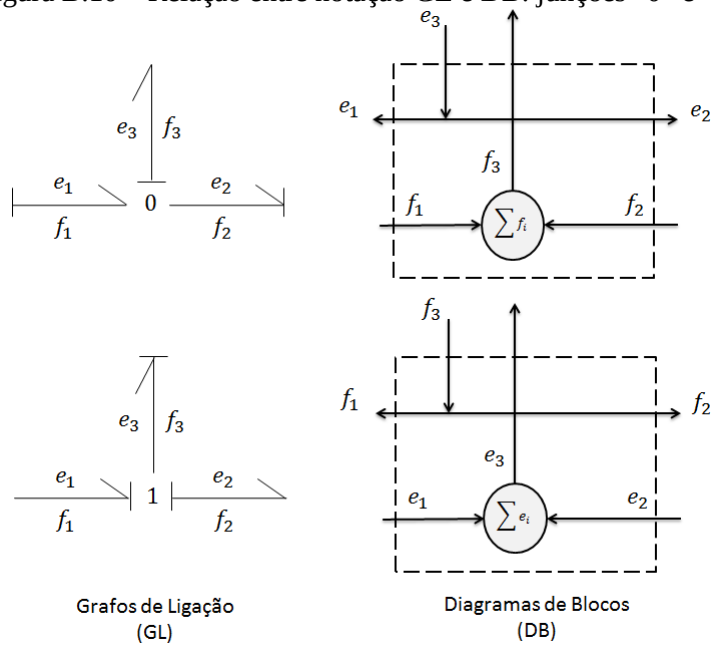
  Se caracteriza pelo fluxo comum. Todos os elementos ligados nessa junção estão sob ação do mesmo fluxo, o que implica que o somatório dos esforços deve ser zero. Aí tem-se a generalização da Lei de Kirchoff para tensões (lei das malhas). Logo temos que

$$\boldsymbol{f_1 = f_2 \ldots = f_n}$$

$$\boldsymbol{\sum_i^n e_i = e_1 + e_2 + \cdots + e_n}$$

A Figura B.10 revela a equivalência entre a simbologia GL e DB para os dois tipos de junções.

Figura B.10 – Relação entre notação GL e DB: junções “0” e “1”.



Fonte: Adaptado de Martins (2004).

Para sistemas físicos modelados a partir da abordagem informacional, parte-se de equações algébricas e/ou diferenciais para construção de DB, que será implementado em uma ferramenta computacional para simulação. A proposta da abordagem por fluxos de potência (física) é partir diretamente do diagrama do sistema físico para os GL, que representam os próprios componentes do sistema. Existem ferramentas adequadas para esse tipo de construção, dentre as quais pode-se citar o 20-SIM e o AMESim (*Advanced Modeling Environment for Simulations*), que se baseiam na abordagem por fluxos físicos. Dentre as vantagens da abordagem e simbologia bidirecional em relação à abordagem unidirecional, podemos citar:

a) A visualização de um sistema físico via GL é mais intuitiva, possibilitando uma comunicação mais fluída entre o engenheiro de requisitos / sistemas e o engenheiro de modelagem / simulação;

b) As variáveis generalizadas permitem aos engenheiros de desenvolvimento (M&S) de diversos domínios trabalharem em uma base comum, diminuindo as ambiguidades na comunicação e agilizando a construção de diagramas;

c) A depender do sistema físico e o grau de fidelidade requisitado (erro), a precisão dos resultados usando abordagem física pode ser muito superior.

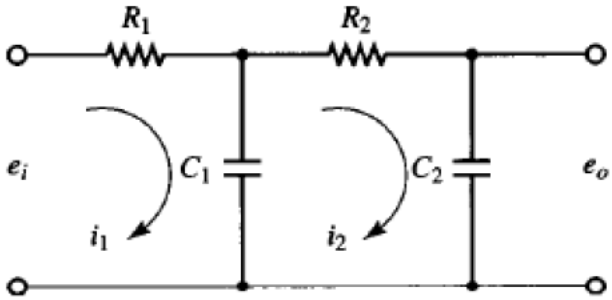
Pode-se ainda perguntar qual é de fato a finalidade dessa representação em termos concretos. Isso será esclarecido a seguir.

Em trabalhos anteriores já foi demonstrado a eficácia de se trabalhar com a abordagem física em contraponto à abordagem informacional para representar um sistema multidomínio. Ao se realizar uma validação comportamental em uma parte do subsistema em questão, ficou evidente a eficácia da abordagem por fluxos físicos (OLIVA, 2012).

Para melhor exemplificar a diferença entre a abordagem por fluxos físicos e por fluxo de informação no modelamento de sistemas, será usado um circuito elétrico com duas malhas. Tomando como exemplo um circuito RC com elementos em cascata (Figura B.11), observa-se que os componentes da malha 1 tem efeito de carga sobre aqueles da

malha 2. Admite-se que $\boldsymbol{e_i}$seja a entrada $\boldsymbol{e_0}$a saída. As capacitâncias C_1 e C_2 não estão carregadas inicialmente.

Figura B.11 – Circuito RC com duas malhas.



Fonte: Produção do autor

As equações diferenciais que descrevem esse sistema são:

$$\frac{1}{C_1}\int(i_1 - i_2)dt + R_1 \cdot i_1 = e_1 \quad \text{(B.1)}$$

$$\frac{1}{C_1}\int(i_2 - i_1)dt + R_2 \cdot i_2 + \frac{1}{C_2}\int i_2 dt = 0 \quad \text{(B.2)}$$

$$\frac{1}{C_2}\int i_2 dt = e_0 \quad \text{(B.3)}$$

Aplicando a Transformada de Laplace nas Equações (B.1) a (B.3), e considerando condições iniciais nulas, temos:

$$\frac{1}{C_1 \cdot s} \cdot [I_1(s) - I_2(s)] + R_1 \cdot I_1(s) = E_i(s) \quad \text{(B.4)}$$

$$\frac{1}{C_1 \cdot s} \cdot [I_2(s) - I_1(s)] + R_2 \cdot I_2(s) + \frac{1}{C_2 \cdot s} \cdot I_2(s) = E_i(s) \quad \text{(B.5)}$$

$$\frac{1}{C_2 \cdot s} \cdot I_2(s) = E_0(s) \quad \text{(B.6)}$$

Eliminando $I_1(s)$das Equações (B.4) e (B.5) e escrevendo $E_i(s)$, em termos de $I_2(s)$, encontramos a função de transferência entre $E_0(s)$e $E_i(s)$.

$$\frac{E_0(s)}{E_i(s)} = \frac{1}{(R_1C_1s+1)\cdot(R_2C_2s+1)+R_1C_2s}$$

$$= \frac{\mathbf{1}}{\boldsymbol{R_1C_1R_2C_2s^2+(R_1C_1+R_2C_2+R_1C_2)s}} \quad \text{(B.7)}$$

O termo$\boldsymbol{c_c c_c c}$ no denominador da função de transferência representa a interação entre as duas malhas do circuito RC. Essa análise mostra que, se dois circuitos RC estão conectados em cascata, de modo que a entrada do primeiro é a entrada do segundo, a função de transferência global não é simplesmente o produto $\frac{\mathbf{1}}{(\boldsymbol{R1C1s+1})}$ e $\frac{\mathbf{1}}{(\boldsymbol{R2C2s+1})}$. Isso ocorre porque quando deduzimos a função de transferência para um circuito isolado, estamos assumindo que a saída do circuito esteja sem carga. Isto é, nenhuma potência está sendo retirada da saída, o que não é verdade para esse caso.

Conclui-se desse exemplo que o sistema da malha 2 influencia o comportamento do sistema da malha 1, e vice-versa.

Se for construído um modelo em diagrama de blocos, que considera apenas o sinal enviado de um submodelo para outro, deve-se ter uma função de transferência definida por uma função de transferência, construída a partir da Transformada de Laplace (OGATA,2010). Não é possível representar o sistema usando duas funções de transferência com atraso de 1ª ordem. O princípio da aditividade não se aplica. Ou seja, trata-se de um sistema não-linear, que é uma das características de sistemas complexos.

Um último aspecto importante da modelagem – em particular da abordagem física – se refere à questão dos parâmetros, que podem ser representados pontualmente, na qual alguma grandeza física flui através de um circuito que transporta massa, energia, quantidade de movimentos, corrente elétrica, etc. (concentrados, ou 1D); ou pela dispersão dos mesmos por todo espaço (contínuos, ou 3D).

Em ciência da computação, a modelagem tridimensional (3D) é o processo de desenvolvimento de uma representação matemática de qualquer superfície tridimensional de um objeto (seja inanimado ou vivo), através de software especializado.

A simulação 3D está relacionada à análise estrutural, escoamento externo ou interno de fluídos, tensões térmicas, entre outras questões dessa natureza.

A modelagem unidimensional (1D) é aquela que representa sistemas através de parâmetros concentrados. Isso significa, por exemplo, que a massa de um corpo sujeito a uma força, é considerada concentrada num único ponto – o de atuação da força. Ou que o escoamento de um fluído numa tubulação terá sua variável "fluxo mássico", "temperatura" ou "pressão", por exemplo, medida em seções específicas, pré-definidas. Em suma, reproduz-se o comportamento do modelo no tempo, mas as variáveis de interesse são mensuradas em pontos específicos.

Por que realizar simulações 1D ao invés de 3D, uma vez que esta parece englobar mais aspectos? Primeiro: a simulação 3D, por representar um componente ou sistema em sua integridade, terá um tempo de execução maior do que a simulação 1D – às vezes muito maior, da ordem de dias, comparada a uma simulação de minutos num modelo de parâmetros concentrados. Segundo: esse tempo maior exige uma memória maior do computador executando-a, o que significa necessidade de processadores mais rápidos e com mais memória – tanto para armazenamento quanto para execução. Terceiro: muitos fenômenos podem ser descritos com um bom grau de fidelidade sem necessidade de saber como as variáveis estão se comportando em todos os pontos no sistema ou circunjacentes a ele (em termos espaciais ou de influência). Isso vale em larga medida para fenômenos mecânicos e elétricos, e em casos particulares, em fenômenos hidráulicos, pneumáticos e térmicos. Essas observações são válidas para o campo das ciências físicas e químicas. Ao entrarmos no campo de controle não se trabalha o conceito de tridimensionalidade espacial. Ou seja, ele se situa na simulação 1D. Além disso, como – no âmbito do controlador – não há fenômenos físicos, a dinâmica característica destes, com seus atrasos, não é representada. A relação causa-efeito assume apenas um sentido, ao contrário dos sistemas físicos. Trata-se de fenômenos que exigem abordagens diferentes para serem representados com o devido grau de fidelidade.

Essas características que diferenciam sistemas físicos e sistemas de controle serão observadas mais pormenorizadamente no próximo tópico.

APÊNDICE C – CÓDIGOS MATLAB E SCILAB

mission_and_environment.m

```
%% INSERÇÃO DOS PARÂMETROS DE MISSÃO E DE AMBIENTE

clear all

close all

clc

%% PARÂMETROS DE ENTRADA DO SISTEMA (dimensões e materiais)

quantidade_param = 14;     % número de parâmetros de entrada

%% Corpo do satélite
valores originais

a = 0.4;       % largura do corpo do satélite [m]
0.4

b = 0.3;       % comprimento do corpo do satélite [m]

h1 = 0.4;      % altura do corpo do satélite [m]

e1 = 0.01;     % espessura das placas que compõem corpo do satélite
[m]

ro1 = 2000;    % densidade do material do corpo [kg/m3]

%% Painéis solares

c = 0.7;       % largura da placa solar [m]
0.7

h2 = 0.30;     % altura da placa solar [m]
0.3

e2 = 0.02;     % espessura das placas dos painéis solares [m]

L = 0.05;      % distância entre placa e corpo do satélite [m]

ro2 = 2000;    % densidade do material [kg/m3]
2000.0

%% Tanque esférico

Ri = 0.29;        % raio interno do tanque [m]
0.29

Re = 0.30;        % raio externo do tanque [m]
0.30

R = (Ri+Re)/2;    % raio médio do tanque (caso esférico)
0.295
```

```
ro3 = 2000;       % densidade do material [kg/m3]

%% Salvar para Input_parameters_num =
fopen('Input_parameters_num.data','w');

Input_parameters_num = fopen('Input_parameters_num.data','w');

fprintf(Input_parameters_num, '%f \n', a);

fprintf(Input_parameters_num, '%f \n', b);

fprintf(Input_parameters_num, '%f \n', c);

fprintf(Input_parameters_num, '%f \n', h1);

fprintf(Input_parameters_num, '%f \n', h2);

fprintf(Input_parameters_num, '%f \n', e1);

fprintf(Input_parameters_num, '%f \n', e2);

fprintf(Input_parameters_num, '%f \n', Ri);

fprintf(Input_parameters_num, '%f \n', Re);

fprintf(Input_parameters_num, '%f \n', R);

fprintf(Input_parameters_num, '%f \n', L);

fprintf(Input_parameters_num, '%f \n', ro1);

fprintf(Input_parameters_num, '%f \n', ro2);

fprintf(Input_parameters_num, '%f \n', ro3);

fclose(Input_parameters_num)

%% Salvar parametros de entrada (dimensoes / material[densidade]) -
grandezas e unidades

Input_parameters_unidade = fopen('Input_parameters_unidade.data','w');



fprintf(Input_parameters_unidade, 'a: altura [m] \n');

fprintf(Input_parameters_unidade, 'b: largura [m] \n');

fprintf(Input_parameters_unidade, 'c: profundidade [m] \n');

fprintf(Input_parameters_unidade, 'h1: altura painel [m]\n');

fprintf(Input_parameters_unidade, 'h2: altura sat. [m] \n');

fprintf(Input_parameters_unidade, 'e1: espessura corpo satelite [m]
\n');
```

```
fprintf(Input_parameters_unidade, 'e2: espessura paineis [m] \n');

fprintf(Input_parameters_unidade, 'Ri: raio interno do tanque [m]
\n');

fprintf(Input_parameters_unidade, 'Re: raio externo do tanque [m]
\n');

fprintf(Input_parameters_unidade, 'R: raio medio do tanque [m] \n');

fprintf(Input_parameters_unidade, 'L: disância entre painéis e corpo
sat. [m] \n');

fprintf(Input_parameters_unidade, 'ro1: densidade media do corpo
satelite [kg/m3] \n', ro1);

fprintf(Input_parameters_unidade, 'ro2: densidade media paineis
[kg/m3] \n', ro2);

fprintf(Input_parameters_unidade, 'ro3: densidade tanque esferico
[kg/m3] \n', ro3);



fclose(Input_parameters_unidade)



%% AMBIENTE ESPACIAL: torques externos, radiação, órbita

%% Torque aerodinâmico

alfa = 0;               % Inclinação dos painéis solares: [º]
dado de entrada

H = 400;                % Órbita [km]
dado de entrada

mi = 3.986*10^14;

Rt = 6378 + H;                % distância CM_Terra / CM_Sat [km]

n = sqrt(mi/(Rt*10^3)^3);     % frequência orbital sat [Hz]



% Determibação da acelareação da gravidade na órbita do satélite

M = 6E24;          % massa Terra

G = 6.7E-11;       % cte gravitacional universal

d = Rt*1000;       % distância centro-Terra e satélite

g = G*M/(d^2);     % g=f(M,G,d)
```

```
%% Configuração dos painéis

config = 1;   % Tipo de posicionamento 1-> painéis paralelos ou
olbíquos /    dado de entrada

              %                        2-> painéis perpendiculares)

% área perpendicular ao vetor velocidade [m2]

if config == 1

    S = a*h1 + 2*c*(h2*sin(alfa));

end


if config ==2

    S = b*h1;

end


%% Informações atmosféricas

ro_amb = 3e-11;      % densidade ambiente = f(H) [kg/m3]

V_sat = 7613;             % velocidade satélite [m/s]

Cd = 2;                   % coeficiente de arrasto [ ]

U_Scp = 0.2;              % produto vetorial |UxScp|  [m]

%% Salvar parametros de entrada (ambiente) - magnitude

Ambiente_num = fopen('Ambiente_num.data','w');

fprintf(Ambiente_num, '%f \n', n);

fprintf(Ambiente_num, '%f \n', ro_amb);

fprintf(Ambiente_num, '%f \n', V_sat);

fprintf(Ambiente_num, '%f \n', Cd);

fprintf(Ambiente_num, '%f \n', U_Scp);

fprintf(Ambiente_num, '%f \n', S);

fclose(Ambiente_num)

%% Salvar parametros de entrada (ambiente) - magnitudeGrandezas e
unidades dos dados

Ambiente = fopen('Ambiente_grandeza_unidade.data','w');
```

```
fprintf(Ambiente, 'n = frequencia orbital [Hz] \n');

fprintf(Ambiente, 'ro_amb = densidade atmosferica \n');

fprintf(Ambiente, 'V_sat = velocidade orbital do satelite [m/s] \n');

fprintf(Ambiente, 'Cd = coeficiente de arrasto [ ] \n');

fprintf(Ambiente, 'U_Scp = produto vetorial velocidade-area frontal,
UxScp [m] \n');

fprintf(Ambiente, 'S = area dos paineis projetada na direcao de
deslocamento [m2] \n');

fclose(Ambiente)

%% OBS

% Torque do gradiente gravitacional [já implícito no modelo]
```

