sid.inpe.br/mtc-m21b/2014/03.24.13.39-TDI

EMPREGO DE TÉCNICAS DE REALIDADE VIRTUAL EM INTERAÇÃO EM SIMULADORES OPERACIONAIS DE SATÉLITES

Christopher Shneider Cerqueira

Dissertação de Mestrado do Curso de Pós-Graduação em Engenharia e Tecnologia Espaciais/Engenharia e Gerenciamento de Sistemas Espaciais, orientada pelos Drs. Walter Abrahão dos Santos, e Ana Maria Ambrosio, aprovada em 17 de fevereiro de 2014.

URL do documento original:
<http://urlib.net/8JMKD3MGP5W34M/3G28FG2>

INPE
São José dos Campos
2014

PUBLICADO POR:

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

CONSELHO DE EDITORAÇÃO E PRESERVAÇÃO DA PRODUÇÃO INTELECTUAL DO INPE (RE/DIR-204):
Presidente:
Marciana Leite Ribeiro - Serviço de Informação e Documentação (SID)
Membros:
Dr. Antonio Fernando Bertachini de Almeida Prado - Coordenação Engenharia e Tecnologia Espacial (ETE)
Drª Inez Staciarini Batista - Coordenação Ciências Espaciais e Atmosféricas (CEA)
Dr. Gerald Jean Francis Banon - Coordenação Observação da Terra (OBT)
Dr. Germano de Souza Kienbaum - Centro de Tecnologias Especiais (CTE)
Dr. Manoel Alonso Gan - Centro de Previsão de Tempo e Estudos Climáticos (CPT)
Drª Maria do Carmo de Andrade Nono - Conselho de Pós-Graduação
Dr. Plínio Carlos Alvalá - Centro de Ciência do Sistema Terrestre (CST)
BIBLIOTECA DIGITAL:
Dr. Gerald Jean Francis Banon - Coordenação de Observação da Terra (OBT)
REVISÃO E NORMALIZAÇÃO DOCUMENTÁRIA:
Marciana Leite Ribeiro - Serviço de Informação e Documentação (SID)
Yolanda Ribeiro da Silva Souza - Serviço de Informação e Documentação (SID)
EDITORAÇÃO ELETRÔNICA:
Maria Tereza Smith de Brito - Serviço de Informação e Documentação (SID)
André Luis Dias Fernandes - Serviço de Informação e Documentação (SID)

sid.inpe.br/mtc-m21b/2014/03.24.13.39-TDI

EMPREGO DE TÉCNICAS DE REALIDADE VIRTUAL EM INTERAÇÃO EM SIMULADORES OPERACIONAIS DE SATÉLITES

Christopher Shneider Cerqueira

Dissertação de Mestrado do Curso de Pós-Graduação em Engenharia e Tecnologia Espaciais/Engenharia e Gerenciamento de Sistemas Espaciais, orientada pelos Drs. Walter Abrahão dos Santos, e Ana Maria Ambrosio, aprovada em 17 de fevereiro de 2014.

URL do documento original:
<http://urlib.net/8JMKD3MGP5W34M/3G28FG2>

INPE
São José dos Campos
2014

Dados Internacionais de Catalogação na Publicação (CIP)

Cerqueira, Christopher Shneider.

C336e Emprego de técnicas de realidade virtual em interação em simuladores operacionais de satélites / Christopher Shneider Cerqueira. – São José dos Campos : INPE, 2014.

xxi + 157 p. ; (sid.inpe.br/mtc-m21b/2014/03.24.13.39-TDI)

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

Orientadores : Drs. Walter Abrahão dos Santos, e Ana Maria Ambrosio.

1. simulação. 2. interação homem-máquina. 3. interface-usuário. I.Título.

CDU 629.783

Esta obra foi licenciada sob uma Licença Creative Commons Atribuição-NãoComercial 3.0 Não Adaptada.

This work is licensed under a Creative Commons Attribution-NonCommercial 3.0 Unported License.

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

Engenharia e Tecnologia Espaciais/Gerenciamento de Sistemas Espaciais

Dr. Marcelo Lopes de Oliveira e Souza

Presidente / INPE / SJCampos - SP

Dr. Walter Abrahão dos Santos

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

Dra. Ana Maria Ambrosio

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

Dr. Claudio Kirner

Convidado(a) / UNIFEI / Itajubá - MG

Este trabalho foi aprovado por:

( ) maioria simples

(X) unanimidade

Aluno (a): ***Christopher Shneider Cerqueira***

São José dos Campos, 17 de Fevereiro de 2014

"The product is no longer the basis of value. The experience is".

Venkat Ramaswamy

RESUMO

O desenvolvimento de sistemas espaciais requer o domínio de diversas técnicas de modelagem e simulação para alcançar os resultados esperados. O maior e mais complexo simulador desenvolvido no contexto de uma missão espacial é o *simulador operacional*, o qual visa dar apoio a todas as atividades de operação do satélite antes, durante e após o seu lançamento. Esta dissertação apresenta uma proposta de interface-usuário baseada em realidade virtual para simuladores operacionais de satélite, e propõe uma visualização interativa tridimensional com a utlização das técnicas de multiplas visões, modelos 3D, árvores cônicas e painéis contextualizados como metáforas para navegação e manipulação do conteúdo simulado. A interface-usuário proposta utiliza conceitos de interação computacional de terceira geração, ao invés de interfaces baseadas em metáforas de janelas, menus, ícones e apontadores chamados metáforas WIMP (janela, ícone, menu e ponteiro, do inglês, *Window, Icon, Menu, Pointer*). Foi desenvolvido um simulador operacional típico simplificado para ilustração e avaliação da interface-usuário, contendo modelos simplificados de funcionamento do satélite e de órbita. A interface-usuário proposta foi avaliada por usuários experientes do INPE, através da aplicação de um questionário de usabilidade. Os resultados mostraram que as técnicas de interação foram bem recebidas por diferentes perfis de usuários. Os passos de execução aplicados e orientações para expansão e reutilização deste trabalho são também discutidos.

VIRTUAL REALITY TECHNIQUES EMPLOYED INTO SATELLITE OPERATIONAL SIMULATORS

ABSTRACT

Space Systems development requires understanding of several modelling and simulation resources to accomplish space mission goals. The largest and more complex simulator developed into the space mission context is the operational simulator, which aims to support all operational activities before, during and after launch. This work proposes a user interface based on virtual reality to satellite operational simulator, proposing a 3D interactive visualization with multiple views, 3D models, conic views and contextualized panels techniques to navigate through simulated data. The user interface uses third generation interaction concepts instead of the windows, icon, menu and pointer metaphor (called WIMP metaphor - Windows, Icon, Menu and Pointer). A simple operational simulator to illustrate and evaluate the user-interface was developed with simplified behaviour and orbit propagation. The proposed user interface was evaluated by an usability questionnaire with experienced INPE users. The results show that the the interactive techniques applied were well accepted by different types of users. The tests, expansion and reuse of this work are discussed in this essay.

LISTA DE FIGURAS

Pág.

1.1 Segmentos de uma missão espacial com destaque aos elementos do CCS. 3
1.2 Metodologia aplicada 6

2.1 Elementos de uma simulação. 10
2.2 Simulação visual. 13
2.3 Exemplos de saídas visuais dos programas 14
2.4 Relação da quantidade de envolvimento do condutor no tempo de simulação. 15

3.1 Disciplinas relacionadas à experiencia do usuário. 18
3.2 Classificação da geração da interface relacionando a abstração, o poder computacional, o hardware, a metáfora e a direção da interação. 20
3.3 Contínuo Realidade-Virtualidade de Milgram 20
3.4 Diagrama de Kirner de múltiplas realidades. 21
3.5 Atributos da aceitação de um sistema. 24
3.6 *Feedback loop*. 26
3.7 Exemplo de *sketches* de um *storyboard*. 29
3.8 Arquitetura de elementos do oF disponíveis para uma interface interativa. 31
3.9 Exemplo de múltiplas visões. 35
3.10 Exemplo de viewports do oF 36
3.11 Alguns tipos de distribuição de dados em árvores. (a) hiperbólica, (b) mapa e (c) árvore cônica. 37
3.12 Árvore de discos reconfiguráveis 38
3.13 Exemplo de painéis com informações sensíveis ao contexto 39
3.14 Exemplo dos elementos de interação de um painel montado com o componente ofxUI 39