input_mass_prop.m

```
%% CÁLCULO DE PROPRIEDADES INERCIAIS DA ESTRUTURA

clc

%% Referencia CG do veiculo

XG = 0; YG = 0; ZG = 0;       % referencial

%% LEITURA DOS ARQUIVOS DE PARÂMETROS DE ENTRADA
(mission_and_envirionment.m)

param = fopen('Input_parameters_num.data');

p1 = fscanf(param,'%f \n',[12,inf]);

fclose(param);


param2 = fopen('Input_parameters_unidade.data');

p2 = fscanf(param2,'%s           \n',[12,inf]);

fclose(param2);


% for k=1:12

%     p2(k) = p1(k);

% end
```

```
a = p1(1);

b = p1(2);

c = p1(3);

h1 = p1(4);

h2 = p1(5);

e1 = p1(6);

e2 = p1(7);

Ri = p1(8);

Re = p1(9);

R = p1(10);

L = p1(11);

ro1 = p1(12);

ro2 = p1(13);

ro3 = p1(14);


%% CÁLCULO DOS VOLUMES E MASSAS (parte 1)

Vol_1 = a*b*h1 - (a - 2*e1)*(b - 2*e1)*(h1 - 2*e1);   % volume das
partes que compõem o corpo do satélite

m1 = ro1*Vol_1;                                       % massa do corpo


CM_1 = [b/2; a/2; h1/2];      % centro de massa {X1, Y1, Z1} do corpo


%% MOMENTO DE INÉRCIA DO CORPO DO SATÉLITE (parte 1)

Ix_1 = (m1/3)*(a.^2 + h1.^2);

Iy_1 = (m1/3)*(b.^2 + h1.^2);

Iz_1 = (m1/3)*(a.^2 + b.^2);


%% CENTRO DE MASSA PRINCIPAL (parte 1)

XG_1 = XG - CM_1(1);         % XG_1 = XG - X1
```

```
YG_1 = YG - CM_1(2);         % YG_1 = YG - Y1

ZG_1 = ZG - CM_1(3);         % ZG_1 = ZG - Z1

CM_1p = [XG_1; YG_1; ZG_1];


%% MOMENTO DE INÉRCIA PRINCIPAL (parte 1)

Ix_1p = (m1/3)* (  ( ( (a - e1)^3 - e1^3) / (a - 2*e1) ) + ((h1 -
e1)^3/ (h1 - 2*e1)) );

Iy_1p = (m1/3)* (  ( ( (b - e1)^3 - e1^3) / (b - 2*e1) ) + ((h1 -
e1)^3/ (h1 - 2*e1)) );

Iz_1p = (m1/3)* (  ( ( (a - e1)^3 - e1^3) / (a - 2*e1) ) + ((b -
e1)^3/ (b - 2*e1)) );


I_1 = [Ix_1p 0 0; 0 Iy_1p 0; 0 0 Iz_1p];


%% CÁLCULO DOS VOLUMES E MASSAS (parte 2)

Vol_2 = c*h2*e2;     % volume das placas

m2 = ro2*Vol_2;      % massa do corpo


CM_21 = [b/2; (a+L+c/2); h2/2];      % centro de massa {X1, Y1, Z1} da
placa 1

CM_22 = [b/2; (-L-c/2); h2/2];       % centro de massa {X1, Y1, Z1} da
placa 2


%% MOMENTO DE INÉRCIA DOS PAINÉIS SOLARES (parte 2)

Ix_2 = (m2/12)*(c^2 + h2^2);

Iy_2 = (m2/12)*h2^2;

Iz_2 = (m2/12)*c^2;

%% CENTRO DE MASSA PRINCIPAL (parte 2)

XG_21 = XG - CM_21(1);         % XG_2 = XG - X21

YG_21 = YG - CM_21(2);         % YG_2 = YG - Y21

ZG_21 = ZG - CM_21(3);         % ZG_2 = ZG - Z21
```

```
XG_22 = XG - CM_22(1);          % XG_2 = XG - X22

YG_22 = YG - CM_22(2);          % YG_2 = YG - Y22

ZG_22 = ZG - CM_22(3);          % ZG_2 = ZG - Z22


CM_21p = [XG_21; YG_21; ZG_21];

CM_22p = [XG_22; YG_22; ZG_22];


%% MOMENTO DE INÉRCIA PRINCIPAL (parte 2)

Ix_21p = (m2/12)*(c^2 + h2^2) + m2*( (-(a+L+c/2))^2 + (-h2/2)^2  );
% painel solar #1

Iy_21p = (m2/12)*(h2^2) + m2*( (-b/2)^2 + (h2/2)^2  );

Iz_21p = (m2/12)*(c^2) + m2*( (-b/2)^2 + (-(a+L+c/2))^2  );

Ix_22p = (m2/12)*(c^2 + h2^2) + m2*( (L+c/2)^2 + (-h2/2)^2  );
% painel solar #2

Iy_22p = (m2/12)*(h2^2) + m2*( (-b/2)^2 + (h2/2)^2  );

Iz_22p = (m2/12)*(c^2) + m2*( (-b/2)^2 + (L+c/2)^2  );


Ix_2p = Ix_21p + Ix_22p;
% conjunto painéis

Iy_2p = Iy_21p + Iy_22p;

Iz_2p = Iz_21p + Iz_22p;

I_2 = [Ix_2p 0 0; 0 Iy_2p 0; 0 0 Iz_2p];


%% CÁLCULO DOS VOLUMES E MASSAS TANQUE ESFÉRICO (parte 3)

Vol_3 = (4/3)*pi*(Re^3 - Ri^3);    % volume do tanque

m3 = ro3*Vol_3;                    % massa do corpo


CM_3 = [b/2; a/2; R];              % centro de massa {X1, Y1, Z1} do
tanque
```

```
%% MOMENTO DE INÉRCIA DO CORPO DO SATÉLITE (parte 3)

Ix_3 = (3/2)*m3* ((Re^5 - Ri^5) / (Re^3 - Ri^3));

Iy_3 = (3/2)*m3* ((Re^5 - Ri^5) / (Re^3 - Ri^3));

Iz_3 = (3/2)*m3* ((Re^5 - Ri^5) / (Re^3 - Ri^3));


%% CENTRO DE MASSA PRINCIPAL (parte 3)

XG_3 = XG - CM_3(1);         % XG_3 = XG - X3

YG_3 = YG - CM_3(2);         % YG_3 = YG - Y3

ZG_3 = ZG - CM_3(3);         % ZG_3 = ZG - Z3


CM_3p = [XG_21; YG_21; ZG_21];

CM_3p = [XG_22; YG_22; ZG_22];


%% MOMENTO DE INÉRCIA PRINCIPAL (parte 2)

Ix_3p = (3/2)*m3*(R^2) + m3*( (-a/2)^2 + (-R)^2  );

Iy_3p = (3/2)*m3*(R^2) + m3*( (-b/2)^2 + (-R)^2  );

Iz_3p = (3/2)*m3*(R^2) + m3*( (-b/2)^2 + (-a/2)^2  );

I_3 = [Ix_3p 0 0; 0 Iy_3p 0; 0 0 Iz_3p];


%% PROPRIEDADES DO SATÉLITE (parte 1 + parte 2 + parte 3)

%% Centro de massa

CM_x = (m1*CM_1(1) + m2*(CM_21(1)+CM_22(1)) + m3*CM_3(1)) /
(m1+2*m2+m3);

CM_y = (m1*CM_1(2) + m2*(CM_21(2)+CM_22(2)) + m3*CM_3(2)) /
(m1+2*m2+m3);

CM_z = (m1*CM_1(3) + m2*(CM_21(3)+CM_22(3)) + m3*CM_3(3)) /
(m1+2*m2+m3);

CM_sat = [CM_x; CM_y; CM_z];

%% Momento de inércia

I_sat = I_1 + I_2 + I_3;
```

```
I1 = I_sat(1,1);

I2 = I_sat(2,2);

I3 = I_sat(3,3);


%% Definição dos arquivos de saída - Matriz inercia com nomes

Inercia = fopen('Momento_inercia.data','w');

fprintf(Inercia,'Ixx = %f \n',I1)

fprintf(Inercia,'Iyy = %f \n',I2)

fprintf(Inercia,'Izz = %f \n',I3)


fclose(Inercia)

%% Definição dos arquivos de saída - Matriz inercia sem nomes (apenas
numeros)

Inercia_num = fopen('Momento_inercia_num.data','w');

fprintf(Inercia_num,'%f \n',I1)

fprintf(Inercia_num,'%f \n',I2)

fprintf(Inercia_num,'%f \n',I3)

fclose(Inercia_num)

%% Vetor centro de massa

CM = fopen('Centro_massa.data','w');

fprintf(CM,'CM_sat_x = %f \n',CM_x)

fprintf(CM,'CM_sat_y = %f \n',CM_y)

fprintf(CM,'CM_sat_z = %f \n',CM_z)


fclose(CM)

CM_num = fopen('Centro_massa_num.data','w');

fprintf(CM_num,' %f \n',CM_x)

fprintf(CM_num,' %f \n',CM_y)

fprintf(CM_num,' %f \n',CM_z)
```

```
fclose(CM_num)
```

input_mass_prop_v1.m

```
%% CÁLCULO DE PROPRIEDADES INERCIAIS DA ESTRUTURA
clc
%% Referencia CG do veiculo
XG = 0; YG = 0; ZG = 0;       % referencial
%% LEITURA DOS ARQUIVOS DE PARÂMETROS DE ENTRADA
token = fopen('token_ADE.data');
p3 = fscanf(token,'%f');
fclose(token);
if p3==0
    param = fopen('Input_parameters_num.data');
    p1 = fscanf(param,'%f \n',[12,inf]);
    fclose(param);
else
    param = fopen('Input_parameters_num_2.data');
    p1 = fscanf(param,'%f \n',[12,inf]);
    fclose(param);
end
param2 = fopen('Input_parameters_unidade.data');
p2 = fscanf(param2,'%s \n',[12,inf]);
fclose(param2);

a = p1(1);
b = p1(2);
c = p1(3);
h1 = p1(4);
h2 = p1(5);
```

```
e1 = p1(6);

e2 = p1(7);

Ri = p1(8);

Re = p1(9);

R = p1(10);

L = p1(11);

ro1 = p1(12);

ro2 = p1(13);

ro3 = p1(14);

% for k=1:12

%     p2(k) = p1(k);

% end


%% CÁLCULO DOS VOLUMES E MASSAS (parte 1)

Vol_1 = a*b*h1 - (a - 2*e1)*(b - 2*e1)*(h1 - 2*e1);   % volume das
partes que compõem o corpo do satélite

m1 = ro1*Vol_1;                                       % massa do corpo

CM_1 = [b/2; a/2; h1/2];      % centro de massa {X1, Y1, Z1} do corpo


%% MOMENTO DE INÉRCIA DO CORPO DO SATÉLITE (parte 1)

Ix_1 = (m1/3)*(a.^2 + h1.^2);

Iy_1 = (m1/3)*(b.^2 + h1.^2);

Iz_1 = (m1/3)*(a.^2 + b.^2);


%% CENTRO DE MASSA PRINCIPAL (parte 1)

XG_1 = XG - CM_1(1);         % XG_1 = XG - X1

YG_1 = YG - CM_1(2);         % YG_1 = YG - Y1

ZG_1 = ZG - CM_1(3);         % ZG_1 = ZG - Z1

CM_1p = [XG_1; YG_1; ZG_1];
```

```
%% MOMENTO DE INÉRCIA PRINCIPAL (parte 1)

Ix_1p = (m1/3)* (  ( ( (a - e1)^3 - e1^3) / (a - 2*e1) ) + ((h1 -
e1)^3/ (h1 - 2*e1)) );

Iy_1p = (m1/3)* (  ( ( (b - e1)^3 - e1^3) / (b - 2*e1) ) + ((h1 -
e1)^3/ (h1 - 2*e1)) );

Iz_1p = (m1/3)* (  ( ( (a - e1)^3 - e1^3) / (a - 2*e1) ) + ((b -
e1)^3/ (b - 2*e1)) );

I_1 = [Ix_1p 0 0; 0 Iy_1p 0; 0 0 Iz_1p];



%% CÁLCULO DOS VOLUMES E MASSAS (parte 2)

Vol_2 = c*h2*e2;     % volume das placas

m2 = ro2*Vol_2;      % massa do corpo



CM_21 = [b/2; (a+L+c/2); h2/2];      % centro de massa {X1, Y1, Z1} da
placa 1

CM_22 = [b/2; (-L-c/2); h2/2];       % centro de massa {X1, Y1, Z1} da
placa 2

%% MOMENTO DE INÉRCIA DOS PAINÉIS SOLARES (parte 2)

Ix_2 = (m2/12)*(c^2 + h2^2);

Iy_2 = (m2/12)*h2^2;

Iz_2 = (m2/12)*c^2;

%% CENTRO DE MASSA PRINCIPAL (parte 2)

XG_21 = XG - CM_21(1);         % XG_2 = XG - X21

YG_21 = YG - CM_21(2);         % YG_2 = YG - Y21

ZG_21 = ZG - CM_21(3);         % ZG_2 = ZG - Z21



XG_22 = XG - CM_22(1);         % XG_2 = XG - X22

YG_22 = YG - CM_22(2);         % YG_2 = YG - Y22

ZG_22 = ZG - CM_22(3);         % ZG_2 = ZG - Z22

CM_21p = [XG_21; YG_21; ZG_21];

CM_22p = [XG_22; YG_22; ZG_22];
```

```
%% MOMENTO DE INÉRCIA PRINCIPAL (parte 2)

Ix_21p = (m2/12)*(c^2 + h2^2) + m2*( (-(a+L+c/2))^2 + (-h2/2)^2  );
% painel solar #1

Iy_21p = (m2/12)*(h2^2) + m2*( (-b/2)^2 + (h2/2)^2  );

Iz_21p = (m2/12)*(c^2) + m2*( (-b/2)^2 + (-(a+L+c/2))^2  );

Ix_22p = (m2/12)*(c^2 + h2^2) + m2*( (L+c/2)^2 + (-h2/2)^2  );
% painel solar #2

Iy_22p = (m2/12)*(h2^2) + m2*( (-b/2)^2 + (h2/2)^2  );

Iz_22p = (m2/12)*(c^2) + m2*( (-b/2)^2 + (L+c/2)^2  );

Ix_2p = Ix_21p + Ix_22p;
% conjunto painéis

Iy_2p = Iy_21p + Iy_22p;

Iz_2p = Iz_21p + Iz_22p;

I_2 = [Ix_2p 0 0; 0 Iy_2p 0; 0 0 Iz_2p];



%% CÁLCULO DOS VOLUMES E MASSAS TANQUE ESFÉRICO (parte 3)

Vol_3 = (4/3)*pi*(Re^3 - Ri^3);    % volume do tanque

m3 = ro3*Vol_3;                    % massa do corpo

CM_3 = [b/2; a/2; R];              % centro de massa {X1, Y1, Z1} do
tanque

%% MOMENTO DE INÉRCIA DO CORPO DO SATÉLITE (parte 3)

Ix_3 = (3/2)*m3* ((Re^5 - Ri^5) / (Re^3 - Ri^3));

Iy_3 = (3/2)*m3* ((Re^5 - Ri^5) / (Re^3 - Ri^3));

Iz_3 = (3/2)*m3* ((Re^5 - Ri^5) / (Re^3 - Ri^3));



%% CENTRO DE MASSA PRINCIPAL (parte 3)

XG_3 = XG - CM_3(1);         % XG_3 = XG - X3

YG_3 = YG - CM_3(2);         % YG_3 = YG - Y3

ZG_3 = ZG - CM_3(3);         % ZG_3 = ZG - Z3

CM_3p = [XG_21; YG_21; ZG_21];
```

```
CM_3p = [XG_22; YG_22; ZG_22];

%% MOMENTO DE INÉRCIA PRINCIPAL (parte 2)

Ix_3p = (3/2)*m3*(R^2) + m3*( (-a/2)^2 + (-R)^2  );

Iy_3p = (3/2)*m3*(R^2) + m3*( (-b/2)^2 + (-R)^2  );

Iz_3p = (3/2)*m3*(R^2) + m3*( (-b/2)^2 + (-a/2)^2  );


I_3 = [Ix_3p 0 0; 0 Iy_3p 0; 0 0 Iz_3p];


%% PROPRIEDADES DO SATÉLITE (parte 1 + parte 2 + parte 3)

%% Centro de massa

CM_x = (m1*CM_1(1) + m2*(CM_21(1)+CM_22(1)) + m3*CM_3(1)) /
(m1+2*m2+m3);

CM_y = (m1*CM_1(2) + m2*(CM_21(2)+CM_22(2)) + m3*CM_3(2)) /
(m1+2*m2+m3);

CM_z = (m1*CM_1(3) + m2*(CM_21(3)+CM_22(3)) + m3*CM_3(3)) /
(m1+2*m2+m3);

CM_sat = [CM_x; CM_y; CM_z];

%% Momento de inércia

I_sat = I_1 + I_2 + I_3;

I1 = I_sat(1,1);

I2 = I_sat(2,2);

I3 = I_sat(3,3);


%% Definição dos arquivos de saída - Matriz inercia com nomes

Inercia = fopen('Momento_inercia.data','w');

fprintf(Inercia,'Ixx = %f \n',I1)

fprintf(Inercia,'Iyy = %f \n',I2)

fprintf(Inercia,'Izz = %f \n',I3)

fclose(Inercia)
```

```
%% Definição dos arquivos de saída - Matriz inercia sem nomes (apenas
numeros)

Inercia_num = fopen('Momento_inercia_num.data','w');

fprintf(Inercia_num,'%f \n',I1)

fprintf(Inercia_num,'%f \n',I2)

fprintf(Inercia_num,'%f \n',I3)

fclose(Inercia_num)

%% Vetor centro de massa

CM = fopen('Centro_massa.data','w');

fprintf(CM,'CM_sat_x = %f \n',CM_x)

fprintf(CM,'CM_sat_y = %f \n',CM_y)

fprintf(CM,'CM_sat_z = %f \n',CM_z)

fclose(CM)

CM_num = fopen('Centro_massa_num.data','w');

fprintf(CM_num,' %f \n',CM_x)

fprintf(CM_num,' %f \n',CM_y)

fprintf(CM_num,' %f \n',CM_z)

fclose(CM_num)
```