4.1 Resumo das características dos simuladores. 42
4.2 Fluxo de desenvolvimento do documento de especificação do modelo. . . 43
4.3 Elementos que descrevem o subsistema. 44
4.4 Comportamento dos elementos do modelo. 45
4.5 Stakeholders interessados na simulação do CBERS. 46
4.6 Ambiente de Análise para o SIMC3. 47
4.7 Ambiente de Operação para o SIMC3. 47
4.8 Caso de uso geral. 48
4.9 Caso de uso da interação com os parâmetros em evolução. 49

4.10 Caso de uso da interação com parâmetros. 50
4.11 Caso de uso da interação com log. 51
4.12 Caso de uso da interação com a agenda de execução e de eventos. 52
4.13 Janela principal do Analix. 55
4.14 Navegação para monitoração via tabelas. (a) escolha do contexto (modelo) para visualização. (b) janela com a lista de variáveis do modelo em evolução. 56
4.15 Navegação para monitoração via gráficos. (a) escolha do contexto (modelo) para visualização. (b) janela com a lista de variáveis do modelo em evolução. 57
4.16 Janela de escolha de telecomando a ser enviado. 57

5.1 Exemplos de interfaces da ficção. (a) Exemplo com área central de manipulação com auxiliares ao redor. Utilização de árvore. (b) Exemplo com área de manipulação central e botoes ao redor, interação com toque no objeto. (c) Exmeplo com mapa, árvores, áreas de manipulações espalhadas, não possui uma área central. 60
5.2 Sketch da tela principal, mostrando as áreas de interação (*Views*). 61
5.3 Relação das *Views* com os Casos de Uso. 62
5.4 *Sketch* das *Views*. 63
5.5 *Sketches* de comportamento. 64
5.6 Diagrama de classe com uma simplificação da relação dos objetos da interface. 65
5.7 Conceito do *layout* das *Views* para esta proposta 66
5.8 Classes iView e iViewport que controlam as áreas de interação. 67
5.9 Diagrama de classes simplificado com as especializações da classe iView. 67
5.10 Fluxo de mudanças da View ativa para interação. 68
5.11 Primeiras renderizações do modelo do satélite. 69
5.12 Satélite com equipamentos agrupados por subsistema. 69
5.13 Renderização do modelo aberto com texturas metálicas. 70
5.14 Renderização do modelo fechado com texturas metálicas. 70
5.15 Cabeçalho da classe Node3D 72
5.16 Exemplo de RDT com equipamentos. 73
5.17 Cabeçalho da classe NodeRDT. 74
5.18 Cabeçalho da classe ParamView. 74
5.19 Teste de interação natural *multi-touch* em *SmartBoard*. 75
5.20 Exemplo da utilização de multiplos painéis para interação de controle, de log e com conteúdo. 76
5.21 Classes decoradoras dos nós e parâmetros para criação dos painéis. . . . 77

5.22 Relação das classes de controle de interação com a aplicação, tela e modelo. 78
5.23 Cabeçalhos das classes de controle de interação iAction e do modal de interação mouseClick. 79
5.24 Diagrama de classes completo da interface iSIMC3. 81
5.25 Tela inicial da interface proposta. 82
5.26 SatView sem nenhum subsistema selecionado. 83
5.27 SatView com subsistema selecionado. 84
5.28 Painel contextualizado aberto com as condições instantâneas. 84
5.29 Tela com a árvore RDT aberta. 85
5.30 Tela com a árvore RDT aberta e indicando uma nó selecionado. 85
5.31 Tela com a árvore RDT e com o painel aberto. 86
5.32 Desacoplamento do código, da estrutura e do conteúdo. 89

6.1 Resultados da avaliação preliminar. 96
6.2 Médias das respostas de cada perfil. 98
6.3 Resultados para o grupo de modeladores. 98
6.4 Resultados para o grupo de desenvolvedores. 99
6.5 Resultados para o grupo de operadores. 99
6.6 Resultados para o grupo de engenheiros. 100
6.7 Resultados para idades de aproximadamente trinta e cinquenta anos. . . 100
6.8 Médias das dimensões do questionário de usabilidade aplicado. 101

7.1 Inserção da interação na simulação visual. 106

A.1 Representação do modelo simplificado para controle de potência. 120
A.2 Modelo de carga no openModellica. 123
A.3 Curva de carga no openModellica. 123
A.4 Resposta do SciLab à função de carga. 125
A.5 Modelo de descarga no openModellica. 126
A.6 Curva de descarga no openModellica 127
A.7 Resposta do SciLab à função de descarga. 128
A.8 Curva com capacitâncias corrigidas para ficar semelhante ao tempo real de carga e descarga. 131
A.9 Fluxo de execução da biblioteca Spacetrack 133
A.10 Exemplo de saída do ofxSpacetrack 136
A.11 Elementos que definem uma situação de eclipse. 137
A.12 Elementos que definem uma situação de visibilidade entre satélite e estação terrena. 139
A.13 Exemplo do modelo ambiental, com satélite iluminado e fora da visada das três estações. 140

A.14 Exemplo do modelo ambiental, com satélite em eclipse e com visada das três estações. 140
A.15 Cabeçalho mostrando principais métodos de interface com o analisador léxico matemático muParser. 141

B.1 Diagrama de classes da estrutura de dados. 151
B.2 Diagrama de classes simplificado da estrutura de dados do modelo, com o acoplamento do modelo ambiental e da análise matemática. 152

C.1 *Specifying Satellite Behavior for an Operational Simulator* 154
C.2 *Development of an Interface to a Spacecraft Simulator Empowered by Virtual Reality* 155
C.3 Introdução à utilização de openFrameworks para o desenvolvimento de aplicações de RVA 156
C.4 *Serious Game Interaction Techniques Applied to an Operational Satellite Simulator* 157

LISTA DE TABELAS

Pág.

2.1 Taxonomia do ponto de vista de elementos gráficos. Adaptado de Wenzel et al. (2003) 11
2.2 Taxonomia do ponto de vista da informação relacionada. Adaptado de Wenzel et al. (2003) 12

6.1 Questionário da primeira etapa. 92
6.2 Questionário da segunda etapa. 93
6.3 Resumo e sugestão da ordem de utilização 103

7.1 Adaptação com novos elementos de interação à taxonomia proposta por Wenzel et al. (2003). 107

A.1 Modos operacionais do subsistema de suprimento de energia. 121
A.2 Relação do modo operacional, com consumo e modos operacionais dos subsistemas. 132

LISTA DE ABREVIATURAS E SIGLAS

2D	–	Duas Dimensões
3D	–	Três Dimensões
3DSMax	–	3D Studio Max
AA++	–	Algoritmos Astronômicos
AIAA	–	*American Institute of Aeronautics and Astronautics*
API	–	*Application Programming Interface*
ASSIMP	–	*Open Asset Import Library*
C	–	Linguagem de programação estruturada
C++	–	Linguagem de programação orientada a objeto
CAD	–	*Computer Aided Design*
CBERS	–	*China-Brazil Earth Resources Satellite 3&4*
CCS	–	Centro de Controle de Satélites
CCSDS	–	*Consultative Committee for Space Data Systems*
CNES	–	*Centre National d'Etudes Spatiales*
COLLADA	–	*Collaborative Design Activity*
DCS	–	*Data Collection Subsystem*
EGSE	–	*Electronic Ground Support Equipment*
EPSS	–	*Electrical Power Supply Subsystem*
EQ	–	Equipamento
ESA	–	*European Space Agency*
ESTEC	–	*European Space Research and Technology Centrer*
EX	–	*External Event*
FBM	–	*French-Brazilian Micro-satellite*
FBMSIM	–	*French-Brazilian Micro-satellite Simulator*
Gb	–	*GigaByte*
GPS	–	*Global Positioning System*
HIL	–	*Hardware in the Loop*
HILT	–	*Human in the Loop*
HLA	–	*High Level Architecture*
IDE	–	*Integrated Development Environment*
INPE	–	Instituto Nacional de Pesquisas Espaciais
ISO	–	*International Organization for Standardization*
iOS	–	*iPhone Operational System*
iSIMC3	–	Interface SIMC3
IU	–	Interface-Usuário
LVC	–	*Live, Virtual and Constructive*
MIDI	–	*Musical Instrument Digital Interface*
MIT	–	*Massachusetts Institute of Technology*
MP3	–	*MPEG(Moving Picture Experts Group)-1/2 Audio Layer 3*
oF	–	openFrameworks

ofxUI	–	*User Interface Library/Addon for openFrameworks*
OGG	–	*Ogg Vorbis*
OM	–	*Operational Mode*
openGL	–	*Open Graphics Library*
OSX	–	*Operational System Ten*
PIL	–	*Processor in the Loop*
POCO	–	*Plain Old CLR (Common Language Runtime) Object*
PW	–	*Electrical Power*
Qt	–	*Cute Toolkit*
RAM	–	*Random Access Memory*
RC	–	Circuito Resistor-Capacitor
RDT	–	*Reconfigurable Disk Tree*
RF	–	*Radio Frequency Irradiated Power*
RGB	–	Sistema de cores aditivas Red, Green, Blue
RV	–	Realidade Virtual
S/S	–	Subsistema
SATCS	–	*Satellite Control System*
SBC	–	Sociedade Brasileira de Computação
SBGames	–	Sociedade Brasileira de *Games*
SCD	–	Sistema de Coleta de Dados
SIL	–	*Software in the Loop*
SIMC3	–	Simulador dos Satélites CBERS 3&4
SIMRq	–	*Simulator Requirements*
SOA	–	*Service-Oriented Architecture*
STK	–	*System Tool Kit*
STL	–	*StereoLithography*
SW	–	*Switch*
TC	–	Telecomando
TH	–	*Thermal Power*
TLE	–	*Two Lines Elements*
TBC	–	*To be Confirmed*
TM	–	Telemetria
UX	–	*User Experience*
WAV	–	*WAVEform audio format*
WIMP	–	*Window, Icon, Menu, Pointer*
XMA	–	*XBox-360 Native Audio*
XML	–	*Extensible Markup Language*

SUMÁRIO

Pág.

1 INTRODUÇÃO 1
1.1 Contexto 1
1.2 Motivação 4
1.3 Objetivo do trabalho 4
1.4 Metodologia 5
1.5 Organização do texto 6

2 SIMULADORES DE SATÉLITES 9
2.1 Simulação 9
2.2 Visualizações em Simulação 11
2.3 Simulador Operacional 13

3 INTERAÇÃO HOMEM-MÁQUINA 17
3.1 Gerações e Realidades 17
3.2 Cognição, Representação e Usabilidade 22
3.3 Interação e Interface 25
3.4 *Framework* para desenvolvimento de interface - openFrameworks 28
3.4.1 Outros *frameworks* disponíveis para desenvolvimento de interfaces de terceira geração 30
3.5 Técnicas de interação de RV 32
3.5.1 Modelos 3D 32
3.5.2 Audio 33
3.5.3 Múltiplas visões e mapas 34
3.5.4 *RDT - Reconfigurable Disk Tree* 36
3.5.5 Painéis 37

4 O SIMULADOR SIMC3 41
4.1 Simuladores operacionais de satélites do INPE 41
4.2 Especificação do comportamento do satélite 42
4.3 Estudo dos requisitos do simulador SIMC3 44
4.3.1 Stakeholders e contexto 45
4.3.2 Casos de uso 47
4.3.2.1 Caso de Uso: Geral 48

4.3.2.2 Caso de uso: Evolução dos valores de parâmetros - *Evolution Interaction* . . . 49
4.3.2.3 Caso de uso: Parâmetros - *Parameter Interaction* . . . 49
4.3.2.4 Caso de uso: Log - *Log Interaction* . . . 51
4.3.2.5 Caso de uso: Agenda - *Schedule Interaction* . . . 51
4.4 Requisitos para o layout da interface . . . 53
4.5 Analix: análise da interface-usuário . . . 54

5 PROPOSTA DA INTERFACE-USUÁRIO COM TÉCNICAS DE REALIDADE VIRTUAL . . . 59
5.1 Design da interface . . . 59
5.2 Mapeamento das técnicas de interação . . . 64
5.2.1 Multiplas visões e mapas . . . 65
5.2.1.1 Aplicação da metáfora ao contexto . . . 65
5.2.1.2 Projeto de Software . . . 66
5.2.2 Modelos 3D . . . 67
5.2.2.1 Aplicação da metáfora ao contexto . . . 67
5.2.2.2 Projeto de Software . . . 70
5.2.3 RDT . . . 71
5.2.3.1 Aplicação da metáfora ao contexto . . . 71
5.2.3.2 Projeto de Software . . . 72
5.2.4 Painéis . . . 74
5.2.4.1 Aplicação da metáfora ao contexto . . . 74
5.2.4.2 Projeto de Software . . . 75
5.3 Controle da interação . . . 76
5.3.1 Projeto de Software . . . 78
5.4 Integração entre modelo, elementos de interface e comunicação . . . 79
5.5 Fluxo de execução . . . 82
5.6 Extensão e troca de artefatos . . . 86
5.6.1 Utilização de software livre e desacoplamento do código, da estrutura e do conteúdo . . . 88

6 AVALIAÇÕES . . . 91
6.1 Questionários para avaliação da interface . . . 91
6.1.1 Questionário da primeira etapa . . . 91
6.1.2 Questionário da segunda etapa . . . 92
6.2 Resultados da primeira etapa da avaliação . . . 95
6.3 Resultados da segunda etapa da avaliação . . . 97

7 CONCLUSÕES 105
7.1 Contribuições em conceitos 106
7.1.1 Visualização Interativa em Simulação 106
7.1.2 Adição de elementos na taxonomia de interação 107
7.2 Contribuições para o INPE 108
7.3 Contribuições de desenvolvimento 108
7.4 Trabalhos futuros 109

REFERÊNCIAS BIBLIOGRÁFICAS 111

APÊNDICE A - SIMULADOR SIMPLIFICADO 119
A.1 Modelagem do simulador simplificado 119
A.1.1 Modelo do Ambiente 119
A.1.2 Modelo do Satélite 120
A.1.2.1 Modelo do subsistema suprimento de energia 120
A.1.2.2 Subsistema representativo de todo o satélite 130
A.2 Implementação do simulador simplificado 132
A.2.1 Simulação do ambiente espacial 133
A.2.1.1 Órbita do satélite 133
A.2.1.2 Posição do Sol 136
A.2.1.3 Relação com as estações terrenas 138
A.2.2 Simulação do comportamento do satélite 141

APÊNDICE B - ARQUITETURA DE DADOS - XML 143
B.1 Arquitetura de arquivos 143
B.1.1 Nó da estrutura 143
B.1.2 Parâmetros 145
B.1.2.1 Parâmetro do tipo DATAVALUE 146
B.1.2.2 Parâmetro do tipo OPERATIONALMODE 147
B.1.2.3 Parâmetro do tipo EXTERNALEVENT 148
B.1.3 Expressões 148
B.1.4 Modelo 3D 149
B.2 Leitor de XML 149
B.2.1 Projeto de Software 150

APÊNDICE C - PUBLICAÇÕES REALIZADAS DURANTE A ELABORAÇÃO DESTA DISSERTAÇÃO 153

1 INTRODUÇÃO

1.1 Contexto

A atividade de Simulação é considerada fundamental para o apoio ao trabalho de engenharia e de operação desempenhado durante o ciclo de vida de uma missão espacial. De acordo com o Memorando Técnico publicado pela Agência Espacial Europeia em 2010, criar produtos de simulação de uma maneira consistente ao longo da missão espacial é reconhecido como boa prática de engenharia (EUROPEAN COOPERATION FOR SPACE STANDARDIZATION (ECSS), 2010).

O uso de simulações pode ocorrer em treinamentos, execução e comparação de teste, pesquisa de novas soluções, manufatura, exploração em engenharia, jogos de guerra, bem com em entretenimento. Em programas de missões espaciais a simulação possui vários potenciais de uso ao longo de todo o ciclo de vida da missão (EICKHOFF, 2009) (HOFFMANN; PERONDI, 2010). Alguns destes usos são ESA (EUROPEAN COOPERATION FOR SPACE STANDARDIZATION (ECSS), 2010) :

a) Análise, definição e validação de requisitos de sistema e requisitos técnicos.

b) Validação de que o projeto satisfaz os requisitos de alto nível.

c) Verificação e validação dos softwares.

d) Desenvolvimento de equipamentos (*EGSE - Electronic Ground Support Equipment*) e de procedimentos de teste.

e) Apoio as atividades de teste de unidade e de subsistemas do satélite.

f) Previsão de performance do sistema.

g) Desenvolvimento e validação de procedimentos operacionais.

h) Avaliação de solução de problemas causados por anomalias e falhas no segmento espacial.

i) Treinamento de equipes de operação de voo, seja para centros de controle ou para estações terrenas

Os três últimos usos são típicos de simuladores conhecidos como Simuladores Operacionais. Em uma missão espacial o simulador operacional é uma ferramenta de software usada antes do lançamento do veículo espacial para validar a prontidão e

exatidão de funcionamento de todo segmento solo e treinar as equipes de operadores para responder rápida e corretamente a situações rotineiras ou emergenciais inesperadas (REGGESTAD et al., 2011).