input_mass_prop.sce

```
// CÁLCULO DE PROPRIEDADES INERCIAIS DA ESTRUTURA

clc


// Referencia CG do veiculo

XG = 0; YG = 0; ZG = 0;       // referencial

// LEITURA DOS ARQUIVOS DE PARÂMETROS DE ENTRADA
(mission_and_envirionment.m)

// param = mopen( + '/Input_parameters_num.data', 'rt')

param = read('Input_parameters_num.data',14,1)

//param = mopen('Input_parameters_num.data','rt');
```

```
//param = mfscanf(param,'%f \n %f \n %f \n %f \n %f \n %f \n %f \n %f
\n %f \n %f \n %f \n %f \n %f \n %f \n');

//mclose(param)

param2 = mopen('Input_parameters_unidade.data');

//p2 = mfscanf(param2,'%s           \n',[12,inf]);

//mclose(param2)

// for k=1:12

//     p2(k) = param(k);

// end

a = param(1);

b = param(2);

c = param(3);

h1 = param(4);

h2 = param(5);

e1 = param(6);

e2 = param(7);

Ri = param(8);

Re = param(9);

R = param(10);

L = param(11);

ro1 = param(12);

ro2 = param(13);

ro3 = param(14);


//// CÁLCULO DOS VOLUMES E MASSAS (parte 1)

Vol_1 = a*b*h1 - (a - 2*e1)*(b - 2*e1)*(h1 - 2*e1);   // volume das
partes que compõem o corpo do satélite

m1 = ro1*Vol_1;                                       // massa do
corpo

CM_1 = [b/2; a/2; h1/2];      // centro de massa {X1, Y1, Z1} do corpo
```

```
//// MOMENTO DE INÉRCIA DO CORPO DO SATÉLITE (parte 1)

Ix_1 = (m1/3)*(a.^2 + h1.^2);

Iy_1 = (m1/3)*(b.^2 + h1.^2);

Iz_1 = (m1/3)*(a.^2 + b.^2);

//// CENTRO DE MASSA PRINCIPAL (parte 1)

XG_1 = XG - CM_1(1);         // XG_1 = XG - X1

YG_1 = YG - CM_1(2);         // YG_1 = YG - Y1

ZG_1 = ZG - CM_1(3);         // ZG_1 = ZG - Z1

CM_1p = [XG_1; YG_1; ZG_1];

// MOMENTO DE INÉRCIA PRINCIPAL (parte 1)

Ix_1p = (m1/3)* (  ( ( (a - e1)^3 - e1^3) / (a - 2*e1) ) + ((h1 -
e1)^3/ (h1 - 2*e1)) );

Iy_1p = (m1/3)* (  ( ( (b - e1)^3 - e1^3) / (b - 2*e1) ) + ((h1 -
e1)^3/ (h1 - 2*e1)) );

Iz_1p = (m1/3)* (  ( ( (a - e1)^3 - e1^3) / (a - 2*e1) ) + ((b -
e1)^3/ (b - 2*e1)) );



I_1 = [Ix_1p 0 0; 0 Iy_1p 0; 0 0 Iz_1p];



// CÁLCULO DOS VOLUMES E MASSAS (parte 2)

Vol_2 = c*h2*e2;     // volume das placas

m2 = ro2*Vol_2;      // massa do corpo

CM_21 = [b/2; (a+L+c/2); h2/2];      // centro de massa {X1, Y1, Z1}
da placa 1

CM_22 = [b/2; (-L-c/2); h2/2];       // centro de massa {X1, Y1, Z1}
da placa 2

// MOMENTO DE INÉRCIA DOS PAINÉIS SOLARES (parte 2)

Ix_2 = (m2/12)*(c^2 + h2^2);

Iy_2 = (m2/12)*h2^2;

Iz_2 = (m2/12)*c^2;

//// CENTRO DE MASSA PRINCIPAL (parte 2)
```

```
XG_21 = XG - CM_21(1);          // XG_2 = XG - X21

YG_21 = YG - CM_21(2);          // YG_2 = YG - Y21

ZG_21 = ZG - CM_21(3);          // ZG_2 = ZG - Z21

XG_22 = XG - CM_22(1);          // XG_2 = XG - X22

YG_22 = YG - CM_22(2);          // YG_2 = YG - Y22

ZG_22 = ZG - CM_22(3);          // ZG_2 = ZG - Z22

CM_21p = [XG_21; YG_21; ZG_21];

CM_22p = [XG_22; YG_22; ZG_22];



// MOMENTO DE INÉRCIA PRINCIPAL (parte 2)

Ix_21p = (m2/12)*(c^2 + h2^2) + m2*( (-(a+L+c/2))^2 + (-h2/2)^2  );
// painel solar #1

Iy_21p = (m2/12)*(h2^2) + m2*( (-b/2)^2 + (h2/2)^2  );

Iz_21p = (m2/12)*(c^2) + m2*( (-b/2)^2 + (-(a+L+c/2))^2  );

Ix_22p = (m2/12)*(c^2 + h2^2) + m2*( (L+c/2)^2 + (-h2/2)^2  );
// painel solar #2

Iy_22p = (m2/12)*(h2^2) + m2*( (-b/2)^2 + (h2/2)^2  );

Iz_22p = (m2/12)*(c^2) + m2*( (-b/2)^2 + (L+c/2)^2  );

Ix_2p = Ix_21p + Ix_22p;
// conjunto painéis

Iy_2p = Iy_21p + Iy_22p;

Iz_2p = Iz_21p + Iz_22p;

I_2 = [Ix_2p 0 0; 0 Iy_2p 0; 0 0 Iz_2p];




// CÁLCULO DOS VOLUMES E MASSAS TANQUE ESFÉRICO (parte 3)

Vol_3 = (4/3)*(3.14159)*(Re^3 - Ri^3);    // volume do tanque

m3 = ro3*Vol_3;                    // massa do corpo

CM_3 = [b/2; a/2; R];              // centro de massa {X1, Y1, Z1} do
tanque
```

```
//// MOMENTO DE INÉRCIA DO CORPO DO SATÉLITE (parte 3)

Ix_3 = (3/2)*m3* ((Re^5 - Ri^5) / (Re^3 - Ri^3));

Iy_3 = (3/2)*m3* ((Re^5 - Ri^5) / (Re^3 - Ri^3));

Iz_3 = (3/2)*m3* ((Re^5 - Ri^5) / (Re^3 - Ri^3));

//// CENTRO DE MASSA PRINCIPAL (parte 3)

XG_3 = XG - CM_3(1);         // XG_3 = XG - X3

YG_3 = YG - CM_3(2);         // YG_3 = YG - Y3

ZG_3 = ZG - CM_3(3);         // ZG_3 = ZG - Z3

CM_3p = [XG_21; YG_21; ZG_21];

CM_3p = [XG_22; YG_22; ZG_22];

// MOMENTO DE INÉRCIA PRINCIPAL (parte 2)

Ix_3p = (3/2)*m3*(R^2) + m3*( (-a/2)^2 + (-R)^2  );

Iy_3p = (3/2)*m3*(R^2) + m3*( (-b/2)^2 + (-R)^2  );

Iz_3p = (3/2)*m3*(R^2) + m3*( (-b/2)^2 + (-a/2)^2  );

I_3 = [Ix_3p 0 0; 0 Iy_3p 0; 0 0 Iz_3p];


// PROPRIEDADES DO SATÉLITE (parte 1 + parte 2 + parte 3)

// Centro de massa

CM_x = (m1*CM_1(1) + m2*(CM_21(1)+CM_22(1)) + m3*CM_3(1)) /
(m1+2*m2+m3);

CM_y = (m1*CM_1(2) + m2*(CM_21(2)+CM_22(2)) + m3*CM_3(2)) /
(m1+2*m2+m3);

CM_z = (m1*CM_1(3) + m2*(CM_21(3)+CM_22(3)) + m3*CM_3(3)) /
(m1+2*m2+m3);


CM_sat = [CM_x; CM_y; CM_z];

//// Momento de inércia

I_sat = I_1 + I_2 + I_3;

I1 = I_sat(1,1);

I2 = I_sat(2,2);
```

```
I3 = I_sat(3,3);


//// Definição dos arquivos de saída - Matriz inercia com nomes

Inercia = mopen('Momento_inercia.data','w');

mfprintf(Inercia,'Ixx = %f \n',I1)

mfprintf(Inercia,'Iyy = %f \n',I2)

mfprintf(Inercia,'Izz = %f \n',I3)

mclose(Inercia)

// Definição dos arquivos de saída - Matriz inercia sem nomes (apenas
numeros)

Inercia_num = mopen('Momento_inercia_num.data','w');

mfprintf(Inercia_num,'%f \n',I1)

mfprintf(Inercia_num,'%f \n',I2)

mfprintf(Inercia_num,'%f \n',I3)

mclose(Inercia_num)


// Vetor centro de massa

CM = mopen('Centro_massa.data','w');

mfprintf(CM,'CM_sat_x = %f \n',CM_x)

mfprintf(CM,'CM_sat_y = %f \n',CM_y)

mfprintf(CM,'CM_sat_z = %f \n',CM_z)

mclose(CM)


CM_num = mopen('Centro_massa_num.data','w');

mfprintf(CM_num,' %f \n',CM_x)

mfprintf(CM_num,' %f \n',CM_y)

mfprintf(CM_num,' %f \n',CM_z)

mclose(CM_num)
```

parametric_study.m

```
%% PARAMETER VARIATION

clc

%% LEITURA DOS ARQUIVOS DE PARÂMETROS DE ENTRADA

param_in = fopen('Input_parameters_num_start.data');

p1 = fscanf(param_in,'%f \n',[14,inf]);

fclose(param_in);

delta1 = fopen('delta_param_min.data');

p21 = fscanf(delta1,'%f \n',[14,1]);

fclose(delta1);

delta2 = fopen('delta_param_med.data');

p22 = fscanf(delta2,'%f \n',[14,1]);

fclose(delta2);

delta3 = fopen('delta_param_max.data');

p23 = fscanf(delta3,'%f \n',[14,1]);

fclose(delta3);

rodada = fopen('rodada.data');

p3 = fscanf(rodada,'%f');

fclose(rodada);

a = p1(1);

b = p1(2);

c = p1(3);

h1 = p1(4);

h2 = p1(5);

e1 = p1(6);

e2 = p1(7);

Ri = p1(8);

Re = p1(9);

R = p1(10);
```

```
L = p1(11);
ro1 = p1(12);
ro2 = p1(13);
ro3 =p1(14);
if p3 == 1
d_a = p21(1);
    d_b = p21(2);
    d_c = p21(3);
    d_h1 = p21(4);
    d_h2 = p21(5);
    d_e1 = p21(6);
    d_e2 = p21(7);
    d_Ri = p21(8);
    d_Re = p21(9);
    d_R = p21(10);
    d_L = p21(11);
    d_ro1 = p21(12);
    d_ro2 = p21(13);
    d_ro3 = p21(14);

a_out = p1(1)+ p21(1);
    b_out = p1(2) + p21(2);
    c_out = p1(3) + p21(3);
    h1_out = p1(4) + p21(4);
    h2_out = p1(5) + p21(5);
    e1_out = p1(6) + p21(6);
    e2_out = p1(7) + p21(7);
    Ri_out = p1(8) + p21(8);
    Re_out = p1(9) + p21(9);
```

```
    R_out = p1(10) + p21(10);
    L_out = p1(11) + p21(11);
    ro1_out = p1(12) + p21(12);
    ro2_out = p1(13) + p21(13);
    ro3_out = p1(14) + p21(14);

    param_out = fopen('Input_parameters_num.data','w');

    fprintf(param_out,'%f \n',a_out)
    fprintf(param_out,'%f \n',b_out)
    fprintf(param_out,'%f \n',c_out)
    fprintf(param_out,'%f \n',h1_out)
    fprintf(param_out,'%f \n',h2_out)
    fprintf(param_out,'%f \n',e1_out)
    fprintf(param_out,'%f \n',e2_out)
    fprintf(param_out,'%f \n',Ri_out)
    fprintf(param_out,'%f \n',Re_out)
    fprintf(param_out,'%f \n',R_out)
    fprintf(param_out,'%f \n',L_out)
    fprintf(param_out,'%f \n',ro1_out)
    fprintf(param_out,'%f \n',ro2_out)
    fprintf(param_out,'%f \n',ro3_out)

    fclose(param_out)

else
    if p3 == 2
d_a = p22(1);
        d_b = p22(2);
```

```
        d_c = p22(3);
        d_h1 = p22(4);
        d_h2 = p22(5);
        d_e1 = p22(7);
        d_e2 = p22(8);
        d_Ri = p22(9);
        d_Re = p22(9);
        d_R = p22(10);
        d_L = p22(11);
        d_ro1 = p22(12);
        d_ro2 = p22(13);
        d_ro3 = p22(14);

a_out = p1(1)+ p22(1);
        b_out = p1(2) + p22(2);
        c_out = p1(3) + p22(3);
        h1_out = p1(4) + p22(4);
        h2_out = p1(5) + p22(5);
        e1_out = p1(6) + p22(6);
        e2_out = p1(7) + p22(7);
        Ri_out = p1(8) + p22(8);
        Re_out = p1(9) + p22(9);
        R_out = p1(10) + p22(10);
        L_out = p1(11) + p22(11);
        ro1_out = p1(12) + p22(12);
        ro2_out = p1(13) + p22(13);
        ro3_out = p1(14) + p22(14);

        param_out = fopen('Input_parameters_num.data','w');
```

```
        fprintf(param_out,'%f \n',a_out)
        fprintf(param_out,'%f \n',b_out)
        fprintf(param_out,'%f \n',c_out)
        fprintf(param_out,'%f \n',h1_out)
        fprintf(param_out,'%f \n',h2_out)
        fprintf(param_out,'%f \n',e1_out)
        fprintf(param_out,'%f \n',e2_out)
        fprintf(param_out,'%f \n',Ri_out)
        fprintf(param_out,'%f \n',Re_out)
        fprintf(param_out,'%f \n',R_out)
        fprintf(param_out,'%f \n',L_out)
        fprintf(param_out,'%f \n',ro1_out)
        fprintf(param_out,'%f \n',ro2_out)
        fprintf(param_out,'%f \n',ro3_out)

        fclose(param_out)

    else
        if p3 == 3
d_a = p23(1);
        d_b = p23(2);
        d_c = p23(3);
        d_h1 = p23(4);
        d_h2 = p23(5);
        d_e1 = p23(7);
        d_e2 = p23(8);
        d_Ri = p23(9);
        d_Re = p23(9);
```

```
        d_R = p23(10);
        d_L = p23(11);
        d_ro1 = p23(12);
        d_ro2 = p23(13);
        d_ro3 = p23(14);

a_out = p1(1)+ p23(1);
        b_out = p1(2) + p23(2);
        c_out = p1(3) + p23(3);
        h1_out = p1(4) + p23(4);
        h2_out = p1(5) + p23(5);
        e1_out = p1(6) + p23(6);
        e2_out = p1(7) + p23(7);
        Ri_out = p1(8) + p23(8);
        Re_out = p1(9) + p23(9);
        R_out = p1(10) + p23(10);
        L_out = p1(11) + p23(11);
        ro1_out = p1(12) + p23(12);
        ro2_out = p1(13) + p23(13);
        ro3_out = p1(14) + p23(14);

        param_out = fopen('Input_parameters_num.data','w');

        fprintf(param_out,'%f \n',a_out)
        fprintf(param_out,'%f \n',b_out)
        fprintf(param_out,'%f \n',c_out)
        fprintf(param_out,'%f \n',h1_out)
        fprintf(param_out,'%f \n',h2_out)
        fprintf(param_out,'%f \n',e1_out)
```

```
        fprintf(param_out,'%f \n',e2_out)
        fprintf(param_out,'%f \n',Ri_out)
        fprintf(param_out,'%f \n',Re_out)
        fprintf(param_out,'%f \n',R_out)
        fprintf(param_out,'%f \n',L_out)
        fprintf(param_out,'%f \n',ro1_out)
        fprintf(param_out,'%f \n',ro2_out)
        fprintf(param_out,'%f \n',ro3_out)

        fclose(param_out)
        end
    end
end
```

liquid_properties.m

```
%% CÁLCULO DE PROPRIEDADES INERCIAIS DO PROPELENTE %%
% dm_dt_tank = fopen('dm.data');
% I = fscanf(dm_dt_tank,'%f',[inf,1]);
% fclose(dm_dt_tank);
clc
%% Leitura do arquivo de vazão mássica (dm/dt) vindo do modelo
thruster.ame (AMESim)
format long
load m1_dot.data
load m2_dot.data
load m3_dot.data
load m4_dot.data
load m5_dot.data
load m6_dot.data
for i=1:size(m1_dot,1)
```

```
    dm1_dt(i) = m1_dot(i,2);

    dm2_dt(i) = m2_dot(i,2);

    dm3_dt(i) = m3_dot(i,2);

    dm4_dt(i) = m4_dot(i,2);

dm5_dt(i) = m5_dot(i,2);

    dm6_dt(i) = m6_dot(i,2);

    time(i) = i/10;

end

%% Determinação do momento de inércia Izz do líquido (propelente) em
função da redução de massa do mesmo

R = 0.29;                       % diâmetro interno do tanque Ri [m]
0.29

%% Propriedades do propelente

ro = 1397;                         % densidade do propelente [kg/m3]
(T = 300 K)

h = 0:2*R/(size(m1_dot,1)):2*R;  % vetor variação de altura do líquido

V_inicial = 14*1e-3;             % volume inicial de propelente [m3]

for k = 1:1:size(h')

    ZG(k) = (-3/4)*((2*R - h(k)).^2/(3*R - h(k)));

  % Vp1(k) = (4/3)*pi*R^3 - (pi/3)*((2*R - h(k)).^2)*(R - h(k));

    Vp(k) = (4/3)*pi*R^3 - (pi*R)*((2*R - h(k)).^2) + (pi/3)*((2*R -
h(k)).^3);

    Izz(k) = ro*pi*((R^4)*(h(k)+R) - (2/3)*(R^2)*(h(k).^3 + R^3) +
(1/5)*(h(k).^5 + R^5));

end



% plot(h,ZG), grid, title('Centro de massa versus nível de
propelente'), xlabel('h [m]'), ylabel('ZG [m]')

plot(h,Vp), grid, title('Volume de propelente versus nível do mesmo'),
xlabel('h [m]'), ylabel('Vp [m3]')

plot(h,Izz), grid, title('Momento de Inércia em z versus nível de
propelente'), xlabel('h [m]'), ylabel('Izz_v1 [kg.m2]')
```

```
% plot(Vp,Izz), grid, title('Momento de Inércia em z versus volume de
propelente'), xlabel('Vp [m]'), ylabel('Izz [kg.m2]')

%% Tempo de simulação

t_sim_inicial = 0;

t_sim_final = 60;

%% Execução do modelo simulink (LMP.mdl)

modelo_LMP = 'LMP';

load_system(modelo_LMP)

sim(modelo_LMP)

%% Salvar arquivos de propriedades mássicas (liq) e volume de
propelente

save Izz_liq.data Izz_liq '-ascii'

Izz_liq_final = Izz_liq(k-1);

save Izz_prop.data Izz_liq_final '-ascii'

%% Token afirmando que deve-se acionar cálculo de Izz_prop na
ferramenta ADE_param

token_mp = fopen('token_mp.data','w');

fprintf(token_mp,'%f', 1);

fclose(token_mp)

clc
```