No INPE, o principal uso dos simuladores operacionais tem sido para treinamento de operadores, com uso nas atividades do segmento solo, sob responsabilidade do Centro de Controle de Satélites (CCS) (TOMINAGA et al., 2012). A Figura 1.1 ilustra um resumo dos segmentos, como aplicado pela EUROPEAN COOPERATION FOR SPACE STANDARDIZATION (ECSS) (2008): segmento espaço, segmento usuário e segmento solo; destacando, dentre as atividades do segmento solo, a atividade de simulação contida nas atribuições do CCS.

O desenvolvimento de um simulador operacional envolve diversos stakeholders interessados, os quais podem usar a simulação para diferentes fins, de acordo com sua área de atuação. Um exemplo de uso da simulação por um engenheiro de sistemas é para a análise de uma condição de falha, para a qual é requerida uma busca da fonte do problema, neste caso, deve-se testar diferentes condições e analisar o comportamento do artefato "real". Durante a fase de operação de um satélite, os engenheiros especialistas, engenheiros de sistemas e o gerente da missão, em geral, estão envolvidos com projetos de outros satélites; entretanto, quando ocorre uma falha no satélite em voo, eles podem ser acionados para dar apoio à equipe de operação. Na tentativa de salvar a missão, eles precisam rapidamente compreender as condições operacionais correntes para entender o problema, encontrar um procedimento de correção e validar tal procedimento antes de sua aplicação no satélite real.

O INPE possui uma história de desenvolvimento de simuladores operacionais computacionais, descrita em Ambrosio et al. (2006). Dentre os simuladores de maior porte em desenvolvimento no INPE (AZEVEDO et al., 2012), encontra-se o simulador operacional para o satélite *China-Brazil Earth Resources Satellite* (CBERS) 3&4, denominado Simulador dos satélites CBERS 3&4 (SIMC3), com a justificativa de suporte de atividades de operação reproduzindo as reações do satélite com o máximo de fidelidade possível (TOMINAGA et al., 2012). A interface-usuário (IU) deste simulador é caracterizada como de segunda geração, permitindo ver dados em gráficos, tabelas, explorar e inspecionar valores ou grupo de valores. A interface-usuário da versão atual considera que os engenheiros especialistas e usuários do simulador conhecem profundamente nomes e siglas de parâmetros e suas relações, de forma a validar gráficos e valores limites, entretanto, ainda é difícil manter o entendimento de muitas das mudanças globais ocorridas durante uma rodada de simulação.

Figura 1.1 - Segmentos de uma missão espacial com destaque aos elementos do CCS. Fonte: Adaptado de EUROPEAN COOPERATION FOR SPACE STANDARDIZATION (ECSS) (2008)

Um dos desafios do desenvolvimento de um simulador operacional está relacionado ao projeto da interface-usuário para manipulação e visualização de todos os parâmetros contidos na simulação. A interatividade do usuário em simuladores de satélite é abordada no trabalho (HOFFMANN; PERONDI, 2012), no qual a aplicação de técnicas de *steering* computacional é proposta. Nele são explorados os elementos necessários para dotar a infraestrutura básica de simulação com recursos para fechar a malha com o engenheiro de simulação.

Uma das formas de representar e possibilitar a manipulação destes parâmetros é utilizar representações visuais, como Realidade Virtual (RV), e oferecer uma interface-usuário de terceira geração (BOTEGA; CRUVINEL, 2009). Nesta representação, a metáfora de uso e interpretação dos resultados acompanha a forma de visualizar o sistema simulado, no caso um satélite. Simulações de natureza computacional com homem na malha, podem ser categorizadas de acordo com a taxonimia de geração de interação, definida por Botega e Cruvinel (2009):

- Primeira Geração: Comandos digitados;
- Segunda Geração: Metáfora de uso de janelas, ícones e botões;
- Terceira Geração: Manipulação natural e sistemas 3D.

Elementos visuais que compõem uma interface de terceira geração representam cognitivamente a informação do sistema de forma a melhorar a acumulação de informação, o ajuste dos dados ao conhecimento prévio do usuário e a reconstrução do entendimento (NORMAN, 1993). Norman (1993) descreve que o poder da representação auxilia na exploração e na navegação da informação, de forma que a naturalidade da representação diante do problema auxilia o mapeamento de relações, tornando-a mais fácil, confiável e natural.

1.2 Motivação

A motivação desta dissertação é a utilização de técnicas de RV num simulador operacional, que é considerado o maior e mais complexo simulador no contexto de uma missão espacial, e que estas técnicas possibilitem uma redução do tempo de interpretação das informações apresentadas através de representações tridimensionais.

1.3 Objetivo do trabalho

O objetivo desta dissertação é propor e avaliar a aplicabilidade dos conceitos de RV em simuladores operacionais utilizando as técnicas de múltiplas visões, modelos 3D, árvores cônicas e painéis de interação como metáforas de uso. Avaliar este tipo de abordagem como forma de representação de dados simulados no contexto de simuladores operacionais de satélites.

Espera-se com esta dissertação:

a) Exemplificar uma forma de representação dos elementos de um satélite em

um ambiente tridimensional de RV para compor futuros simuladores, seja para a operação ou para análise de um comportamento parcial, seja este distribuído ou não.

b) Indicar uma tecnologia viável para ser usada na solução e programação do software, direcionada para o alto reuso e componentização de funções.

c) Avaliar o potencial de visualização de dados simulados para redução do tempo e carga cognitiva para trabalhar com eventos que requerem rápida análise de dados em simuladores operacionais.

1.4 Metodologia

A metodologia de trabalho para realização desta dissertação constou de estudos em simuladores de satélites e RV.

Inicialmente, foram estudados conceitos básicos de simuladores operacionais, arquiteturas de simulação, técnicas de representação de dados simulados do comportamento de um satélite e do ambiente espacial, os requisitos de um simulador desenvolvido no INPE em relação a interação do usuário.

Em paralelo, foram estudados conceitos de RV, componentes de interação, paradigmas e técnicas de desenvolvimento e frameworks open-sources disponíveis.

Visando compreender de forma ampla as interações específicas de engenheiros e operadores de satélites, foi desenvolvido um protótipo de um simulador de satélites simplificado. Este protótipo serviu também para demonstrar a independencia entre o simulador e a interface-usuário. Foram descritos os procedimentos de mapeamento dos objetos disponíveis em uma interface exemplo para representações de terceira geração.

A avaliação da nova proposta de interface-usuário para um simulador operacional foi feita de três formas: (i) uma pesquisa previa a para verificar os componentes utilizados e (ii) de uma pesquisa de aceitação com usuários experientes.

A Figura 1.2, resume as atividades realizadas neste trabalho, da etapa de pesquisa dos conceitos, da prototipagem de uma proposta de interface-usuário e da avaliação.

Figura 1.2 - Metodologia aplicada

1.5 Organização do texto

O texto desta dissertação está organizado na sequência dos capítulos conforme apresentado a seguir:

- Capítulo 2: Este capítulo apresenta a revisão bibliográfica dos conceitos de simuladores. Exemplos e trabalhos correlatos de simuladores que foram utilizados de referência nesta solução.

- Capítulo 3: Este capítulo apresenta a revisão bibliográfica dos conceitos de interação e interfaces, do paradigma de uso *top-down* e *botton-up* e o *frameworks* para desenvolvimento de interfaces de terceira geração utilizado.

- Capítulo 4: Este capítulo apresenta o estudo de caso, descrevendo os simuladores operacionais desenvolvidos no INPE, a técnica de modelagem utilizada, um estudo dos requisitos de interface-usuário de um simulador operacional do INPE e de um simulador desenvolvido.

- Capítulo 5: Este capítulo apresenta a descrição do proposta da interface-usuário com RV, descrevendo como foi feito o processo de design, o mapeamento dos elementos de interação, os requisitos, como foram implementados e algumas telas que mostram a processo de navegação nesta proposta de interface. Finaliza com uma discussão de extensão do trabalho, do ferramental utilizado, do uso do software livre e da separação do desenvolvimento em camadas.

- Capítulo 6: Este capítulo apresenta as avaliações de uso, preliminarmente para avaliação dos recursos utilizados e posteriormente com engenheiros stakeholders de simulação operacional.

- Capítulo 7: Este capítulo apresenta as conclusãos, lições aprendidas, contribuições e propostas para trabalhos futuros.

- Apêndice A: Este apêndice descreve o simulador simplificado desenvolvido no contexto desta dissertação.

- Apêndice B: Este apêndice descreve a hierarquia de documentos e blocos em XML que representam o simulador para a interface e para o simulador simplificado.

- Apêndice C: Este apêndice apresesenta os artigos publicados no contexto desta dissertação.

2 SIMULADORES DE SATÉLITES

2.1 Simulação

Simulação é definida como o método de implementar um modelo sobre o tempo (SOUZA; TRIVELATO, 2003). Os elementos relacionados à simulação são (DEPARTMENT OF DEFENSE (DOD), 2010):

- Tempo da simulação: É a representação interna do tempo para a simulação. O tempo da simulação pode ser mais veloz, mais lento ou no mesmo passo que o tempo sideral;
- Relógio da simulação: O contador usado para representar o tempo simulado;
- Entidade de simulação: Elementos do cenário de simulação que são criados e controlados por um simulador pela troca de dados distribuídos
- Cenário de simulação: Conjunto de entidades que interagem para compor uma dada simulação;
- Gerência da simulação: Mecanismo que provê um controle centralizado da simulação;
- Interação: A ação explícita que gera um efeito ou impacto em um elemento da simulação durante a sua execução;
- Formalismo: Método aplicado para capturar a essência de um processo, como: contínuo, discreto ou por eventos.

Um simulador é composto de modelos, definidos como representações físicas, matemáticas ou computacionais de um sistema, entidade, fenômeno ou processo. Modelos analíticos consistem de equações; modelos estruturais são representações de estruturas físicas ou lógicas de um sistema. (SOUZA; TRIVELATO, 2003)

A atividade para gerar um modelo é chamada de modelagem. A modelagem, geralmente, envolve: a aplicação de um padrão, de uma metodologia estruturada para criar uma representação de um sistema, de uma entidade, de um processo ou fenômeno. O ator responsável pela modelagem é chamado de modelador. (SOUZA; TRIVELATO, 2003)

Um simulador pode incluir hardware (HIL - *Hardware in the Loop*), com processadores (PIL - *Processor in the loop*), pessoas (HILT - *Human in the loop*) ou com outros softwares (SIL - *Software in the loop*). (SOUZA; TRIVELATO, 2003)

Dependendo da entidade que interage com a simulação, a simulação pode ser chamada de: *live* quando há pessoas reais em ambientes reais, *virtual* quando há pessoas reais em ambientes simulados e *constructive* quando há pessoas simuladas em ambientes simulados, ou ainda, LVC (*Live, Virtual and Constructive*) que combina os três elementos. (DEPARTMENT OF DEFENSE (DOD), 2010)

A Figura 2.1 mostra os elementos que compõem o conceito geral sobre o simulação.

Figura 2.1 - Elementos de uma simulação.
Fonte: Adaptado das definições de DEPARTMENT OF DEFENSE (DOD) (2010)

2.2 Visualizações em Simulação

Computação gráfica é uma técnica que produz animações virtuais que representam a dinâmica da evolução dos dados, transformando uma simulação de dados numa simulação visual. (FOLEY et al., 1990)

Wenzel et al. (2003) discute tipos de técnicas de visualização que apresentam dados simulados e apresenta uma taxonomia dos elementos gráficos, tanto do ponto de vista dos elementos gráficos em si (ver Figura 2.1), como do ponto de vista da informação relacionada (ver Figura 2.2).

Tabela 2.1 - Taxonomia do ponto de vista de elementos gráficos. Adaptado de Wenzel et al. (2003)

Critério	Especificações						
Dimensão	1D	2D	2 ½ D	3D			
Representação	Simbólica/ Caracter	Simbólica/ Abstrato	Icônica/ Estilizada	Icônica/ Realista	Foto-realista		
Formato da exibição	Texto	Tabela	Gráfico	Diagrama	Fotográfico	Mundo Virtual	Realidade Aumentada
Escala	Nenhuma	Linear	Logaritmica	Exponencial	Categorizada		
Projeção geométrica	Nenhuma	Ortogonal	Oblíqua	Perspectiva			
Dimensão temporal do modelo	Nenhuma	Discreto	Contínuo				
Velocidade de apresentação	Estático	Sem relação	Lento	Tempo Real	Acelerado		
Interação	Nenhuma	Navegação durante apresentação	Interação com modelo gráfico	Interação com modelo simulado			

Segundo Lee (1999), de maneira geral, a comunidade de simulação cria visualizações para verificar e validar o modelo simulado; a execução do modelo é mais importante do que a aparência ou o controle da simulação. Lee (1999) descreve uma maneira de integrar a simulação com a computação gráfica, caracterizando três processos: arquitetura do software, coordenação do software e mapeamento.

- Arquitetura de software: define a estrutura dos elementos, os relacionamentos e a forma como eles são acoplados.

- Coordenação do software: define a gestão da comunicação entre os elementos.
- Mapeamento: define a cena do conteúdo virtual que irá representar o dado simulado e a forma de interação.

Tabela 2.2 - Taxonomia do ponto de vista da informação relacionada. Adaptado de Wenzel et al. (2003)

Critério		Especificações							
Função Primária: Intenção		Identificação	Localização	Correlação	Associação	Comparação	Estrutura e Padrões	Agrupamento	Classificação
Tipo de informação a ser representada		Qualidatativa	Quantitativa	Quantitativa e Qualitativa					
Relacionamento da informação		Independente	Relacoinal	Circular	Hierárquico	Rede			
Codificação da informação	**Nível de medida**	Nenhuma	Discreta nominal	Discreta ordinal	Contínuo ordinal	Intervalo discreto	Intervalo contínuo	Razão discreta	Razão contínua
	Dimensão das variáveis dependentes	Nenhuma	1D	2D	3D	nD			

Segundo Lee (1999), a computação gráfica gera a cena em três etapas: (i) geração dos modelos virtuais (*assets*) que representam o conteúdo desejado, (ii) animação dos modelos no tempo e (iii) renderização para exibição. A simulação possui quatro etapas: (i) modelagem do comportamento do sistema de interesse, (ii) decisão dos caminhos que o modelo deve evoluir, (iii) gerenciamento do tempo da simulação e (iv) aquisição dos resultados dos modelos, como mostrado na Figura 2.2. A integração da computação gráfica e da simulação computacional se dá através do acoplamento dos resultados da simulação com o controle da animação, como simulação visual.

Alguns exemplos de simuladores de satélites com forte apelo visual são: o *System Tool Kit* (STK)(ANALYTICAL GRAPHICS, 2014), o *Celestia*(CELESTIA DEVELOPMENT TEAM, 2010) e o *Orbiter*(SCHWEIGER, 2012).

O STK, da *Analytical Graphics*, é o principal simulador comercial utilizado no projeto de missões espaciais . Este simulador permite modelar, analisar e visualizar sistemas espaciais, de defesa e de inteligência. Provê gráficos, permite modelagem e customização das telas. Também permite integração com componentes externos como Google Earth, Bing e MatLab. A Figura2.3a mostra um exemplo de visuali-

zação de uma operação de mapeamento.

Figura 2.2 - Simulação visual.
Fonte: Adaptado de Lee (1999)

Uma alternativa gratuita de código livre é o *Celestia*, este sistema é focado em astronomia, educação, contudo possibilita a inserção de mais objetos e controle via novos modelos. Possui gráficos, e interação com *mouse*, teclado e *joystick*. A Figura 2.3b mostra um exemplo da tela do Celestia.

Outra alternativa gratuita é o *Orbiter*, para Windows. Este permite realizar simulações de missões do ponto de vista de um piloto, controlando lançamento, inserção de orbita, rendez-vous, etc. O Orbiter possui excelentes gráficos e capacidade de configuração interativa de parâmetros. A Figura 2.3c exemplifica uma visualização do *Orbiter*.