liquid_properties_v1.m

```
%% CÁLCULO DE PROPRIEDADES INERCIAIS DO PROPELENTE %%

clc



%% Leitura do arquivo de vazão mássica (dm/dt) vindo do modelo
thruster.ame (AMESim)

format long

load m1_dot.data

load m2_dot.data

load m3_dot.data
```

```
load m4_dot.data

load m5_dot.data

load m6_dot.data

for i=1:size(m1_dot,1)

    dm1_dt(i) = m1_dot(i,2);

    dm2_dt(i) = m2_dot(i,2);

    dm3_dt(i) = m3_dot(i,2);

    dm4_dt(i) = m4_dot(i,2);

dm5_dt(i) = m5_dot(i,2);

    dm6_dt(i) = m6_dot(i,2);

    time(i) = i/10;

end

%% Determinação do momento de inércia Izz do líquido (propelente) em
função da redução de massa do mesmo

R = 0.29;                          % diâmetro interno do tanque Ri [m]

%% Propriedades do(s) propelente(s)

ro(1) = 1397;                         % densidade do propelente H2O2
[kg/m3] (T = 300 K)

% 1º elemento: Peróxido de Hidrogênio (propelente)

% O2 (produto #1)

gamma(1) = 1.393;              % relação cp/cv

n(1) = 32;                     % massa molar [g/mol]

Ru(1) = 259.8;                 % cte. universal [kJ/kg*K]

% H2O (produto #2)

gamma(2) = 1.327;

n(2) = 36;

Ru(2) = 461.5;

% 2ºelemento: Hidrazina (propelente)

% N2 (produto #1)

gamma(3) = 1.400;
```

```
n(3) = 28;

Ru(3) = 296.0;

% H2 (produto #2)

gamma(4) = 1.409;

n(4) = 2;

Ru(4) = 4124.0;


%% Dados iniciais de operação

h = 0:2*R/(size(m1_dot,1)):2*R;  % vetor variação de altura do líquido

V_inicial = 14*1e-3;             % volume inicial de propelente [m3]

for k = 1:1:size(h')

    ZG(k) = (-3/4)*((2*R - h(k)).^2/(3*R - h(k)));

  % Vp1(k) = (4/3)*pi*R^3 - (pi/3)*((2*R - h(k)).^2)*(R - h(k));

    Vp(k) = (4/3)*pi*R^3 - (pi*R)*((2*R - h(k)).^2) + (pi/3)*((2*R -
h(k)).^3);

    Izz(k) = ro*pi*((R^4)*(h(k)+R) - (2/3)*(R^2)*(h(k).^3 + R^3) +
(1/5)*(h(k).^5 + R^5));

end


% plot(h,ZG), grid, title('Centro de massa versus nível de
propelente'), xlabel('h [m]'), ylabel('ZG [m]')

plot(h,Vp), grid, title('Volume de propelente versus nível do mesmo'),
xlabel('h [m]'), ylabel('Vp [m3]')

plot(h,Izz), grid, title('Momento de Inércia em z versus nível de
propelente'), xlabel('h [m]'), ylabel('Izz_v1 [kg.m2]')


% plot(Vp,Izz), grid, title('Momento de Inércia em z versus volume de
propelente'), xlabel('Vp [m]'), ylabel('Izz [kg.m2]')


%% Tempo de simulação

t_sim_inicial = 0;

t_sim_final = 60;
```

```
% save Izz_liq.data Izz_liq 'ascii'

% save u1.data u1 '-ascii'

%% Execução do modelo simulink (LMP.mdl)

modelo_LMP = 'LMP';

load_system(modelo_LMP)

sim(modelo_LMP)

%% Salvar arquivos de propriedades mássicas (liq) e volume de
propelente

save Izz_liq.data Izz_liq '-ascii'

Izz_liq_final = Izz_liq(k-1);

save Izz_prop.data Izz_liq_final '-ascii'

%% Token afirmando que deve-se acionar cálculo de Izz_prop na
ferramenta ADE_param

token_mp = fopen('token_mp.data','w');

fprintf(token_mp,'%f', 1);

fclose(token_mp)

%% Salvar arquivo com dados a serem usador por modelo do propulsor
(Thruster.ame / ferramenta PROP_2 + Thruster_param)

thruster_gp1 = fopen('gas_properties_1.data','w');

fprintf(thruster_gp1,'%f \n', gamma(1));

fprintf(thruster_gp1,'%f \n', Ru(1));

fprintf(thruster_gp1,'%f \n', n(1));

fprintf(thruster_gp1,'%f \n', gamma(2));

fprintf(thruster_gp1,'%f \n', Ru(2));

fprintf(thruster_gp1,'%f \n', n(2));

fclose(thruster_gp1)

thruster_gp11 = fopen('gas_properties_1_grandezas.data','w');

fprintf(thruster_gp11,'gamma_11 = %f \n', gamma(1));

fprintf(thruster_gp11,'R_11 = %f \n', Ru(1));

fprintf(thruster_gp11,'n_11 = %f \n', n(1));
```

```
fprintf(thruster_gp11,'gamma_12 = %f \n', gamma(2));
fprintf(thruster_gp11,'R_12 = %f \n', Ru(2));
fprintf(thruster_gp11,'n_12 = %f \n', n(2));
fclose(thruster_gp11)

thruster_gp2 = fopen('gas_properties_2.data','w');
fprintf(thruster_gp2,'%f \n', gamma(3));
fprintf(thruster_gp2,'%f \n', Ru(3));
fprintf(thruster_gp2,'%f \n', n(3));
fprintf(thruster_gp2,'%f \n', gamma(4));
fprintf(thruster_gp2,'%f \n', Ru(4));
fprintf(thruster_gp2,'%f \n', n(4));
fclose(thruster_gp2)

thruster_gp22 = fopen('gas_properties_2_grandezas.data','w');
fprintf(thruster_gp22,'gamma_11 = %f \n', gamma(3));
fprintf(thruster_gp22,'R_11 = %f \n', Ru(3));
fprintf(thruster_gp22,'n_11 = %f \n', n(3));
fprintf(thruster_gp22,'gamma_12 = %f \n', gamma(4));
fprintf(thruster_gp22,'R_12 = %f \n', Ru(4));
fprintf(thruster_gp22,'n_12 = %f \n', n(4));
fclose(thruster_gp22)
```

Gains_K_Satellite.m

```
%% CONTROLE DE ATITUDE DO SATÉLITE - DETERMINAÇÃO DOS GANHOS K E Ki
clc
%% LEITURA DO ARQUIVO: MOMENTOS DE INÉRCIA PRINCIPAIS (Ixx, Iyy, Izz)
Inercia = fopen('Momento_inercia_num.data');
I = fscanf(Inercia,'%f',[3,inf]);
```

```
fclose(Inercia);

%% LEITURA DO ARQUIVO: CONDIÇÕES DO AMBIENTE (atmosfera, freq.
orbital)

Ambiente = fopen('Ambiente_num.data');

env = fscanf(Ambiente,'%f',[6,inf]);

fclose(Ambiente);

%% ESCREVER Ixx, Iyy e Izz (estrutura)

I1 = I(1);

I2 = I(2);

I3 = I(3);

% Distâncias entre CM e propulsores

d1 = .2;    d2 = .2;    d3 = .2;

%% ESCREVER DADOS DO AMBIENTE

n = env(1);  ro_amb = env(2);  V_sat = env(3);

Cd = env(4);  U_Scp = env(5);  S = env(6);


%% ENTRADAS E PARÂMETRO DE SIMULAÇÃO

%% MANOBRA ou REFERÊNCIA: phi, theta, psi (x1_ref ,x2_ref ,x3_ref)

t_phi = 20;

phi_inicial = 0;

phi_final = 10;

t_theta = 20;

theta_inicial = 0;

theta_final = 0;

t_psi = 20;

psi_inicial = 0;

psi_final = 0;


%% PARÂMETROS DE SIMULAÇÃO E GERAÇÃO DE ARQUIVO DE DADOS
CORRESPONDENTE (sim_param.data)
```

```
%% Parâmetros principais
format
t_sim_inicial = 0;          % tempo inicial            1
t_sim_final = 60;           % tempo final              2
format bank
print_interval = 0.1;       % intervalo de impressão   3
format shortEng
max_time_step = 1e30;       % máximo passo (tempo)     4
tolerance = 1e-5;           % tolerância               5
format short
fixed_step = 0.001;         % passo (fixo)             6
format
order = 4;                  % ordem                    7
% Simplificação
format
pp1 = t_sim_inicial;
pp2 = t_sim_final;
format bank
pp3 = print_interval;
format shortEng
pp4 = max_time_step;
pp5 = tolerance;

format short
pp6 = fixed_step;
format
pp7 = order;
%% Parâmetros secundários
```

```
error_ctrl = 0;                % controle do erro (0: misto / 1:
relativo / 2: absoluto)

monitor_time = 0;              % monitoração do tempo (0: sem / 2:
com)

disc_printouts = 0;            % impressão discontinuidades (0: sem
extras / 1: com extras)

run_stats = 1;                 % estatísticas das corridas (0: sem /
1: com)

run_types = 72;                % tipos de corridas (Bit0, Bit1, Bit2,
Bit3, Bit4, Bit5, Bit6, Bit7) ... especificar

solver_type = 0;               % método (0: convencional / 1:
conservador)

opts_stabilizing_runs = 0;     % opções para corridas estabilizadas
(Bit0, Bit1)

disc_handling = 0;             % lidar com descontinuidades (0:
normal('default') / 1: mínimo)

activity_index_calc = 0;       % cálculo do índice de atividade (0:
desligado/ 1: ligado)



% Simplificação

ps1 = error_ctrl;

ps2 = monitor_time;

ps3 = disc_printouts;

ps4 = run_stats;

ps5 = run_types;

ps6 = solver_type;

ps7 = opts_stabilizing_runs;

ps8 = disc_handling;

ps9 = activity_index_calc;

%% GERAÇÃO DO ARQUIVO DE SIMULAÇÃO PARA FERRAMENTAS: ADE (ADE.ame) e
PROP (Propulsion_sys.ame)

sim_param = fopen('ADE_.sim','w');

fprintf(sim_param,'%d %d %f %d %d %f %d \n', pp1, pp2, pp3, pp4, pp5,
pp6, pp7);
```

```
fprintf(sim_param,'%d %d %d %d %d %d %d %d %d', ps1, ps2, ps3, ps4,
ps5, ps6, ps7, ps8, ps9);

fclose(sim_param)

sim_param_2 = fopen('Propulsion_sys_.sim','w');

fprintf(sim_param_2,'%d %d %f %d %d %f %d \n', pp1, pp2, pp3, pp4,
pp5, pp6, pp7);

fprintf(sim_param_2,'%d %d %d %d %d %d %d %d %d', ps1, ps2, ps3, ps4,
ps5, ps6, ps7, ps8, ps9);

fclose(sim_param_2)

sim_param_3 = fopen('Thruster_.sim','w');

fprintf(sim_param_3,'%d %d %f %d %d %f %d \n', pp1, pp2, pp3, pp4,
pp5, pp6, pp7);

fprintf(sim_param_3,'%d %d %d %d %d %d %d %d %d', ps1, ps2, ps3, ps4,
ps5, ps6, ps7, ps8, ps9);

fclose(sim_param_3)

format short

%%  MAZTRIZES DE ESTADO (APENAS SATÉLITE = PLANTA S/ ATUADOR)

% A 6x6  B 6x6  C 6x6   D 6x6

A = [0                       0                    0
1                       0              0  ;

     0                       0                    0
0                       1              0  ;

     0                       0                    0
0                       0              1  ;

     -4*n^2*(I2-I3)/I1       0                    0
0                       0              n*(I1-I2+I3)/I1 ;

     0                       -3*n^2*(I1-I3)/I2    0
0                       0              0  ;

     0                       0                    -n^2*(I2-I1)/I3
-n*(I1-I2+I3)/I3        0              0 ];

B =  [0           0            0          0          0           0;

      0           0            0          0          0           0;

      0           0            0          0          0           0;

     d1/I1       d1/I1         0          0          0           0;
```

```
      0           0            d2/I2      d2/I2       0           0;

      0           0             0          0         d3/I3
d3/I3];

C = [1 0 0 0 0 0;

     0 1 0 0 0 0;

     0 0 1 0 0 0];

C_ss = eye(6);

D = zeros(6,3);


%% MATRIZES AUMENTADAS LQR

AA = [A zeros(6,3); -C zeros(3,3)];

BB = [B; zeros(3,6)];

CC = eye(9);

DD = zeros(9,3);


% Valores máximos para estados

phi_max = 30/pi;

theta_max = 30/pi;

psi_max = 30/pi;

phi_dot_max = 0.1/pi;

theta_dot_max = 0.1/pi;

psi_dot_max = 0.1/pi;

m_dot_max = 2.5E-3;


% Matrizes peso da função custo J (Q e R). Baseado em KUGA H.K. e
GADELHA, L.C.

Q1 = 7.7; Q2 = 1; Rn = 1;

Q = [Q1  0  0  0  0  0  0  0  0;

     0  Q1  0  0  0  0  0  0  0;

     0   0 Q1  0  0  0  0  0  0;
```

```
     0   0  0  Q2 0  0  0  0  0;

     0   0  0  0  Q2 0  0  0  0;

     0   0  0  0  0  Q2 0  0  0;

     0   0  0  0  0  0  Q2 0  0;

     0   0  0  0  0  0  0  Q2 0;

  0   0  0  0  0  0  0  0  Q2];

R = Rn*eye(6);


%% DETERMINAÇÃO DOS GANHOS {K} e {Ki}

KK = lqr(AA,BB,Q,R);

ACL = AA - BB*KK;


%% ESPECIFICAÇÃO DOS GANHOS DE ESTADO

k1 = KK(1,1); k2 = KK(1,2); k3 = KK(1,3); k4 = KK(1,4); k5 = KK(1,5);
k6 = KK(1,6);

k7 = KK(2,1); k8 = KK(2,2); k9 = KK(2,3); k10 = KK(2,4); k11 =
KK(2,5); k12 = KK(2,6);

k13 = KK(3,1); k14 = KK(3,2); k15 = KK(3,3); k16 = KK(3,4); k17 =
KK(3,5); k18 = KK(3,6);

k19 = KK(4,1); k20 = KK(4,2); k21 = KK(4,3); k22 = KK(4,4); k23 =
KK(4,5); k24 = KK(4,6);

k25 = KK(5,1); k26 = KK(5,2); k27 = KK(5,3); k28 = KK(5,4); k29 =
KK(5,5); k30 = KK(5,6);

k31 = KK(6,1); k32 = KK(6,2); k33 = KK(6,3); k34 = KK(6,4); k35 =
KK(6,5); k36 = KK(6,6);


K1 = [k1 k2 k3 k4 k5 k6];

K2 = [k7 k8 k9 k10 k11 k12];

K3 = [k13 k14 k15 k16 k17 k18];

K4 = [k19 k20 k21 k22 k23 k24];

K5 = [k25 k26 k27 k28 k29 k30];

K6 = [k31 k32 k33 k34 k35 k36];
```

```
K = [K1; K2; K3; K4; K5; K6];


Ki = abs([KK(1,7) KK(1,8)  KK(1,9);

          KK(2,7) KK(2,8)  KK(2,9);

          KK(3,7) KK(3,8)  KK(3,9);

          KK(4,7) KK(4,8)  KK(4,9);

          KK(5,7) KK(5,8)  KK(5,9);

KK(6,7) KK(6,8)  KK(6,9)]);


%% DEFINIÇÃO DOS ARQUIVOS DE SAÍDA

Ganhos_K = fopen('Ganhos_K.data','w');

fprintf(Ganhos_K,'K11 = %f \n K12 = %f \n K13 = %f \n K14 = %f \n K15
= %f \n K16 = %f \n', K(1,1), K(1,2), K(1,3), K(1,4), K(1,5), K(1,6))

fprintf(Ganhos_K,'K21 = %f \n K22 = %f \n K23 = %f \n K24 = %f \n K25
= %f \n K26 = %f \n', K(2,1), K(2,2), K(2,3), K(2,4), K(2,5), K(2,6))

fprintf(Ganhos_K,'K31 = %f \n K32 = %f \n K33 = %f \n K34 = %f \n K35
= %f \n K36 = %f \n', K(3,1), K(3,2), K(3,3), K(3,4), K(3,5), K(3,6))

fprintf(Ganhos_K,'K41 = %f \n K42 = %f \n K43 = %f \n K44 = %f \n K45
= %f \n K46 = %f \n', K(4,1), K(4,2), K(4,3), K(4,4), K(4,5), K(4,6))

fprintf(Ganhos_K,'K51 = %f \n K52 = %f \n K53 = %f \n K54 = %f \n K55
= %f \n K56 = %f \n', K(5,1), K(5,2), K(5,3), K(5,4), K(5,5), K(5,6))

fprintf(Ganhos_K,'K61 = %f \n K62 = %f \n K63 = %f \n K64 = %f \n K65
= %f \n K66 = %f \n', K(6,1), K(6,2), K(6,3), K(6,4), K(6,5), K(6,6))

fclose(Ganhos_K)


Ganhos_K_num = fopen('Ganhos_K_num.data','w');


fprintf(Ganhos_K_num,'%f %f %f %f %f %f \n', K(1,1), K(1,2), K(1,3),
K(1,4), K(1,5), K(1,6))

fprintf(Ganhos_K_num,'%f %f %f %f %f %f \n', K(2,1), K(2,2), K(2,3),
K(2,4), K(2,5), K(2,6))

fprintf(Ganhos_K_num,'%f %f %f %f %f %f \n', K(3,1), K(3,2), K(3,3),
K(3,4), K(3,5), K(3,6))
```

```
fprintf(Ganhos_K_num,'%f %f %f %f %f %f \n', K(4,1), K(4,2), K(4,3),
K(4,4), K(4,5), K(4,6))

fprintf(Ganhos_K_num,'%f %f %f %f %f %f \n', K(5,1), K(5,2), K(5,3),
K(5,4), K(5,5), K(5,6))

fprintf(Ganhos_K_num,'%f %f %f %f %f %f \n', K(6,1), K(6,2), K(6,3),
K(6,4), K(6,5), K(6,6))


fclose(Ganhos_K_num)

%% CHAMAR MODELO SIMULINK PARA SIMULAÇÃO

modelo = 'Control_Satellite';

load_system(modelo)

sim(modelo)

%% SALVAR VETORES DE SINAL EM ARQUIVOS

% 2 colunas (tempo, variável: sinais de entrada)

save u1.data u1 '-ascii'

save u2.data u2 '-ascii'

save u3.data u3 '-ascii'

save u4.data u4 '-ascii'

save u5.data u5 '-ascii'

save u6.data u6 '-ascii'


% 1 coluna (variável): sinais de entrada

save u1_1D.data u1_1D '-ascii'

save u2_1D.data u2_1D '-ascii'

save u3_1D.data u3_1D '-ascii'

save u4_1D.data u4_1D '-ascii'

save u5_1D.data u5_1D '-ascii'

save u6_1D.data u6_1D '-ascii'


% 1 coluna (variável): estados
```

```
save x1_1D.data x1_1D '-ascii'
save x2_1D.data x2_1D '-ascii'
save x3_1D.data x3_1D '-ascii'
save x4_1D.data x4_1D '-ascii'
save x5_1D.data x5_1D '-ascii'
save x6_1D.data x6_1D '-ascii'
%% SALVAR MÓDULOS DOS TORUES MAGNÉTICOS E REGISTRAR EM ARQUIVO
% save Tm1.data Tm1 '-ascii'
% save Tm2.data Tm2 '-ascii'
Tm = fopen('Torques_magneticos.data','w');
fprintf(Tm,'%f \n', Tm1);
fprintf(Tm,'%f \n', Tm2);
fclose(Tm)

%% ESCREVER VETORES DE CONTROLE
Vector_u1 = fopen('Vector_u1.data','w');
Vector_u2 = fopen('Vector_u2.data','w');
Vector_u3 = fopen('Vector_u3.data','w');
Vector_u4 = fopen('Vector_u4.data','w');
Vector_u5 = fopen('Vector_u5.data','w');
Vector_u6 = fopen('Vector_u6.data','w');
for i=1:size(u1,1)
        fprintf(Vector_u1,'%f \n ',i , u1(i,2))
        fprintf(Vector_u2,'%f \n ',i , u2(i,2))
        fprintf(Vector_u3,'%f \n ',i , u3(i,2))
        fprintf(Vector_u4,'%f \n ',i , u4(i,2))
        fprintf(Vector_u5,'%f \n ',i , u5(i,2))
        fprintf(Vector_u6,'%f \n ',i , u6(i,2))
```

```
end


fclose(Vector_u1)

fclose(Vector_u2)

fclose(Vector_u3)

fclose(Vector_u4)

fclose(Vector_u5)

fclose(Vector_u6)


%% CRIAÇÃO DE TOKEN PARA MODELO ADE.ame

token1 = fopen('token_ADE.data','w');

fprintf(token1,'%f',1)

fclose(token1)


%% CRIAÇÃO DE TOKEN PARA ferramenta ade_PARAM

token2 = fopen('token_PROP.data','w');

fprintf(token2,'%f',0)

fclose(token2)
```