2.3 Simulador Operacional

Um simulador operacional é um simulador de suporte às atividades do segmento solo no pós-lançamento e da equipe de operação, com alta fidelidade nos modelos da espaçonave e nas interfaces do segmento solo, o qual prove simulação da plataforma e da carga útil. Esta categoria de simulação é denominada: Simulador de Treinamento, Operação e Manutenção, pela ESA. (EUROPEAN COOPERATION FOR SPACE STANDARDIZATION (ECSS), 2010)

Simuladores podem ser classificados em diferentes três modalidades, de acordo com as possibilidades de interação no tempo de utilização:

(a) Exemplo de tela do STK

(b) Exemplo de tela do Celestia

(c) Exemplo de tela do Orbiter

Figura 2.3 - Exemplos de saídas visuais dos programas
Fonte: (a) - Retirado de Analytical Graphics (2014), (b) - Retirado de Celestia Development Team (2010) e (c) - Retirado de Schweiger (2012)

- (A) após a modelagem, o condutor insere os parâmetros iniciais e inicia a simulação. Ao fim da simulação o simulador responde com os resultados e permite que o condutor manipule os dados. (DEPARTMENT OF DEFENSE (DOD), 2010)

- (B) após a modelagem, é permitida a visualização da evolução do modelo. O condutor insere os parâmetros iniciais, inicia a simulação e acompanha a evolução do modelo, seja em função do tempo de simulação ou em função da completude de um objetivo estabelecido. Podendo ou não manipular os

resultados após o tempo de simulação. (DEPARTMENT OF DEFENSE (DOD), 2010) (ANALYTICAL GRAPHICS, 2014)

- (C) após a modelagem, é permitida a visualização e interação com o modelo. O condutor insere os parâmetros iniciais, inicia a simulação, acompanha e interage com a evolução do modelo, seja em função do tempo de simulação ou em função da completude de um objetivo estabelecido. Podendo ou não manipular os resultados após o tempo de simulação. (DEPARTMENT OF DEFENSE (DOD), 2010) (MORRIS; ROTHWELL, 2002)

Destas três modalidades de simulação, é possível traçar uma característica do nível de interação do condutor durante o processo de evolução do modelo no tempo de simulação, como mostrado na Figura 2.4. Pela graduação da cor do preenchimento da etapa de evolução indicando a quantidade de envolvimento do condutor durante a fase.

Figura 2.4 - Relação da quantidade de envolvimento do condutor no tempo de simulação.

Os simuladores que possuem grande quantidade de envolvimento do condutor permitem uma grande operabilidade da simulação, interagindo, monitorando e recebendo os feedbacks. Esta característica foi definida como chave por Morris e Rothwell (2002) para distinguir uma categoria de simuladores, chamando-os de simuladores operacionais [1]. Nestes simuladores através de interfaces sensoriais (visual, áudio e

[1] Cabe observar que o termo Simulador Operacional, em um contexto mais amplo, significa que o simulador permite manipulação continua de um condutor (DEPARTMENT OF DEFENSE (DOD), 2010) (MORRIS; ROTHWELL, 2002). No contexto da área espacial, Simulador Operacional é aquele usado na fase pós-lançamento, para as atividades de operação de um satélite.(EUROPEAN COOPERATION

toque), usuários (condutores) podem receber treinamento, validar procedimentos, investigar comportamentos anômalos e responder à situações catastróficas antes do acontecimento e na ausência do artefato real.

FOR SPACE STANDARDIZATION (ECSS), 2010)

3 INTERAÇÃO HOMEM-MÁQUINA

A essência da experiência é saber o que fazer, rapidamente e eficientemente. Um motorista aciona o acelerador, controla o volante, utiliza a alavanca de câmbio. Isso tudo é realizado com prática e habilidade, e continuamente integrando diversas fontes de informação - as cenas das janelas, a conversa com as pessoas, leituras de instrumentos (GPS, velocímetros, conta giros), o som do motor e os procedimentos memorizados para cada situação. (NORMAN, 1993)

Atualmente, tem-se utilizado a terminologia *UX Design* (UX - *User Experience*), como a maneira de denominar o profissional, ou equipe, que desenvolve experiências de uso, neste são necessários conhecimentos de frentes multidisciplinares como psicologia, antropologia, sociologia, semiótica, epistemologia, história, etc (MELBOURNE, 2011). O desenvolvedor de experiência colhe as meta-mensagens necessárias num sistema e as recobre com uma interface que contém a tecnologia para a interação. Saffer (2008) propõe que *UX Designers* necessitam de conhecimentos de design industrial, arquitetura de software, arquitetura da informação, design visual e interativo. A Figura 3.1 apresenta os conhecimentos e as áreas do saber para desenvolvimento de experiência do usuário, como: arquiteturas de computador, design industrial, design interativo, interação homem-computador, design visual e arquitetura da informação.

Neste capítulo são apresentados alguns conceitos sobre gerações e realidades, cognição, representação e usabilidade; e os elementos a serem analisados para a criação de uma interface-usuário com interação, alinhadas com os paradigmas de abstração do acesso à informação. No final do capítulo são apresentados frameworks de desenvolvimento de interface e uma revisão de alguns elementos de interação que serão utilizados na interface-usuário deste trabalho.

3.1 Gerações e Realidades

Gerações

Botega e Cruvinel (2009) definem três gerações de interface, de acordo com a quantidade de dimensões utilizadas nas aplicações, onde a primeira e segunda geração são baseadas em interfaces bidimensionais com interação unidimensional, via comandos e bidimensional no paradigma WIMP e a terceira geração baseada em interfaces tridimensionais, como realidade virtual e realidade aumentada.

Nielsen (1993), contudo, defende a categorização em cinco gerações:

Figura 3.1 - Disciplinas relacionadas à experiencia do usuário.
Fonte: Saffer (2008)

- Geração 0 - até 1945 - Chamada pré-histórica - O paradigma de interação era manipulação direta no hardware.
- Geração 1 - de 1945 a 1955 - Chamada Pioneira - O paradigma de interação era baseada em batchs de comandos.
- Geração 2 - de 1955 a 1965 - Chamada Histórica - O paradigma de interação era baseado em linguagens de comandos.
- Geração 3 - de 1965 a 1990 - Chamada Tradicional - O paradigma de interação era baseado em menus hierarquizados e preenchimento de formulários.
- Geração 4 - de 1980 a 1995 - Chamada Moderna - O paradigma de interação predominante era o WIMP.

- Geração 5 - de 1995 à atualmente - Chamada Futuro - O paradigma de interação seria baseado em interfaces naturais.

Correlacionando as definições de Botega e Cruvinel (2009) e Nielsen (1993), a primeira geração definida por Botega e Cruvinel (2009), inclui as gerações 0,1,2 de Nielsen (1993), a segunda é correlata a geração 4 e a terceira é correlata a geração 5.

É interessante mencionar que a definição destas gerações não é simplesmente baseada no tempo, mas na forma como as interações são realizadas. Ivan Sutherland em 1963 já tinha desenvolvido o primeiro sistema gráfico interativo, que interpretava desenhos como entrada e realizava associações com topologias conhecidas (SUTHERLAND, 1964). Todas as gerações poderiam coexistir atualmente, contudo eles seriam utilizados para funções e meios diferentes (MYERS, 1998). A diferença temporal apenas reflete os requisitos do poder computacional e dos fatores culturais disponíveis para executar as interfaces (OH et al., 2011).

Uma relação entre os componentes de uma interface e a correspondente geração pode ser vista na Figura 3.2. Da abstração dos elementos manipulados, os sistemas podem ter comandos específicos ou mais próximos do problema. Do poder computacional, os sistemas podem variar pelo esforço de entender a interação. A quantidade de hardware específico necessária pode variar de teclado, monitor a cavernas, realidade virtual e aumentada e reconhecimento de gestos. Da metáfora de uso o sistema fica mais próximo da metáfora do problema quanto mais sofisticado o sistema fica.

Essa categorização possibilita dizer que quanto mais elementos da Primeira Geração a interface tiver, mais a interação será *bottom-up*, exigindo maior conhecimento específico e detalhado para se entender um contexto e em contrapartida, quanto mais elementos de Terceira Geração a interface tiver, mais a interação será *top-down*, partindo-se de conhecimentos gerais para se entender um contexto.

Realidades

Stephenson (1992) escreveu a novela de ficção científica Snow Crash, na qual os seres humanos e os seres\elementos com inteligência artificial interagiam em um mundo tridimensional baseado na metáfora do mundo real, chamando este mundo de Metaverse, definido como: ”coletivo de espaços compartilhados, onde o ser era virtualmente aprimorado”.

Figura 3.2 - Classificação da geração da interface relacionando a abstração, o poder computacional, o hardware, a metáfora e a direção da interação.
Fonte: Adaptado de Cerqueira et al. (2013)

O conceito de Metaverse foi a base da idealização de um contínuo da realidade, como mostrado na Figura 3.3, onde o mundo real é acrescido de elementos virtuais tornando-se um ambiente de realidade misturada chamado realidade aumentada e o mundo virtual é acrescido de elementos reais tornando-se também um elemento de realidade misturada chamado agora de virtualidade aumentada (MILGRAM et al., 1995).