Gains_K_Satellite_and_Actuator.m

```
%% CONTROLE DE ATITUDE DO SATÉLITE E ATUADOR - DETERMINAÇÃO DOS GANHOS
K E Ki

clc

format long


%% LEITURA DO ARQUIVO: MOMENTOS DE INÉRCIA PRINCIPAIS (Ixx, Iyy, Izz)

Inercia = fopen('Momento_inercia_num.data');

I = fscanf(Inercia,'%f',[3,inf]);

fclose(Inercia);
```

```
%% LEITURA DO ARQUIVO: CONDIÇÕES DO AMBIENTE (atmosfera, freq.
orbital)

Ambiente = fopen('Ambiente_num.data');

env = fscanf(Ambiente,'%f',[6,inf]);

fclose(Ambiente);


%% ESCREVER Ixx, Iyy e Izz (estrutura)

I1 = I(1);

I2 = I(2);

I3 = I(3);

% Distâncias entre CM e propulsores

d1 = .2;    d2 = .2;    d3 = .2;

%% ESCREVER DADOS DO AMBIENTE

n = env(1);  ro_amb = env(2);  V_sat = env(3);

Cd = env(4);  U_Scp = env(5);  S = env(6);


%% ENTRADAS E PARÂMETRO DE SIMULAÇÃO

%% MANOBRA ou REFERÊNCIA: phi, theta, psi (x1_ref ,x2_ref ,x3_ref)

t_phi = 20;

phi_inicial = 0;

phi_final = 0;

t_theta = 20;

theta_inicial = 0;

theta_final = 0;

t_psi = 20;

psi_inicial = 0;

psi_final = 20;
```

```
%% PARÂMETROS DE SIMULAÇÃO E GERAÇÃO DE ARQUIVO DE DADOS
CORRESPONDENTE (sim_param.data)

%% Parâmetros principais

format

t_sim_inicial = 0;          % tempo inicial             1

t_sim_final = 60;           % tempo final               2

format bank

print_interval = 0.1;       % intervalo de impressão    3

format shortEng

max_time_step = 1e30;       % máximo passo (tempo)      4

tolerance = 1e-5;           % tolerância                5

format short

fixed_step = 0.001;         % passo (fixo)              6

format

order = 4;                  % ordem                     7


% Simplificação

format

pp1 = t_sim_inicial;

pp2 = t_sim_final;

format bank

pp3 = print_interval;

format shortEng

pp4 = max_time_step;

pp5 = tolerance;

format short

pp6 = fixed_step;

format

pp7 = order;
```

```
%% Parâmetros secundários

error_ctrl = 0;                 % controle do erro (0: misto / 1:
relativo / 2: absoluto)

monitor_time = 0;               % monitoração do tempo (0: sem / 2:
com)

disc_printouts = 0;             % impressão discontinuidades (0: sem
extras / 1: com extras)

run_stats = 1;                  % estatísticas das corridas (0: sem /
1: com)

run_types = 72;                 % tipos de corridas (Bit0, Bit1, Bit2,
Bit3, Bit4, Bit5, Bit6, Bit7) ... especificar

solver_type = 0;                % método (0: convencional / 1:
conservador)

opts_stabilizing_runs = 0;      % opções para corridas estabilizadas
(Bit0, Bit1)

disc_handling = 0;              % lidar com descontinuidades (0:
normal('default') / 1: mínimo)

activity_index_calc = 0;        % cálculo do índice de atividade (0:
desligado/ 1: ligado)



% Simplificação

ps1 = error_ctrl;

ps2 = monitor_time;

ps3 = disc_printouts;

ps4 = run_stats;

ps5 = run_types;

ps6 = solver_type;

ps7 = opts_stabilizing_runs;

ps8 = disc_handling;

ps9 = activity_index_calc;



%% GERAÇÃO DO ARQUIVO DE SIMULAÇÃO PARA FERRAMENTAS: ADE (ADE.ame) e
PROP (Propulsion_sys.ame)
```

```
sim_param = fopen('ADE_.sim','w');

fprintf(sim_param,'%d %d %f %d %d %f %d \n', pp1, pp2, pp3, pp4, pp5,
pp6, pp7);

fprintf(sim_param,'%d %d %d %d %d %d %d %d %d', ps1, ps2, ps3, ps4,
ps5, ps6, ps7, ps8, ps9);

fclose(sim_param)

sim_param_2 = fopen('Propulsion_sys_.sim','w');

fprintf(sim_param_2,'%d %d %f %d %d %f %d \n', pp1, pp2, pp3, pp4,
pp5, pp6, pp7);

fprintf(sim_param_2,'%d %d %d %d %d %d %d %d %d', ps1, ps2, ps3, ps4,
ps5, ps6, ps7, ps8, ps9);

fclose(sim_param_2)

sim_param_3 = fopen('Thruster_.sim','w');

fprintf(sim_param_3,'%d %d %f %d %d %f %d \n', pp1, pp2, pp3, pp4,
pp5, pp6, pp7);

fprintf(sim_param_3,'%d %d %d %d %d %d %d %d %d', ps1, ps2, ps3, ps4,
ps5, ps6, ps7, ps8, ps9);

fclose(sim_param_3)

%% EXECUÇÃO DO SCRIPT COM PARÂMETROS DA VÁLVULA DE CONTROLE (FCV) E
PROPULSOR (THRUSTER)

Actuator_parameters



%% MATRIZES DE ESTADO (s/ atuador)

% A 6x6  B 6x6  C 6x6   D 6x6

A = [0                        0                    0
1                        0              0  ;

     0                        0                    0
0                        1              0  ;

     0                        0                    0
0                        0              1  ;

     -4*n^2*(I2-I3)/I1        0                    0
0                        0              n*(I1-I2+I3)/I1 ;

     0                        -3*n^2*(I1-I3)/I2    0
0                        0              0  ;
```

```
     0                      0                    -n^2*(I2-I1)/I3
-n*(I1-I2+I3)/I3        0              0 ];


B =  [0           0            0          0          0           0;

      0           0            0          0          0           0;

      0           0            0          0          0           0;

     d1/I1       d1/I1         0          0          0           0;

      0           0           d2/I2      d2/I2       0           0;

      0           0            0          0         d3/I3
d3/I3];

C = [1 0 0 0 0 0;

     0 1 0 0 0 0;

     0 0 1 0 0 0];


C_ss = eye(6);

D = zeros(6,3);

% Análise de controlabilidade

% CC = ctrb(AB)

%% MATRIZES AUMENTADAS - MÉTODO LQR

AA = [A zeros(6,3); -C zeros(3,3)];

BB = [B; zeros(3,6)];

CC = eye(9);

DD = zeros(9,3);

% valores máximos para os estados

phi_max = 30/pi;

theta_max = 30/pi;

psi_max = 30/pi;

phi_dot_max = 0.1/pi;

theta_dot_max = 0.1/pi;

psi_dot_max = 0.1/pi;
```

```
m_dot_max = 2.5E-3;

% Matrizes peso da função custo J: Q e R

Q1 = 7.7;

Q2 = 1;

Rn = 1;

Q = [Q1  0  0  0  0  0  0  0  0;

     0  Q1  0  0  0  0  0  0  0;

     0   0 Q1  0  0  0  0  0  0;

     0   0  0  Q2 0  0  0  0  0;

     0   0  0  0  Q2 0  0  0  0;

     0   0  0  0  0  Q2 0  0  0;

     0   0  0  0  0  0  Q2 0  0;

     0   0  0  0  0  0  0  Q2 0;

  0   0  0  0  0  0  0  0  Q2];


R = Rn*eye(6);

%% DETERMINAÇÃO DOS GANHOS {K} e {Ki}

KK = lqr(AA,BB,Q,R);

ACL = AA - BB*KK;

%% DISCRETIZAÇÃO DOS GANHOS DE ESTADO (K)

k1 = KK(1,1); k2 = KK(1,2); k3 = KK(1,3); k4 = KK(1,4); k5 = KK(1,5);
k6 = KK(1,6);

k7 = KK(2,1); k8 = KK(2,2); k9 = KK(2,3); k10 = KK(2,4); k11 =
KK(2,5); k12 = KK(2,6);

k13 = KK(3,1); k14 = KK(3,2); k15 = KK(3,3); k16 = KK(3,4); k17 =
KK(3,5); k18 = KK(3,6);

k19 = KK(4,1); k20 = KK(4,2); k21 = KK(4,3); k22 = KK(4,4); k23 =
KK(4,5); k24 = KK(4,6);

k25 = KK(5,1); k26 = KK(5,2); k27 = KK(5,3); k28 = KK(5,4); k29 =
KK(5,5); k30 = KK(5,6);

k31 = KK(6,1); k32 = KK(6,2); k33 = KK(6,3); k34 = KK(6,4); k35 =
KK(6,5); k36 = KK(6,6);
```

```
K1 = [k1 k2 k3 k4 k5 k6];

K2 = [k7 k8 k9 k10 k11 k12];

K3 = [k13 k14 k15 k16 k17 k18];

K4 = [k19 k20 k21 k22 k23 k24];

K5 = [k25 k26 k27 k28 k29 k30];

K6 = [k31 k32 k33 k34 k35 k36];

K = [K1; K2; K3; K4; K5; K6];


%% GANHOS INTEGRAIS (Ki)

Ki = abs([KK(1,7) KK(1,8)  KK(1,9);

KK(2,7) KK(2,8)  KK(2,9);

          KK(3,7) KK(3,8)  KK(3,9);

          KK(4,7) KK(4,8)  KK(4,9);

          KK(5,7) KK(5,8)  KK(5,9);

KK(6,7) KK(6,8)  KK(6,9)]);


%% MODElO (LINEAR) DO ATUADOR - espaço de estados

% Equações de estado

% Eletromecânico: m_c*x'' + b*x' + k*x = Fm

% Hidráulico:     m' = Kv*x    (Kv = pi*Cd*D*sqrt(2*ro*dp)

% Termodinâmico:  F = m'*Ve

% Estados

% x1 = x      x1'=x2

% x2 = x'     x2' = (Fm/m_c) - (b/m_c)*x2 - (k/m_c)*x1

% x3 = m      x3' = Kv*x1


A_p = [  0      1      0;

       -k/mc   -b/mc   0;

         Kv     0      0];
```

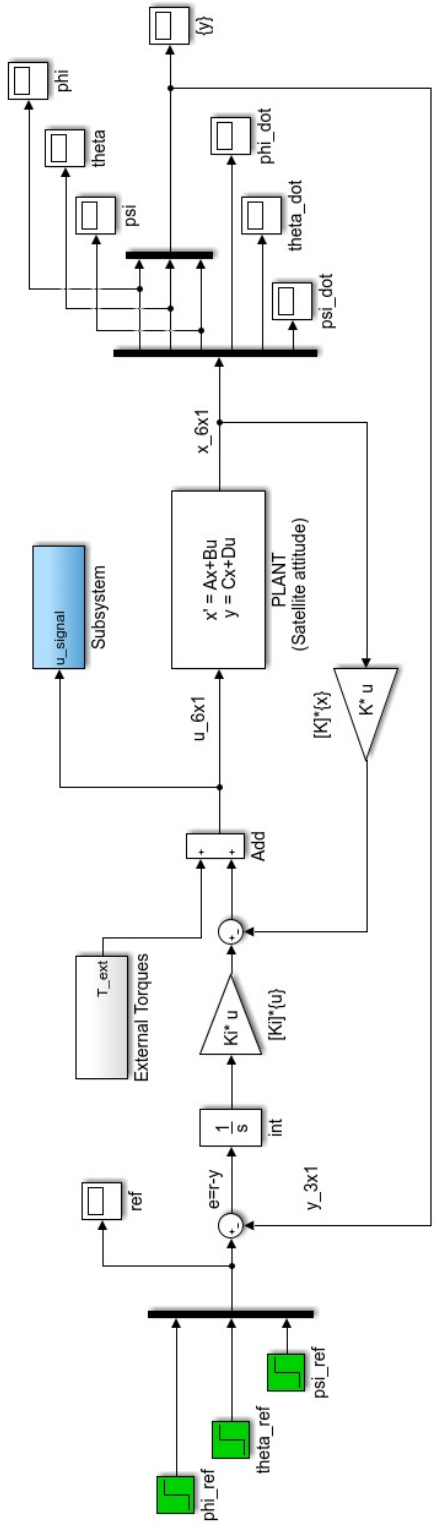
```
B_p_roll = roll*[0; 1/mc; 0];

B_p_pitch = pitch*[0; 1/mc; 0];

B_p_yaw = yaw*[0; 1/mc; 0];

C_p_roll = [Kv*Ve_roll  0  0];

C_p_pitch = [Kv*Ve_pitch   0   0];

C_p_yaw = [Kv*Ve_yaw   0   0];

D_p = [0];


%% GERAÇÃO DOS ARQUIVOS DE SAÍDA

Ganhos_K = fopen('Ganhos_K.data','w');

fprintf(Ganhos_K,'K11 = %f \n K12 = %f \n K13 = %f \n K14 = %f \n K15
= %f \n K16 = %f \n', K(1,1), K(1,2), K(1,3), K(1,4), K(1,5), K(1,6))

fprintf(Ganhos_K,'K21 = %f \n K22 = %f \n K23 = %f \n K24 = %f \n K25
= %f \n K26 = %f \n', K(2,1), K(2,2), K(2,3), K(2,4), K(2,5), K(2,6))

fprintf(Ganhos_K,'K31 = %f \n K32 = %f \n K33 = %f \n K34 = %f \n K35
= %f \n K36 = %f \n', K(3,1), K(3,2), K(3,3), K(3,4), K(3,5), K(3,6))

fprintf(Ganhos_K,'K41 = %f \n K42 = %f \n K43 = %f \n K44 = %f \n K45
= %f \n K46 = %f \n', K(4,1), K(4,2), K(4,3), K(4,4), K(4,5), K(4,6))

fprintf(Ganhos_K,'K51 = %f \n K52 = %f \n K53 = %f \n K54 = %f \n K55
= %f \n K56 = %f \n', K(5,1), K(5,2), K(5,3), K(5,4), K(5,5), K(5,6))

fprintf(Ganhos_K,'K61 = %f \n K62 = %f \n K63 = %f \n K64 = %f \n K65
= %f \n K66 = %f \n', K(6,1), K(6,2), K(6,3), K(6,4), K(6,5), K(6,6))

fclose(Ganhos_K)


Ganhos_K_num = fopen('Ganhos_K_num.data','w');


fprintf(Ganhos_K_num,'%f %f %f %f %f %f \n', K(1,1), K(1,2), K(1,3),
K(1,4), K(1,5), K(1,6))

fprintf(Ganhos_K_num,'%f %f %f %f %f %f \n', K(2,1), K(2,2), K(2,3),
K(2,4), K(2,5), K(2,6))

fprintf(Ganhos_K_num,'%f %f %f %f %f %f \n', K(3,1), K(3,2), K(3,3),
K(3,4), K(3,5), K(3,6))
```

```
fprintf(Ganhos_K_num,'%f %f %f %f %f %f \n', K(4,1), K(4,2), K(4,3),
K(4,4), K(4,5), K(4,6))

fprintf(Ganhos_K_num,'%f %f %f %f %f %f \n', K(5,1), K(5,2), K(5,3),
K(5,4), K(5,5), K(5,6))

fprintf(Ganhos_K_num,'%f %f %f %f %f %f \n', K(6,1), K(6,2), K(6,3),
K(6,4), K(6,5), K(6,6))

fclose(Ganhos_K_num)


%% CHAMAR MODELO SIMULINK PARA SIMULAÇÃO

modelo = 'Control_Satellite_and_Actuator';

load_system(modelo)

sim(modelo)

%% SALVAR ARQUIVOS

% Sinal de entrada (atuador - força magnética)

save u1.data u1 '-ascii'

save u2.data u2 '-ascii'

save u3.data u3 '-ascii'

save u4.data u4 '-ascii'

save u5.data u5 '-ascii'

save u6.data u6 '-ascii'

% 1 coluna (variável): sinais de entrada

save u1_1D.data u1_1D '-ascii'

save u2_1D.data u2_1D '-ascii'

save u3_1D.data u3_1D '-ascii'

save u4_1D.data u4_1D '-ascii'

save u5_1D.data u5_1D '-ascii'

save u6_1D.data u6_1D '-ascii'


% 1 coluna (variável): estados

save x1_acs.data x1_acs '-ascii'
```

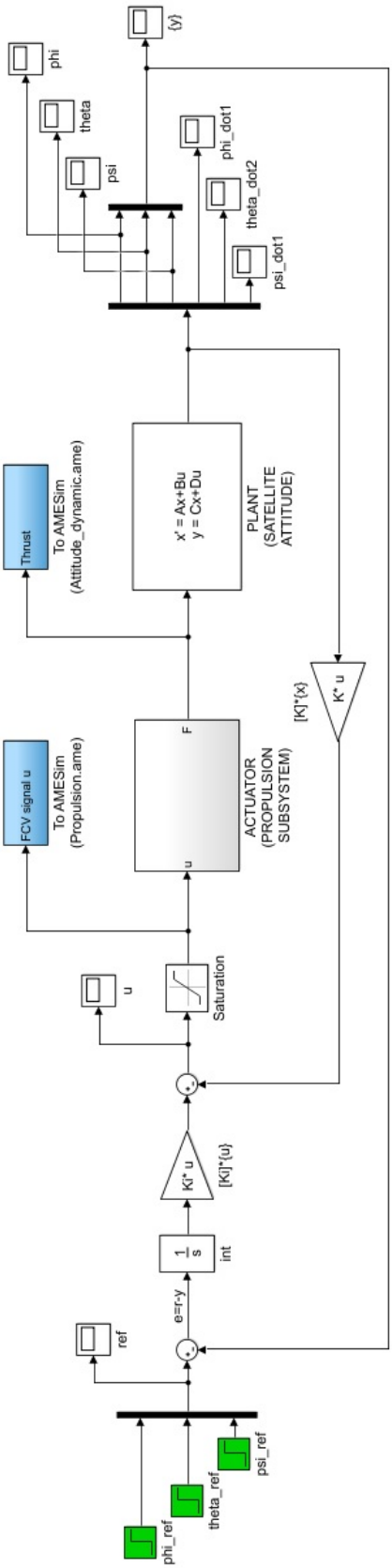
```
save x2_acs.data x2_acs '-ascii'
save x3_acs.data x3_acs '-ascii'
save x4_acs.data x4_acs '-ascii'
save x5_acs.data x5_acs '-ascii'
save x6_acs.data x6_acs '-ascii'

% Sinal de entrada satélite (planta - empuxo)
save Thrust_1_acs.data Thrust_1_acs '-ascii'
save Thrust_2_acs.data Thrust_2_acs '-ascii'
save Thrust_3_acs.data Thrust_3_acs '-ascii'
save Thrust_4_acs.data Thrust_4_acs '-ascii'
save Thrust_5_acs.data Thrust_5_acs '-ascii'
save Thrust_6_acs.data Thrust_6_acs '-ascii'

% Sinal de entrada satélite 1 coluna (planta - empuxo)
save Empuxo_1_1D.data Empuxo_1_1D '-ascii'
save Empuxo_2_1D.data Empuxo_2_1D '-ascii'
save Empuxo_3_1D.data Empuxo_3_1D '-ascii'
save Empuxo_4_1D.data Empuxo_4_1D '-ascii'
save Empuxo_5_1D.data Empuxo_5_1D '-ascii'
save Empuxo_6_1D.data Empuxo_6_1D '-ascii'

%% CRIAÇÃO DE TOKEN PARA FERRAMENTA ADE_load
token = fopen('token_ADE.data','w');
fprintf(token,'%f',1);
fclose(token)

%% CRIAÇÃO DE TOKEN PARA FERRAMENTA PROP_load
token2 = fopen('token_PROP.data','w');
```

```
fprintf(token2,'%f',1);
fclose(token2)
```

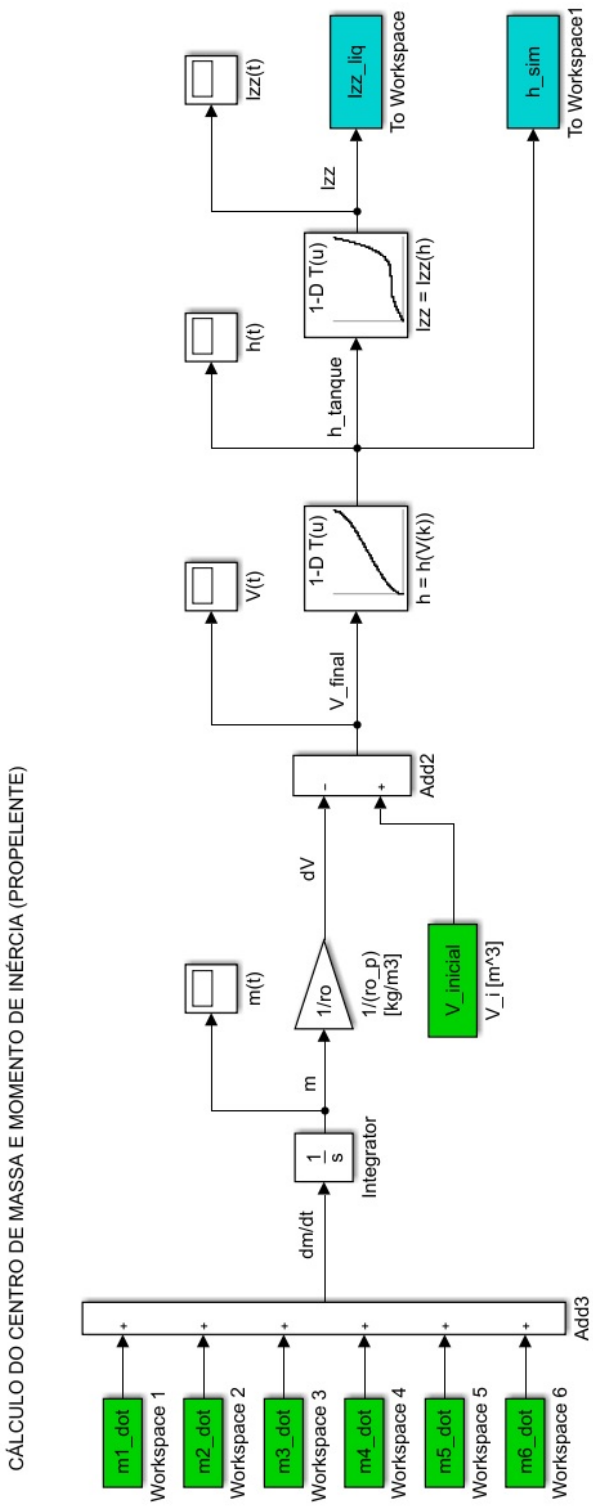
APÊNDICE D – DIAGRAMAS DOS MODELOS DE SIMULAÇÃO



Control_Satellite.mdl



Control_Satellite_and_Actuator.mdl


CÁLCULO DO CENTRO DE MASSA E MOMENTO DE INÉRCIA (PROPELENTE)
m1_dot
Workspace 1
m2_dot
Workspace 2
m3_dot
Workspace 3
m4_dot
Workspace 4
m5_dot
Workspace 5
m6_dot
Workspace 6
Add3
dm/dt
1
s
Integrator
m
m(t)
1/ro
1/(ro_p)
[kg/m3]
V_inicial
V_i [m^3]
dV
Add2
V_final
V(t)
1-D T(u)
h = h(V(k))
h_tanque
h(t)
1-D T(u)
Izz = Izz(h)
Izz
Izz(t)
Izz_liq
To Workspace
h_sim
To Workspace1

LMP.mdl

APÊNDICE E – CÓDIGOSPYTHON PRÉ, PÓS E DE EXECUÇÃO EM RCE

Ferramenta ACS_v1

Pré-execução

```
import os
shutil.copy("${in:Gains_K_Satellite}", "${dir:working}/")
shutil.copy("${in:Control_Satellite}", "${dir:working}/")

shutil.copy("${in:Momento_inercia}", "${dir:working}/")
shutil.copy("${in:Ambiente_num}", "${dir:working}/")
shutil.copy("${in:Ambiente_grandeza_unidade}", "${dir:working}/")
```

Execução

matlab.exe -nodisplay -nosplash -nodesktop -r run('Gains_K_Satellite.m');exit

Pós-execução

```
import os.path
import time

while not os.path.exists("${dir:working}/u1.data"):
      time.sleep(1)
shutil.copy("${dir:working}/u1.data", "${dir:output}/u1.data")
${out:u1} = "${dir:output}/u1.data"

while not os.path.exists("${dir:working}/u2.data"):
      time.sleep(1)
shutil.copy("${dir:working}/u2.data", "${dir:output}/u2.data")
${out:u2} = "${dir:output}/u2.data"

while not os.path.exists("${dir:working}/u3.data"):
      time.sleep(1)
shutil.copy("${dir:working}/u3.data", "${dir:output}/u3.data")
${out:u3} = "${dir:output}/u3.data"

while not os.path.exists("${dir:working}/u4.data"):
      time.sleep(1)
shutil.copy("${dir:working}/u4.data", "${dir:output}/u4.data")
${out:u4} = "${dir:output}/u4.data"

while not os.path.exists("${dir:working}/u5.data"):
      time.sleep(1)
shutil.copy("${dir:working}/u5.data", "${dir:output}/u5.data")
${out:u5} = "${dir:output}/u5.data"

while not os.path.exists("${dir:working}/u6.data"):
      time.sleep(1)
shutil.copy("${dir:working}/u6.data", "${dir:output}/u6.data")
${out:u6} = "${dir:output}/u6.data"


while not os.path.exists("${dir:working}/token_ADE.data"):
      time.sleep(1)
```

```
shutil.copy("${dir:working}/token_ADE.data",
"${dir:output}/token_ADE.data")
${out:token_ADE} = "${dir:output}/token_ADE.data"

while not os.path.exists("${dir:working}/token_PROP.data"):
      time.sleep(1)
shutil.copy("${dir:working}/token_PROP.data",
"${dir:output}/token_PROP.data")
${out:token_PROP} = "${dir:output}/token_PROP.data"


while not os.path.exists("${dir:working}/ADE_.sim"):
      time.sleep(1)
shutil.copy("${dir:working}/ADE_.sim", "${dir:output}/ADE_.sim")
${out:sim_param_ADE} = "${dir:output}/ADE_.sim"
```

Ferramenta ACS_v2

Pré-execução

```
import os

shutil.copy("${in:Gains_K_Satellite_and_Actuator}", "${dir:working}/")

shutil.copy("${in:Control_Satellite_and_Actuator}", "${dir:working}/")

shutil.copy("${in:Actuator_parameters}", "${dir:working}/")

shutil.copy("${in:Momento_inercia}", "${dir:working}/")

shutil.copy("${in:Ambiente_num}", "${dir:working}/")

shutil.copy("${in:Ambiente_grandeza_unidade}", "${dir:working}/")
```

Execução

```
matlab.exe        -nodisplay        -nosplash        -nodesktop        -r
run('Gains_K_Satellite_and_Actuator.m');exit
```

Pós-execução

```
import os.path

import time


while not os.path.exists("${dir:working}/u1.data"):

      time.sleep(1)
```

```
shutil.copy("${dir:working}/u1.data", "${dir:output}/u1.data")
${out:u1} = "${dir:output}/u1.data"

while not os.path.exists("${dir:working}/u2.data"):
      time.sleep(1)
shutil.copy("${dir:working}/u2.data", "${dir:output}/u2.data")
${out:u2} = "${dir:output}/u2.data"

while not os.path.exists("${dir:working}/u3.data"):
      time.sleep(1)
shutil.copy("${dir:working}/u3.data", "${dir:output}/u3.data")
${out:u3} = "${dir:output}/u3.data"

while not os.path.exists("${dir:working}/u4.data"):
      time.sleep(1)
shutil.copy("${dir:working}/u4.data", "${dir:output}/u4.data")
${out:u4} = "${dir:output}/u4.data"

while not os.path.exists("${dir:working}/u5.data"):
      time.sleep(1)
shutil.copy("${dir:working}/u5.data", "${dir:output}/u5.data")
${out:u5} = "${dir:output}/u5.data"

while not os.path.exists("${dir:working}/u6.data"):
      time.sleep(1)
shutil.copy("${dir:working}/u6.data", "${dir:output}/u6.data")
${out:u6} = "${dir:output}/u6.data"
```

```
while not os.path.exists("${dir:working}/token_PROP.data"):

      time.sleep(1)

shutil.copy("${dir:working}/token_PROP.data",
"${dir:output}/token_PROP.data")

${out:token_PROP} = "${dir:output}/token_PROP.data"



while not os.path.exists("${dir:working}/token_ADE.data"):

      time.sleep(1)

shutil.copy("${dir:working}/token_ADE.data",
"${dir:output}/token_ADE.data")

${out:token_ADE} = "${dir:output}/token_ADE.data"



while not os.path.exists("${dir:working}/manobra.data"):

      time.sleep(1)

shutil.copy("${dir:working}/manobra.data",
"${dir:output}/manobra.data")

${out:manobra} = "${dir:output}/manobra.data"




while not os.path.exists("${dir:working}/ADE_.sim"):

      time.sleep(1)

shutil.copy("${dir:working}/ADE_.sim", "${dir:output}/ADE_.sim")

${out:sim_param_ADE} = "${dir:output}/ADE_.sim"



while not os.path.exists("${dir:working}/Propulsion_sys_.sim"):

      time.sleep(1)

shutil.copy("${dir:working}/Propulsion_sys_.sim",
"${dir:output}/Propulsion_sys_.sim")

${out:sim_param_Propulsion_sys} = "${dir:output}/Propulsion_sys_.sim"



while not os.path.exists("${dir:working}/Thruster_.sim"):
```

```
      time.sleep(1)

shutil.copy("${dir:working}/Thruster_.sim",
"${dir:output}/Thruster_.sim")

${out:sim_param_Thruster} = "${dir:output}/Thruster_.sim"
```

Ferramenta ADE

Pré-execução

```
import os

shutil.copy("${in:ADE}", "${dir:working}/")

shutil.copy("${in:param_ADE1}", "${dir:working}/")

shutil.copy("${in:param_ADE2}", "${dir:working}/")

shutil.copy("${in:param_ADE3}", "${dir:working}/")

shutil.copy("${in:param_ADE4}", "${dir:working}/")

shutil.copy("${in:param_ADE5}", "${dir:working}/")

shutil.copy("${in:param_ADE6}", "${dir:working}/")

shutil.copy("${in:param_ADE7}", "${dir:working}/")

shutil.copy("${in:param_ADE8}", "${dir:working}/")

shutil.copy("${in:param_ADE9}", "${dir:working}/")

shutil.copy("${in:param_ADE10}", "${dir:working}/")

shutil.copy("${in:param_ADE11}", "${dir:working}/")

shutil.copy("${in:param_ADE12}", "${dir:working}/")

shutil.copy("${in:param_ADE13}", "${dir:working}/")

shutil.copy("${in:param_ADE14}", "${dir:working}/")

shutil.copy("${in:param_ADE15}", "${dir:working}/")

shutil.copy("${in:param_ADE16}", "${dir:working}/")

shutil.copy("${in:param_ADE17}", "${dir:working}/")

shutil.copy("${in:param_ADE18}", "${dir:working}/")
```

```
shutil.copy("${in:param_ADE20}", "${dir:working}/")
shutil.copy("${in:param_ADE21}", "${dir:working}/")
shutil.copy("${in:param_ADE22}", "${dir:working}/")
shutil.copy("${in:param_ADE23}", "${dir:working}/")
shutil.copy("${in:param_ADE24}", "${dir:working}/")
shutil.copy("${in:param_ADE25}", "${dir:working}/")
shutil.copy("${in:param_ADE26}", "${dir:working}/")

shutil.copy("${in:gp_ADE}", "${dir:working}/")
shutil.copy("${in:sim_param_ADE}", "${dir:working}/")

shutil.copy("${in:u1}", "${dir:working}/")
shutil.copy("${in:u2}", "${dir:working}/")
shutil.copy("${in:u3}", "${dir:working}/")
shutil.copy("${in:u4}", "${dir:working}/")
shutil.copy("${in:u5}", "${dir:working}/")
shutil.copy("${in:u6}", "${dir:working}/")

shutil.copy("${in:Thrust_1}", "${dir:working}/")
shutil.copy("${in:Thrust_3}", "${dir:working}/")
shutil.copy("${in:Thrust_5}", "${dir:working}/")
```

Execução

```
ADE_.exe
```

Pós-execução

```
import os.path
import time
```

```
while not os.path.exists("${dir:working}/x1.data"):
      time.sleep(1)
shutil.copy("${dir:working}/x1.data", "${dir:output}/x1.data")
${out:x1} = "${dir:output}/x1.data"

while not os.path.exists("${dir:working}/x2.data"):
      time.sleep(1)
shutil.copy("${dir:working}/x2.data", "${dir:output}/x2.data")
${out:x2} = "${dir:output}/x2.data"

while not os.path.exists("${dir:working}/x3.data"):
      time.sleep(1)
shutil.copy("${dir:working}/x3.data", "${dir:output}/x3.data")
${out:x3} = "${dir:output}/x3.data"

while not os.path.exists("${dir:working}/x4.data"):
      time.sleep(1)
shutil.copy("${dir:working}/x4.data", "${dir:output}/x4.data")
${out:x4} = "${dir:output}/x4.data"

while not os.path.exists("${dir:working}/x5.data"):
      time.sleep(1)
shutil.copy("${dir:working}/x5.data", "${dir:output}/x5.data")
${out:x5} = "${dir:output}/x5.data"

while not os.path.exists("${dir:working}/x6.data"):
      time.sleep(1)
shutil.copy("${dir:working}/x6.data", "${dir:output}/x6.data")
${out:x6} = "${dir:output}/x6.data"
```

```
ADE_load


Pré-execução

import os

shutil.copy("${in:ADE}", "${dir:working}")


Execução

AMELoad ADE


Pós-execução

import os.path

import time


shutil.copy("${dir:working}/ADE_.000.png",
"${dir:output}/ADE_.000.png")

${out:param_ADE1} = "${dir:output}/ADE_.000.png"


shutil.copy("${dir:working}/ADE_.cir", "${dir:output}/ADE_.cir")

${out:param_ADE2} = "${dir:output}/ADE_.cir"


shutil.copy("${dir:working}/ADE_.data", "${dir:output}/ADE_.data")

${out:param_ADE3} = "${dir:output}/ADE_.data"


shutil.copy("${dir:working}/ADE_.dll", "${dir:output}/ADE_.dll")

${out:param_ADE4} = "${dir:output}/ADE_.dll"


shutil.copy("${dir:working}/ADE_.err", "${dir:output}/ADE_.err")

${out:param_ADE5} = "${dir:output}/ADE_.err"
```

```
shutil.copy("${dir:working}/ADE_.exe", "${dir:output}/ADE_.exe")

${out:param_ADE6} = "${dir:output}/ADE_.exe"


shutil.copy("${dir:working}/ADE_.gp2", "${dir:output}/ADE_.gp2")

${out:param_ADE7} = "${dir:output}/ADE_.gp2"


shutil.copy("${dir:working}/ADE_.jac0", "${dir:output}/ADE_.jac0")

${out:param_ADE8} = "${dir:output}/ADE_.jac0"


shutil.copy("${dir:working}/ADE_.la", "${dir:output}/ADE_.la")

${out:param_ADE9} = "${dir:output}/ADE_.la"


shutil.copy("${dir:working}/ADE_.lock", "${dir:output}/ADE_.lock")

${out:param_ADE10} = "${dir:output}/ADE_.lock"


shutil.copy("${dir:working}/ADE_.make.bck",
"${dir:output}/ADE_.make.bck")

${out:param_ADE11} = "${dir:output}/ADE_.make.bck"


shutil.copy("${dir:working}/ADE_.mask", "${dir:output}/ADE_.mask")

${out:param_ADE12} = "${dir:output}/ADE_.mask"


shutil.copy("${dir:working}/ADE_.modelinfo",
"${dir:output}/ADE_.modelinfo")

${out:param_ADE13} = "${dir:output}/ADE_.modelinfo"


shutil.copy("${dir:working}/ADE_.o", "${dir:output}/ADE_.o")

${out:param_ADE14} = "${dir:output}/ADE_.o"
```

```
shutil.copy("${dir:working}/ADE_.param", "${dir:output}/ADE_.param")

${out:param_ADE15} = "${dir:output}/ADE_.param"


shutil.copy("${dir:working}/ADE_.pl", "${dir:output}/ADE_.pl")

${out:param_ADE16} = "${dir:output}/ADE_.pl"


shutil.copy("${dir:working}/ADE_.results",
"${dir:output}/ADE_.results")

${out:param_ADE17} = "${dir:output}/ADE_.results"


shutil.copy("${dir:working}/ADE_.sad", "${dir:output}/ADE_.sad")

${out:param_ADE18} = "${dir:output}/ADE_.sad"




shutil.copy("${dir:working}/ADE_.ssf", "${dir:output}/ADE_.ssf")

${out:param_ADE20} = "${dir:output}/ADE_.ssf"


shutil.copy("${dir:working}/ADE_.state", "${dir:output}/ADE_.state")

${out:param_ADE21} = "${dir:output}/ADE_.state"


shutil.copy("${dir:working}/ADE_.units", "${dir:output}/ADE_.units")

${out:param_ADE22} = "${dir:output}/ADE_.units"


shutil.copy("${dir:working}/ADE_.var", "${dir:output}/ADE_.var")

${out:param_ADE23} = "${dir:output}/ADE_.var"


shutil.copy("${dir:working}/ADE_.views", "${dir:output}/ADE_.views")

${out:param_ADE24} = "${dir:output}/ADE_.views"
```

```
shutil.copy("${dir:working}/ADE_.vl", "${dir:output}/ADE_.vl")

${out:param_ADE25} = "${dir:output}/ADE_.vl"


shutil.copy("${dir:working}/ADE_.vl.active",
"${dir:output}/ADE_.vl.active")

${out:param_ADE26} = "${dir:output}/ADE_.vl.active"
```