Figura 3.3 - Contínuo Realidade-Virtualidade de Milgram
Fonte: Milgram et al. (1995)

Porém, à época, Milgram não percebeu que o contínuo é uma interpretação errada, e o que acontece é uma mudança das interfaces de manipulação possibilitando a construção de um novo diagrama, mostrado na Figura 3.4, no qual elementos do mundo real e o mundo virtual se cruzam com elementos computacionais (sensores e atuadores), habilitando vários mundos, como: mundo real tangível, realidade aumentada, realidade virtual, mundos virtuais distribuídos, mundos ubíquos e realidade cruzada. (KIRNER et al., 2012)

Figura 3.4 - Diagrama de Kirner de múltiplas realidades.
Fonte: Adaptado de Kirner et al. (2012)

As definições dos mundos representados pelo diagrama estão a seguir:

- Mundo Real é o mundo, onde o usuário está imerso naturalmente, sem o apoio de nenhum artefato tecnológico, interagindo diretamente com o ambiente, através de seu corpo e de suas faculdades mentais, explorando sua capacidade cognitiva multissensorial, em toda sua plenitude.

- Realidade aumentada (*Augmented Reality*), segundo Kirner (2011) é uma interface baseada na sobreposição de informações virtuais geradas por computador (envolvendo imagens estáticas e dinâmicas, sons espaciais e sensações hápticas) com o ambiente físico do usuário, percebida através de

dispositivos tecnológicos e usando as interações naturais do usuário, no mundo físico.

- Realidade virtual (*Virtual Reality*), segundo Kirner (2011) é uma interface computacional que permite ao usuário interagir, em tempo real, em um espaço tridimensional gerado por computador, usando seus sentidos, através de dispositivos especiais. Podendo ser totalmente transportado para o mundo (imersivo), ou parcialmente transportado (não imersivo).

- Mundos virtuais distribuídos (*Shared Virtual Worlds*): segundo Schroeder (2008) são ambientes ou sistemas que usuários interagem com outros participantes como se estivessem no mesmo ambiente.

- Mundos ubíquos (*Ubiquitous Worlds*), para Weiser (1993) que definiu computação ubíqua (*UbiComp*), é a integração entre sistemas móveis, com a presença do usuário distribuída entre elementos computacionais, onde a tecnologia pode ser perceptível ou não, com capacidade de interpretar as ações do ambiente e do usuário e adaptar a interface humana.

- Realidade Cruzada (*Cross-Reality*), segundo Kirner et al. (2012) é um ambiente de realidade misturada ubíqua que vem da fusão de uma rede de sensores e atuadores (que coletam e enviam dados relacionados ao mundo real) com mundos virtuais compartilhados, usando a interface da realidade aumentada, onde a troca de informação é bidirecional entre o mundo real e o mundo virtual. Podendo ser de duas formas: sobreposta (*Overlapped Cross Reality*), onde elementos reativos virtuais e reais são sobrepostos ou não sobrepostos (*Non Overlapped Cross-Reality*) onde elementos virtuais e reais não se sobrepõem.

3.2 Cognição, Representação e Usabilidade

Cognição

Norman (1993) expõe que a experiência é baseada em dois tipos de cognição: experimental e refletiva.

A cognição experimental é definida como padrões de informação que são percebidos e assimilados e geram respostas apropriadas sem atraso ou esforço aparente. Neste tipo de cognição a resposta vem automaticamente, processada pelo subconsciente, age como um reflexo - chamada *data driven*.

A cognição refletiva é definida como a habilidade de inferência de resultados a partir do conhecimento armazenado, seguindo uma cadeia estruturada de considerações. Neste tipo de cognição a resposta é lenta e trabalhosa, processada pelo consciente, após um período de reflexão - chamada de *conceptually driven*.

Representação

Norman (1993) expõe que a cognição vem da abstração e da representação, definida como a habilidade de representar percepções, experiências e pensamentos em um meio diferente do que ocorreu, montando marcas e símbolos que o representam.

As representações possuem relações, estruturas e consistências com o mundo. Uma boa representação captura os elementos essenciais do evento e os transforma em artefatos cognitivos de interação. Sejam estas representações de experiência que provêm maneiras de interagir com o mundo; ou refletivos que provêm maneiras de interagir com as representações.

As duas maiores tarefas do usuário numa interface, segundo Norman (1993), são:

- Identificar a informação relevante;
- Entender a conclusão desejada.

E para isso são indicados dois princípios, que auxiliam na criação de representações naturais ao usuário (NORMAN, 1993):

- Princípio da naturalidade: cognição experimental é ajudado quando as propriedades da representação combinarem com as propriedades do objeto a ser representado.
- Princípio da percepção: Representações espaciais e perceptivas são mais naturais e devem ser preferidas a não perceptivas, não espaciais, apenas se o mapeamento entre a representação e seu significado for natural, isto é, análogo ao real.

Usabilidade

Quando os primeiros computadores pessoais começaram a ser vendidos, um termo publicitário surgiu para melhorar as vendas: *user friendly* (Traduzindo literalmente:

amigável). Este termo não corresponde à realidade conceitual, pois um sistema pode ser "amigável"para um perfil de stakeholders e totalmente desastroso para outro, sendo difícil de ser utilizado como uma métrica de qualidade do sistema.

Uma métrica de aceitação prática é a utilidade (*usefulness*). Utilidade diz se o sistema alcançou os objetivos desejados. Segundo Grudin (1992) a utilidade pode ser divida em: utilização (*utility*) e usabilidade. Utilização é a questão da funcionalidade do sistema e usabilidade é como os usuários manipulam a funcionalidade.

Nielsen (1993) utiliza o termo usabilidade (*usability*) para descrever as características que um sistema deve ter para torná-lo aceito pelos usuários. A aceitação de um sistema, como pode ser visto na Figura 3.5, é a combinação de sua aceitação social, referente ao cenário social e ao cenário histórico que está sendo utilizado, e da aceitação prática, a qual se ramifica em vários critérios, como: custo, utilidade, compatibilidade, confiança, etc. Tais critérios são usados como métricas para avaliação de uma interface-usuário.

Figura 3.5 - Atributos da aceitação de um sistema.
Fonte: Adaptado de Nielsen (1993),INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (2000) e Grudin (1992)

Usabilidade possui várias definições. A definição estabelecida pela (INTERNATIONAL

ORGANIZATION FOR STANDARDIZATION, 2000) diz que: "usabilidade é a capacidade que um sistema interativo tem de oferecer ao usuário, em determinados contextos de operação, facilidades para a realização de tarefas, de maneira eficaz ("resolve o problema"), eficiente ("resolve rapidamente") e satisfatória ("resolve confortavelmente")".

Para Nielsen (1993) usabilidade possui cinco objetivos:

- Facilidade de aprendizado (*Learnability*): o sistema deve ser fácil de aprender de forma que o usuário rapidamente comece a trabalhar com o sistema.
- Eficiência (*Efficiency*): O sistema deve ser eficiente, assim que o usuário aprender a utilizá-lo, este deve ter a maior produtividade possível.
- Facilidade de memorização (*Memorability*): O sistema deve ser fácil de ser lembrado, de forma que o usuário casual possa retomar o uso mesmo após longos períodos e não precise aprender tudo novamente.
- Segurança (*Errors*): O sistema deve ter poucos erros, de forma que os usuários se deparem com poucos erros e caso aconteçam devem ser reparados com facilidade.
- Satisfação subjetiva (*Satisfaction*): O sistema deve ser agradável para o usuário, de forma que os usuários estejam subjetivamente satisfeitos ao usá-lo.

3.3 Interação e Interface

Interação possui diversas definições. Noble (2009) define interação como sendo ”a troca de informação entre dois ou mais participantes”; já Crawford (2004) a define como: ”um processo iterativo de escutar, pensar e falar entre dois ou mais participantes”.

Um conceito importante relacionado a interação, é o *feedback loop* (NORMAN, 1993). Trata-se da ideia de uma entidade comunicando-se com ela mesma e verificando sistemas regulatórios internos e externos. Uma parte do *feedback* de uma interface-usuário é a troca de mensagens, a qual é realizada através das interfaces.

A Figura 3.6 esquematiza os elementos da interação, o *feedback loop*, as mensagens, os participantes e a camada de interface.

As mensagens podem ter diferentes naturezas: texto, fala, cores, feedback visual ou físico mecânico. Entender como o sistema vê as mensagens do usuário e como o usuário vê as mensagens do sistema é a parte mais difícil no design de interface-usuário de softwares interativos. Todas as interações possuem certo vocabulário, símbolos que são compartilhados com os participantes da interação. Noble (2009) expõe que o agrupamento destes símbolos torna possível a categorização da interação, como: manipulação física, por códigos, mouse, presença, háptica, gestos e voz.