Ferramenta ADE_param

Pré-execução

```
import os

shutil.copy("${in:ADE_param}", "${dir:working}")

shutil.copy("${in:Momento_inercia}", "${dir:working}")

shutil.copy("${in:Ambiente_num}", "${dir:working}")

shutil.copy("${in:Ambiente_grandeza_unidade}", "${dir:working}")

shutil.copy("${in:token_ADE}", "${dir:working}")

shutil.copy("${in:token_PROP}", "${dir:working}")
```

Execução

```
matlab.exe -nodisplay -nosplash -nodesktop -r run('ADE_param.m'); exit
```

Pós-execução

```
import os.path

import time


while not os.path.exists("${dir:working}/ADE_.gp"):

      time.sleep(1)

shutil.copy("${dir:working}/ADE_.gp", "${dir:output}/ADE_.gp")

${out:gp_ADE} = "${dir:output}/ADE_.gp"
```

```
while not os.path.exists("${dir:working}/token.data"):
      time.sleep(1)
shutil.copy("${dir:working}/token.data", "${dir:output}/token.data")
${out:token} = "${dir:output}/token.data"
```

Ferramenta LMP

Pré-execução

```
import os
shutil.copy("${in:liquid_properties}", "${dir:working}/")
shutil.copy("${in:LMP}", "${dir:working}/")
shutil.copy("${in:m1_dot}", "${dir:working}/")
shutil.copy("${in:m2_dot}", "${dir:working}/")
shutil.copy("${in:m3_dot}", "${dir:working}/")
shutil.copy("${in:m4_dot}", "${dir:working}/")
shutil.copy("${in:m5_dot}", "${dir:working}/")
shutil.copy("${in:m6_dot}", "${dir:working}/")
```

Execução

```
matlab.exe -nodisplay -nosplash -nodesktop -r
run('liquid_properties.m'); exit
```

Pós-execução

```
import os.path
import time

while not os.path.exists("${dir:working}/Izz_prop.data"):
      time.sleep(1)
```

```
shutil.copy("${dir:working}/Izz_prop.data",
"${dir:output}/Izz_prop.data")

${out:Izz_prop} = "${dir:output}/Izz_prop.data"



while not os.path.exists("${dir:working}/token_mp.data"):

      time.sleep(1)

shutil.copy("${dir:working}/token_mp.data",
"${dir:output}/token_mp.data")

${out:token_mp} = "${dir:output}/token_mp.data"



Ferramenta Mission_Env

Pré-execução

import os

shutil.copy("${in:mission_and_environment}", "${dir:working}/")



Execução

matlab.exe       -nodisplay       -nosplash       -nodesktop       -r
run('mission_and_environment.m');exit



Pós-execução

import os.path

import time



while                                                          not
os.path.exists("${dir:working}/Input_parameters_unidade.data"):

      time.sleep(1)

shutil.copy("${dir:working}/Input_parameters_unidade.data",
"${dir:output}/Input_parameters_unidade.data")

${out:Input_parameters_unidade}                                  =
"${dir:output}/Input_parameters_unidade.data"



while not os.path.exists("${dir:working}/Input_parameters_num.data"):
```

```
      time.sleep(1)

shutil.copy("${dir:working}/Input_parameters_num.data",
"${dir:output}/Input_parameters_num.data")

${out:Input_parameters_num}                                                    =
"${dir:output}/Input_parameters_num.data"



while                                                                        not
os.path.exists("${dir:working}/Ambiente_grandeza_unidade.data"):

      time.sleep(1)

shutil.copy("${dir:working}/Ambiente_grandeza_unidade.data",
"${dir:output}/Ambiente_grandeza_unidade.data")

${out:Ambiente_grandeza_unidade}                                               =
"${dir:output}/Ambiente_grandeza_unidade.data"



while not os.path.exists("${dir:working}/Ambiente_num.data"):

      time.sleep(1)

shutil.copy("${dir:working}/Ambiente_num.data",
"${dir:output}/Ambiente_num.data")

${out:Ambiente_num} = "${dir:output}/Ambiente_num.data"
```

Ferramenta PROP_A

Pré-execução

```
import os

shutil.copy("${in:Propulsion_sys}", "${dir:working}/")

shutil.copy("${in:param_PROP1}", "${dir:working}/")

shutil.copy("${in:param_PROP2}", "${dir:working}/")

shutil.copy("${in:param_PROP3}", "${dir:working}/")

shutil.copy("${in:param_PROP4}", "${dir:working}/")

shutil.copy("${in:param_PROP5}", "${dir:working}/")

shutil.copy("${in:param_PROP6}", "${dir:working}/")
```

```
shutil.copy("${in:param_PROP7}", "${dir:working}/")
shutil.copy("${in:param_PROP8}", "${dir:working}/")
shutil.copy("${in:param_PROP9}", "${dir:working}/")
shutil.copy("${in:param_PROP10}", "${dir:working}/")
shutil.copy("${in:param_PROP11}", "${dir:working}/")
shutil.copy("${in:param_PROP12}", "${dir:working}/")
shutil.copy("${in:param_PROP13}", "${dir:working}/")
shutil.copy("${in:param_PROP14}", "${dir:working}/")
shutil.copy("${in:param_PROP15}", "${dir:working}/")
shutil.copy("${in:param_PROP16}", "${dir:working}/")
shutil.copy("${in:param_PROP17}", "${dir:working}/")
shutil.copy("${in:param_PROP18}", "${dir:working}/")

shutil.copy("${in:sim_param_Propulsion_sys}", "${dir:working}/")

shutil.copy("${in:param_PROP20}", "${dir:working}/")
shutil.copy("${in:param_PROP21}", "${dir:working}/")
shutil.copy("${in:param_PROP22}", "${dir:working}/")
shutil.copy("${in:param_PROP23}", "${dir:working}/")
shutil.copy("${in:param_PROP24}", "${dir:working}/")
shutil.copy("${in:param_PROP25}", "${dir:working}/")

shutil.copy("${in:gp_PROP}", "${dir:working}/")

shutil.copy("${in:u1}", "${dir:working}/")
shutil.copy("${in:u2}", "${dir:working}/")
shutil.copy("${in:u3}", "${dir:working}/")
shutil.copy("${in:u4}", "${dir:working}/")
shutil.copy("${in:u5}", "${dir:working}/")
```

```
shutil.copy("${in:u6}", "${dir:working}/")
```

Execução

Propulsion_sys_.exe

Pós-execução

```
import os.path
import time

shutil.copy("${dir:working}/dV.data", "${dir:output}/dV.data")
${out:dV} = "${dir:output}/dV.data"

shutil.copy("${dir:working}/m1_dot.data", "${dir:output}/m1_dot.data")
${out:m1_dot} = "${dir:output}/m1_dot.data"

shutil.copy("${dir:working}/m2_dot.data", "${dir:output}/m2_dot.data")
${out:m2_dot} = "${dir:output}/m2_dot.data"

shutil.copy("${dir:working}/m3_dot.data", "${dir:output}/m3_dot.data")
${out:m3_dot} = "${dir:output}/m3_dot.data"

shutil.copy("${dir:working}/m4_dot.data", "${dir:output}/m4_dot.data")
${out:m4_dot} = "${dir:output}/m4_dot.data"

shutil.copy("${dir:working}/m5_dot.data", "${dir:output}/m5_dot.data")
${out:m5_dot} = "${dir:output}/m5_dot.data"

shutil.copy("${dir:working}/m6_dot.data", "${dir:output}/m6_dot.data")
${out:m6_dot} = "${dir:output}/m6_dot.data"
```

Ferramenta PROP_B

Pré-execução

```
import os
shutil.copy("${in:Thruster}", "${dir:working}/")

shutil.copy("${in:m1_dot}", "${dir:working}/")
shutil.copy("${in:m2_dot}", "${dir:working}/")
shutil.copy("${in:m3_dot}", "${dir:working}/")
shutil.copy("${in:m4_dot}", "${dir:working}/")
shutil.copy("${in:m5_dot}", "${dir:working}/")
shutil.copy("${in:m6_dot}", "${dir:working}/")

shutil.copy("${in:param_THRUST_1}", "${dir:working}/")
shutil.copy("${in:param_THRUST_2}", "${dir:working}/")
shutil.copy("${in:param_THRUST_3}", "${dir:working}/")
shutil.copy("${in:param_THRUST_4}", "${dir:working}/")
shutil.copy("${in:param_THRUST_5}", "${dir:working}/")
shutil.copy("${in:param_THRUST_6}", "${dir:working}/")
shutil.copy("${in:param_THRUST_7}", "${dir:working}/")
shutil.copy("${in:param_THRUST_8}", "${dir:working}/")
shutil.copy("${in:param_THRUST_9}", "${dir:working}/")
shutil.copy("${in:param_THRUST_10}", "${dir:working}/")
shutil.copy("${in:param_THRUST_11}", "${dir:working}/")
shutil.copy("${in:param_THRUST_12}", "${dir:working}/")
shutil.copy("${in:param_THRUST_13}", "${dir:working}/")
shutil.copy("${in:param_THRUST_14}", "${dir:working}/")
shutil.copy("${in:param_THRUST_15}", "${dir:working}/")
```

```
shutil.copy("${in:param_THRUST_16}", "${dir:working}/")

shutil.copy("${in:param_THRUST_17}", "${dir:working}/")

shutil.copy("${in:param_THRUST_18}", "${dir:working}/")


shutil.copy("${in:sim_param_Thruster}", "${dir:working}/")


shutil.copy("${in:param_THRUST_20}", "${dir:working}/")

shutil.copy("${in:param_THRUST_21}", "${dir:working}/")

shutil.copy("${in:param_THRUST_22}", "${dir:working}/")

shutil.copy("${in:param_THRUST_23}", "${dir:working}/")

shutil.copy("${in:param_THRUST_24}", "${dir:working}/")


Execução

Thruster_.exe


Pós-execução

import os.path

import time


while not os.path.exists("${dir:working}/Thrust_1.data"):

      time.sleep(1)

shutil.copy("${dir:working}/Thrust_1.data",
"${dir:output}/Thrust_1.data")

${out:Thrust_1} = "${dir:output}/Thrust_1.data"


while not os.path.exists("${dir:working}/Thrust_3.data"):

      time.sleep(1)

shutil.copy("${dir:working}/Thrust_3.data",
"${dir:output}/Thrust_3.data")

${out:Thrust_3} = "${dir:output}/Thrust_3.data"
```

```
while not os.path.exists("${dir:working}/Thrust_5.data"):

      time.sleep(1)

shutil.copy("${dir:working}/Thrust_5.data",
"${dir:output}/Thrust_5.data")

${out:Thrust_5} = "${dir:output}/Thrust_5.data"
```

Ferramenta PROP_load_A

Pré-execução

```
import os

shutil.copy("${in:Propulsion_sys}", "${dir:working}")
```

Execução

```
AMELoad Propulsion_sys
```

Pós-execução

```
import os.path

import time


shutil.copy("${dir:working}/Propulsion_sys_.000.png",
"${dir:output}/Propulsion_sys_.000.png")

${out:param_PROP1} = "${dir:output}/Propulsion_sys_.000.png"


shutil.copy("${dir:working}/Propulsion_sys_.cir",
"${dir:output}/Propulsion_sys_.cir")

${out:param_PROP2} = "${dir:output}/Propulsion_sys_.cir"


shutil.copy("${dir:working}/Propulsion_sys_.data",
"${dir:output}/Propulsion_sys_.data")

${out:param_PROP3} = "${dir:output}/Propulsion_sys_.data"
```

```
shutil.copy("${dir:working}/Propulsion_sys_.dll",
"${dir:output}/Propulsion_sys_.dll")

${out:param_PROP4} = "${dir:output}/Propulsion_sys_.dll"



shutil.copy("${dir:working}/Propulsion_sys_.err",
"${dir:output}/Propulsion_sys_.err")

${out:param_PROP5} = "${dir:output}/Propulsion_sys_.err"



shutil.copy("${dir:working}/Propulsion_sys_.exe",
"${dir:output}/Propulsion_sys_.exe")

${out:param_PROP6} = "${dir:output}/Propulsion_sys_.exe"



shutil.copy("${dir:working}/Propulsion_sys_.gp",
"${dir:output}/Propulsion_sys_.gp")

${out:param_PROP7} = "${dir:output}/Propulsion_sys_.gp"



shutil.copy("${dir:working}/Propulsion_sys_.gp2",
"${dir:output}/Propulsion_sys_.gp2")

${out:param_PROP8} = "${dir:output}/Propulsion_sys_.gp2"



shutil.copy("${dir:working}/Propulsion_sys_.jac0",
"${dir:output}/Propulsion_sys_.jac0")

${out:param_PROP9} = "${dir:output}/Propulsion_sys_.jac0"



shutil.copy("${dir:working}/Propulsion_sys_.la",
"${dir:output}/Propulsion_sys_.la")

${out:param_PROP10} = "${dir:output}/Propulsion_sys_.la"



shutil.copy("${dir:working}/Propulsion_sys_.lock",
"${dir:output}/Propulsion_sys_.lock")

${out:param_PROP11} = "${dir:output}/Propulsion_sys_.lock"
```

```
shutil.copy("${dir:working}/Propulsion_sys_.make.bck",
"${dir:output}/Propulsion_sys_.make.bck")

${out:param_PROP12} = "${dir:output}/Propulsion_sys_.make.bck"



shutil.copy("${dir:working}/Propulsion_sys_.mask",
"${dir:output}/Propulsion_sys_.mask")

${out:param_PROP13} = "${dir:output}/Propulsion_sys_.mask"



shutil.copy("${dir:working}/Propulsion_sys_.modelinfo",
"${dir:output}/Propulsion_sys_.modelinfo")

${out:param_PROP14} = "${dir:output}/Propulsion_sys_.modelinfo"



shutil.copy("${dir:working}/Propulsion_sys_.param",
"${dir:output}/Propulsion_sys_.param")

${out:param_PROP15} = "${dir:output}/Propulsion_sys_.param"



shutil.copy("${dir:working}/Propulsion_sys_.pl",
"${dir:output}/Propulsion_sys_.pl")

${out:param_PROP16} = "${dir:output}/Propulsion_sys_.pl"



shutil.copy("${dir:working}/Propulsion_sys_.results",
"${dir:output}/Propulsion_sys_.results")

${out:param_PROP17} = "${dir:output}/Propulsion_sys_.results"



shutil.copy("${dir:working}/Propulsion_sys_.sad",
"${dir:output}/Propulsion_sys_.sad")

${out:param_PROP18} = "${dir:output}/Propulsion_sys_.sad"



shutil.copy("${dir:working}/Propulsion_sys_.ssf",
"${dir:output}/Propulsion_sys_.ssf")

${out:param_PROP20} = "${dir:output}/Propulsion_sys_.ssf"
```

```
shutil.copy("${dir:working}/Propulsion_sys_.state",
"${dir:output}/Propulsion_sys_.state")

${out:param_PROP21} = "${dir:output}/Propulsion_sys_.state"



shutil.copy("${dir:working}/Propulsion_sys_.units",
"${dir:output}/Propulsion_sys_.units")

${out:param_PROP22} = "${dir:output}/Propulsion_sys_.units"



shutil.copy("${dir:working}/Propulsion_sys_.var",
"${dir:output}/Propulsion_sys_.var")

${out:param_PROP23} = "${dir:output}/Propulsion_sys_.var"



shutil.copy("${dir:working}/Propulsion_sys_.views",
"${dir:output}/Propulsion_sys_.views")

${out:param_PROP24} = "${dir:output}/Propulsion_sys_.views"



shutil.copy("${dir:working}/Propulsion_sys_.vl",
"${dir:output}/Propulsion_sys_.vl")

${out:param_PROP25} = "${dir:output}/Propulsion_sys_.vl"



Ferramenta PROP_load_B



Pré-execução

import os

shutil.copy("${in:Thruster}", "${dir:working}")



Execução

AMELoad Thruster



Pós-execução

import os.path
```

```
import time



shutil.copy("${dir:working}/Thruster_.000.png",
"${dir:output}/Thruster_.000.png")

${out:param_THRUST_1} = "${dir:output}/Thruster_.000.png"



shutil.copy("${dir:working}/Thruster_.cir",
"${dir:output}/Thruster_.cir")

${out:param_THRUST_2} = "${dir:output}/Thruster_.cir"



shutil.copy("${dir:working}/Thruster_.data",
"${dir:output}/Thruster_.data")

${out:param_THRUST_3} = "${dir:output}/Thruster_.data"



shutil.copy("${dir:working}/Thruster_.dll",
"${dir:output}/Thruster_.dll")

${out:param_THRUST_4} = "${dir:output}/Thruster_.dll"



shutil.copy("${dir:working}/Thruster_.err",
"${dir:output}/Thruster_.err")

${out:param_THRUST_5} = "${dir:output}/Thruster_.err"



shutil.copy("${dir:working}/Thruster_.exe",
"${dir:output}/Thruster_.exe")

${out:param_THRUST_6} = "${dir:output}/Thruster_.exe"



shutil.copy("${dir:working}/Thruster_.gp",
"${dir:output}/Thruster_.gp")

${out:param_THRUST_7} = "${dir:output}/Thruster_.gp"



shutil.copy("${dir:working}/Thruster_.gp2",
"${dir:output}/Thruster_.gp2")

${out:param_THRUST_8} = "${dir:output}/Thruster_.gp2"
```

```
shutil.copy("${dir:working}/Thruster_.jac0",
"${dir:output}/Thruster_.jac0")

${out:param_THRUST_9} = "${dir:output}/Thruster_.jac0"



shutil.copy("${dir:working}/Thruster_.la",
"${dir:output}/Thruster_.la")

${out:param_THRUST_10} = "${dir:output}/Thruster_.la"



shutil.copy("${dir:working}/Thruster_.lock",
"${dir:output}/Thruster_.lock")

${out:param_THRUST_11} = "${dir:output}/Thruster_.lock"



shutil.copy("${dir:working}/Thruster_.make.bck",
"${dir:output}/Thruster_.make.bck")

${out:param_THRUST_12} = "${dir:output}/Thruster_.make.bck"



shutil.copy("${dir:working}/Thruster_.mask",
"${dir:output}/Thruster_.mask")

${out:param_THRUST_13} = "${dir:output}/Thruster_.mask"



shutil.copy("${dir:working}/Thruster_.modelinfo",
"${dir:output}/Thruster_.modelinfo")

${out:param_THRUST_14} = "${dir:output}/Thruster_.modelinfo"



shutil.copy("${dir:working}/Thruster_.param",
"${dir:output}/Thruster_.param")

${out:param_THRUST_15} = "${dir:output}/Thruster_.param"



shutil.copy("${dir:working}/Thruster_.pl",
"${dir:output}/Thruster_.pl")

${out:param_THRUST_16} = "${dir:output}/Thruster_.pl"
```

```
shutil.copy("${dir:working}/Thruster_.results",
"${dir:output}/Thruster_.results")

${out:param_THRUST_17} = "${dir:output}/Thruster_.results"



shutil.copy("${dir:working}/Thruster_.sad",
"${dir:output}/Thruster_.sad")

${out:param_THRUST_18} = "${dir:output}/Thruster_.sad"




shutil.copy("${dir:working}/Thruster_.ssf",
"${dir:output}/Thruster_.ssf")

${out:param_THRUST_20} = "${dir:output}/Thruster_.ssf"



shutil.copy("${dir:working}/Thruster_.state",
"${dir:output}/Thruster_.state")

${out:param_THRUST_21} = "${dir:output}/Thruster_.state"



shutil.copy("${dir:working}/Thruster_.units",
"${dir:output}/Thruster_.units")

${out:param_THRUST_22} = "${dir:output}/Thruster_.units"



shutil.copy("${dir:working}/Thruster_.var",
"${dir:output}/Thruster_.var")

${out:param_THRUST_23} = "${dir:output}/Thruster_.var"



shutil.copy("${dir:working}/Thruster_.views",
"${dir:output}/Thruster_.views")

${out:param_THRUST_24} = "${dir:output}/Thruster_.views"



shutil.copy("${dir:working}/Thruster_.vl",
"${dir:output}/Thruster_.vl")

${out:param_THRUST_25} = "${dir:output}/Thruster_.vl"
```

```
shutil.copy("${dir:working}/Thruster_.gp",
"${dir:output}/Thruster_.gp")

${out:param_THRUST_26} = "${dir:output}/Thruster_.gp"
```

Ferramenta PROP_param

Pré-execução

```
import os

shutil.copy("${in:PROP_param}", "${dir:working}")

shutil.copy("${in:token_PROP}", "${dir:working}")

shutil.copy("${in:manobra}", "${dir:working}")
```

Execução

```
matlab.exe -nodisplay -nosplash -nodesktop -r run('PROP_param.m');
exit
```

Pós-execução

```
import os.path

import time

while not os.path.exists("${dir:working}/Propulsion_sys_.gp"):

      time.sleep(1)

shutil.copy("${dir:working}/Propulsion_sys_.gp",
"${dir:output}/Propulsion_sys_.gp")

${out:gp_PROP} = "${dir:output}/Propulsion_sys_.gp"
```

Ferramenta Thruster_param

Pré-execução

```
import os

shutil.copy("${in:Thruster_param}", "${dir:working}")

shutil.copy("${in:gas_properties_1}", "${dir:working}")
```

```
shutil.copy("${in:gas_properties_2}", "${dir:working}")
```

Execução

```
matlab.exe -nodisplay -nosplash -nodesktop -r run('Thruster_param.m');
exit
```

Pós-execução

```
import os.path

import time


while not os.path.exists("${dir:working}/Thruster_.gp"):

      time.sleep(1)

shutil.copy("${dir:working}/Thruster_.gp",
"${dir:output}/Propulsion_sys_.gp")

${out:gp_Thruster} = "${dir:output}/Thruster_.gp"
```

Ferramenta SMP

Pré-execução

```
import os

shutil.copy("${in:geometry_and_material_param}", "${dir:working}/")

shutil.copy("${in:input_mass_prop}", "${dir:working}/")

shutil.copy("${in:Input_parameters_unidade}", "${dir:working}/")

shutil.copy("${in:Input_parameters_num}", "${dir:working}/")
```

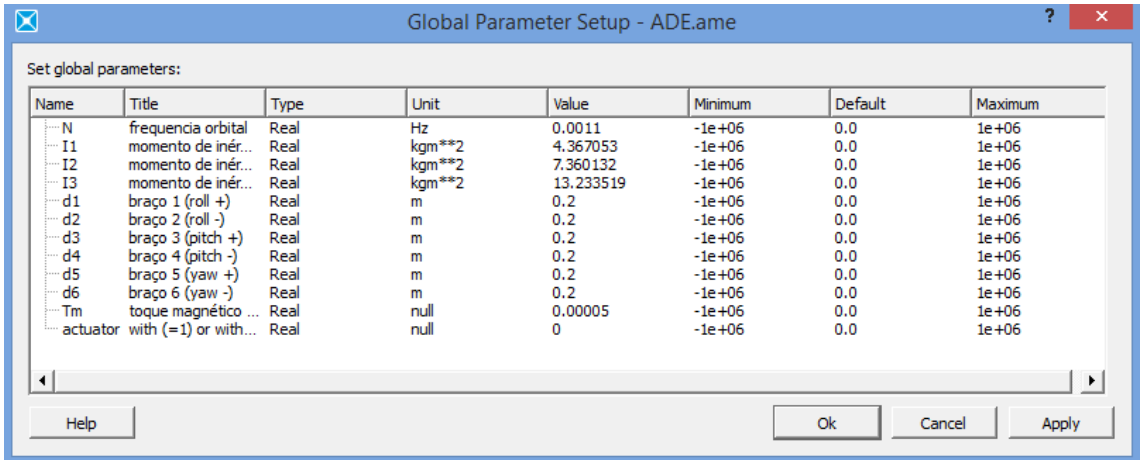
Execução

```
WScilex.exe       -nodisplay       -nosplash       -nodesktop       -r
run('input_mass_prop.sce'); exit
```

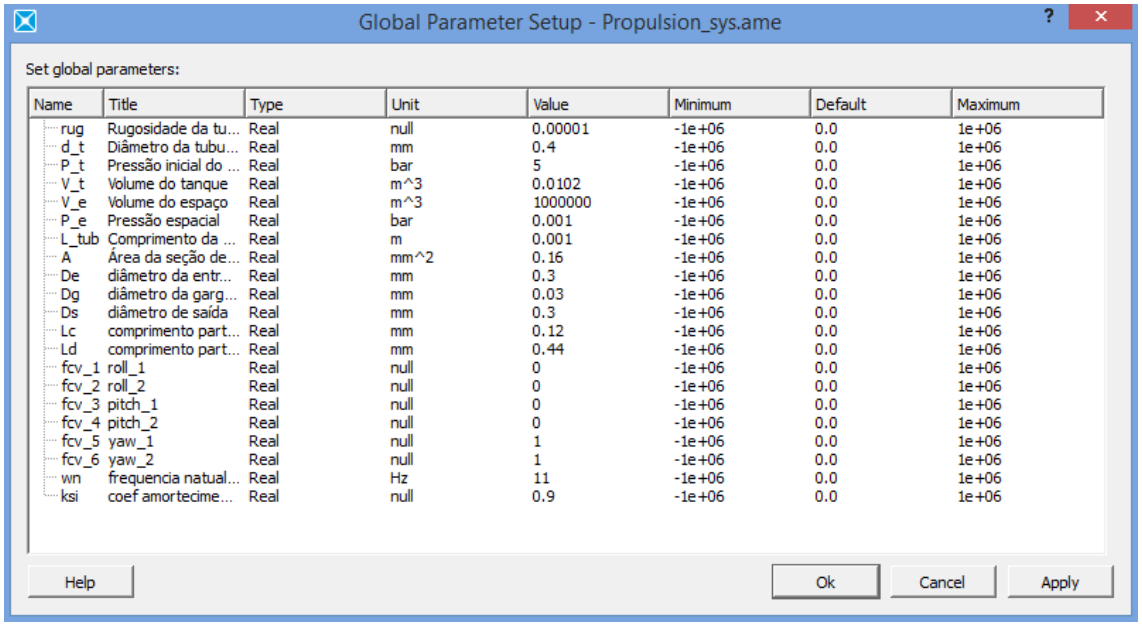
Pós-execução

```
import os.path

import time



while not os.path.exists("${dir:working}/Momento_inercia.data"):

      time.sleep(1)

shutil.copy("${dir:working}/Momento_inercia.data",
"${dir:output}/Momento_inercia.data")

${out:Momento_inercia} = "${dir:output}/Momento_inercia.data"



while not os.path.exists("${dir:working}/Momento_inercia_num.data"):

      time.sleep(1)

shutil.copy("${dir:working}/Momento_inercia_num.data",
"${dir:output}/Momento_inercia_num.data")

${out:Momento_inercia} = "${dir:output}/Momento_inercia_num.data"



while not os.path.exists("${dir:working}/Centro_massa.data"):

      time.sleep(1)

shutil.copy("${dir:working}/Centro_massa.data",
"${dir:output}/Centro_massa.data")

${out:Centro_massa} = "${dir:output}/Centro_massa.data"



while not os.path.exists("${dir:working}/Centro_massa_num.data"):

      time.sleep(1)

shutil.copy("${dir:working}/Centro_massa_num.data",
"${dir:output}/Centro_massa_num.data")

${out:Centro_massa} = "${dir:output}/Centro_massa_num.data"
```

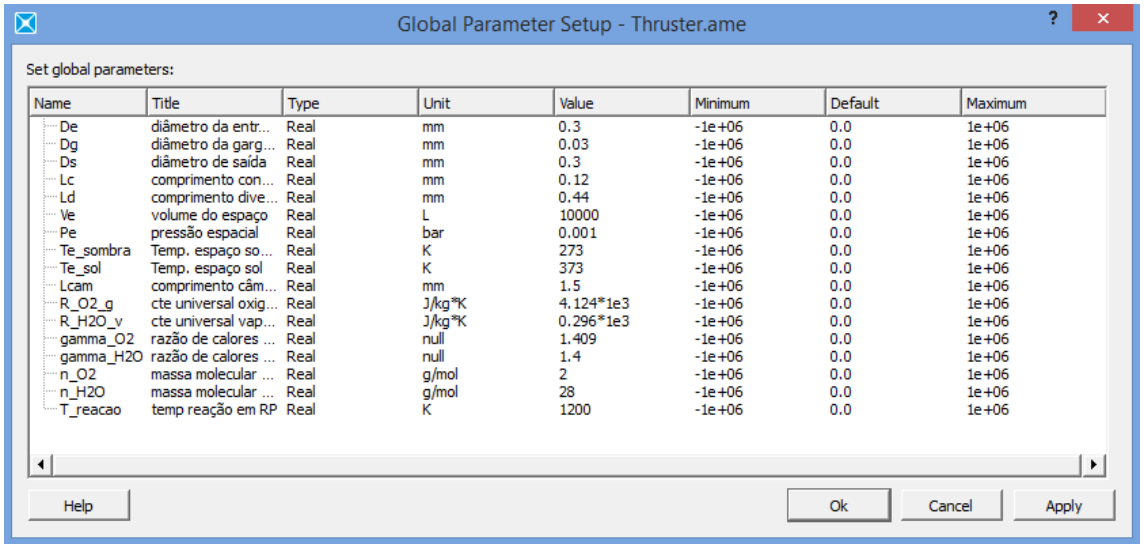
APÊNDICE F – PARÂMETROS GLOBAIS DOS MODELOS FÍSICOS (*AMESim*)



Name	Title	Type	Unit	Value	Minimum	Default	Maximum
N	frequencia orbital	Real	Hz	0.0011	-1e+06	0.0	1e+06
I1	momento de inér...	Real	kgm**2	4.367053	-1e+06	0.0	1e+06
I2	momento de inér...	Real	kgm**2	7.360132	-1e+06	0.0	1e+06
I3	momento de inér...	Real	kgm**2	13.233519	-1e+06	0.0	1e+06
d1	braço 1 (roll +)	Real	m	0.2	-1e+06	0.0	1e+06
d2	braço 2 (roll -)	Real	m	0.2	-1e+06	0.0	1e+06
d3	braço 3 (pitch +)	Real	m	0.2	-1e+06	0.0	1e+06
d4	braço 4 (pitch -)	Real	m	0.2	-1e+06	0.0	1e+06
d5	braço 5 (yaw +)	Real	m	0.2	-1e+06	0.0	1e+06
d6	braço 6 (yaw -)	Real	m	0.2	-1e+06	0.0	1e+06
Tm	toque magnético ...	Real	null	0.00005	-1e+06	0.0	1e+06
actuator	with (=1) or with...	Real	null	0	-1e+06	0.0	1e+06



Name	Title	Type	Unit	Value	Minimum	Default	Maximum
rug	Rugosidade da tu...	Real	null	0.00001	-1e+06	0.0	1e+06
d_t	Diâmetro da tubu...	Real	mm	0.4	-1e+06	0.0	1e+06
P_t	Pressão inicial do ...	Real	bar	5	-1e+06	0.0	1e+06
V_t	Volume do tanque	Real	m^3	0.0102	-1e+06	0.0	1e+06
V_e	Volume do espaço	Real	m^3	1000000	-1e+06	0.0	1e+06
P_e	Pressão espacial	Real	bar	0.001	-1e+06	0.0	1e+06
L_tub	Comprimento da ...	Real	m	0.001	-1e+06	0.0	1e+06
A	Área da seção de...	Real	mm^2	0.16	-1e+06	0.0	1e+06
De	diâmetro da entr...	Real	mm	0.3	-1e+06	0.0	1e+06
Dg	diâmetro da garg...	Real	mm	0.03	-1e+06	0.0	1e+06
Ds	diâmetro de saída	Real	mm	0.3	-1e+06	0.0	1e+06
Lc	comprimento part...	Real	mm	0.12	-1e+06	0.0	1e+06
Ld	comprimento part...	Real	mm	0.44	-1e+06	0.0	1e+06
fcv_1	roll_1	Real	null	0	-1e+06	0.0	1e+06
fcv_2	roll_2	Real	null	0	-1e+06	0.0	1e+06
fcv_3	pitch_1	Real	null	0	-1e+06	0.0	1e+06
fcv_4	pitch_2	Real	null	0	-1e+06	0.0	1e+06
fcv_5	yaw_1	Real	null	1	-1e+06	0.0	1e+06
fcv_6	yaw_2	Real	null	1	-1e+06	0.0	1e+06
wn	frequencia natual...	Real	Hz	11	-1e+06	0.0	1e+06
ksi	coef amortecime...	Real	null	0.9	-1e+06	0.0	1e+06


Global Parameter Setup - Thruster.ame
Set global parameters:
Name | Title | Type | Unit | Value | Minimum | Default | Maximum
De | diâmetro da entr... | Real | mm | 0.3 | -1e+06 | 0.0 | 1e+06
Dg | diâmetro da garg... | Real | mm | 0.03 | -1e+06 | 0.0 | 1e+06
Ds | diâmetro de saída | Real | mm | 0.3 | -1e+06 | 0.0 | 1e+06
Lc | comprimento con... | Real | mm | 0.12 | -1e+06 | 0.0 | 1e+06
Ld | comprimento dive... | Real | mm | 0.44 | -1e+06 | 0.0 | 1e+06
Ve | volume do espaço | Real | L | 10000 | -1e+06 | 0.0 | 1e+06
Pe | pressão espacial | Real | bar | 0.001 | -1e+06 | 0.0 | 1e+06
Te_sombra | Temp. espaço so... | Real | K | 273 | -1e+06 | 0.0 | 1e+06
Te_sol | Temp. espaço sol | Real | K | 373 | -1e+06 | 0.0 | 1e+06
Lcam | comprimento câm... | Real | mm | 1.5 | -1e+06 | 0.0 | 1e+06
R_O2_g | cte universal oxig... | Real | J/kg*K | 4.124*1e3 | -1e+06 | 0.0 | 1e+06
R_H2O_v | cte universal vap... | Real | J/kg*K | 0.296*1e3 | -1e+06 | 0.0 | 1e+06
gamma_O2 | razão de calores ... | Real | null | 1.409 | -1e+06 | 0.0 | 1e+06
gamma_H2O | razão de calores ... | Real | null | 1.4 | -1e+06 | 0.0 | 1e+06
n_O2 | massa molecular ... | Real | g/mol | 2 | -1e+06 | 0.0 | 1e+06
n_H2O | massa molecular ... | Real | g/mol | 28 | -1e+06 | 0.0 | 1e+06
T_reacao | temp reação em RP | Real | K | 1200 | -1e+06 | 0.0 | 1e+06
Help
Ok
Cancel
Apply