Figura 3.6 - *Feedback loop*.
Fonte: Adapatado de Norman (1993)

O design de uma interface, segundo Dennis et al. (2005), é guiado por seis princípios: (i) layout - áreas com diferentes propósitos; (ii) informações contextualizadas (*content awareness*) - conteúdo relacionado com o local da tela; (iii) aparência (*aesthetics*) - interface convidativa; (iv) experiência (*user experience*) - fácil de usar e aprender; (v) consistência (*consistency*) - usuário prevê reações da interface; (vi) trajeto mínimo (*minimal user effort*) - simplicidade de uso.

No design de uma interface interativa de terceira geração, a forma de definir a interface é baseada em pesquisa de analogias de uso e da descrição do seu comportamento. Para isso, usa-se de metáforas e *storyboards*.

Metáfora

No desenvolvimento de uma interface, o estudo da metáfora define como ela funcionará. A metáfora de uma interface é o conceito do mundo real que é usado como modelo do sistema computacional, mimetizando suas formas de uso. A metáfora ajuda o usuário a entender o sistema e prever as suas reações. A metáfora pode ser explícita, sendo igual ao conceito do mundo real, ou implícita, na qual o conceito do uso é embutido de maneira transparente. De qualquer forma, a metáfora deve ser óbvia para não requerer pensamento, aproveitando a cognição experimental. (DOMINGUES, 2001)

Metáfora, segundo o dicionário Aurélio, é

> ”o topo que consiste na transferência de uma palavra para um âmbito semântico que não é o do objeto que ela designa e que se fundamenta numa relação de semelhança sub entendida entre o sentido próprio e o figurado”.

Storyboard

Uma das técnicas de design de interfaces é a utilização de *storyboards* (USABILITY BODY OF KNOWLEDGE, 2014) (MICROSOFT, 2014) (KIM, 2005). *Storyboards* ilustram as interações entre o usuário e o produto em formato de narrativa, que inclui uma série de desenhos (*sketchs*) que contam a história da interação. Em design, um storyboard especifica como uma interface reativa muda em reação às ações do usuário. Landay e Myers (2001) indicam algumas vantagens e desvantagens desta técnica, como:

- Vantagem: possui mais significado do que fluxogramas ou diagramas técnicos.
- Vantagem: Vago o suficiente para o cliente poder entender o sistema e fazer a coleta de requisitos.
- Vantagem: Não requer conhecimento de computação ou artístico.
- Vantagem: Provê rápido design de ideias e complementa cenários verbais.
- Vantagem: Provê uma maneira de complementar o diagrama de casos de uso em coerentes descrições das ações.
- Vantagem: Ajuda ao stakeholders visualizar a experiência.

- Desvantagem: Relutância dos desenvolvedores por considerarem suas capacidades de desenho ruins.
- Desvantagem: Confusão com o storyboard de filmes, que é linear, softwares não são lineares.
- Desvantagem: Não é prático para design detalhado.

A construção de storyboards pode ser feita de diversas maneiras: no papel, em quadros, *flipcharts*, rascunhos digitais ou em softwares de reconhecimento que a partir de desenhos assistidos, criam a interface, como o *Indigo Studio*[1] , *PowerStory*[2] ou *Silk*[3].

A Figura 3.7, ilustra *sketches* de um *storyboard* aplicado para navegação de um *browser* de arquivos.

A técnica de storyboards é útil na alocação de funções, segundo Kim (2005) é muito utilizado para criação de conceitos, para fornecer contribuições visuais à especificação, para simulações de comportamento de interação sem se preocupar com implementação, coletar os primeiros feedbacks de uma análise de interface de software realizada por stakeholders.

No desenvolvimento de um software interativo, depois de completado o storyboard, o time de design extrai os designs e as recomendações alocando-os em outros documentos, se necessário o storyboard é refinado e reavaliado (USABILITY BODY OF KNOWLEDGE, 2014).

3.4 *Framework* para desenvolvimento de interface - openFrameworks

Para o desenvolvimento de interfaces existem diversos *frameworks* que facilitam a criação de interfaces-usuário, as mais tradicionais para as interfaces da segunda geração de Botega atualmente, são: .NET[4], Qt[5] e plug-ins para IDEs como WindowBuilder[6] para o Eclipse. A página da *Wikipédia List of widget toolkits*(WIKIPEDIA, 2014) possui uma lista vasta de frameworks ou pequenos códigos auxiliares para geração de interfaces com usuário. oF está disponível em está disponível em: http://www.openframeworks.cc/.

[1]Disponível em http://www.infragistics.com/products/indigo-studio
[2]Disponível em http://power-story.com/
[3]Disponível em http://dub.washington.edu:2007/denim/research/
[4]Disponível em http://msdn.microsoft.com/pt-br/vstudio/aa496123
[5]Disponível em http://qt-project.org/
[6]Disponível em https://developers.google.com/java-dev-tools/wbpro/

Figura 3.7 - Exemplo de *sketches* de um *storyboard*.
Fonte: Microsoft (2014)

Um *framework* de desenvolvimento de interfaces de terceira geração é o openFrameworks (oF). oF é uma coleção de códigos criados para artistas e designers trabalharem com design interativo. oF é desenvolvido principalmente por Zach Liberman, Theo Watson, Artuno Castro e Chris O'Shea e hoje recebe colaborações de toda comunidade. (NOBLE, 2009)

Deve-se considerar utilizar oF, para Noble (2009), quando:

- O projeto renderiza muitos gráficos 3D, eou;
- Utilizar muita visão computacional, eou;
- Controlar equipamentos, como, por exemplo, o ARDUINO.

oF inclui uma série de bibliotecas, como: openGL, FreeType, FreeImage, QuickTime, POCO

O código é compatível com várias arquiteturas e vários ambientes de desenvolvimento, como os cinco sistemas operacionais (Windows, OSX, Linux, iOS e Android) e quatro IDEs (*XCODE*, *Code::Blocks*, *Visual Studio* e *Android Developer Tools*). Deve-se ainda ressaltar que é *open-source* distribuído sob a Licença MIT (OPEN SOURCE INITIATIVE, 2014).

Uma das vantagens do oF é o fato da comunidade poder desenvolver componentes ou funcionalidades que possam ser inseridas no oF. Estes componentes são chamados de add-ons. A página oficial[7] de contribuições do oF divide os add-ons nos seguintes seguimentos: animação, pontes, visão computacional, gráficos, interfaces de usuário, interfaces de hardware, iOS, inteligencia artificial, física, som, tipografia, utilidades, acesso a camera e a rede.

Os recursos que o oF oferece para o design de uma interface são separados em motor de interação (*Interaction Engine*), motor de saída (*Output Engine*) e link de acesso à rede (*Comm Link*) e são apresentados na Figura 3.8, de acordo com visão do autor desta dissertação apresentado em (CERQUEIRA et al., 2012).

3.4.1 Outros *frameworks* disponíveis para desenvolvimento de interfaces de terceira geração

Processing é um *framework* desenvolvido para Java, *open-source* e tem um forte apelo educacional. É baseada em programação via código e possibilita a utilização de openGL. É possível conectar bibliotecas Java para complementar as aplicações e aumentar as possibilidades de desenvolvimento. Uma das vantagens é a fácil exportação de projetos para web. *Processing* está disponível em: http://processing.org/

Adobe Flash é uma possibilidade bem conhecida para desenvolvimento criativo, baseado em *ActionScript*. É muito utilizado por ter uma estrutura familiar com ferramentas de design como o textitPhotoshop e *Illustrator*. A IDE de desenvolvimento da Adobe é paga, contudo existe a opção do *Flash Builder*, grátis para estudantes e professores. *Flash* é completamente exportável para web, porém isso impacta no desempenho de acesso ao hardware quando comparado a outras tecnologias. *Flash Buider* está disponível em: http://www.adobe.com/products/flash-builder-family.html

Unity é um motor de jogos, grátis para desenvolvimento em Windows e pago para dispositivos móveis, e como motor de jogos permite desenvolvimento com grafo de cena, editor de níveis, etc. A programação pode ser realizada em C#. Não é open-

[7] http://ofxaddons.com/

source, mas deve ser fortemente considerada dependendo do projeto, pela velocidade de desenvolvimento. Unity está disponível em: http://unity3d.com/

Figura 3.8 - Arquitetura de elementos do oF disponíveis para uma interface interativa. Fonte: Cerqueira et al. (2012)

Cinder é um ferramental *open-source* e *cross-plataform* disponível para Mac, iOS e Windows. A programação é baseada em C++ e possui diversos elementos prontos: como manipulação 3D, áudio, vídeo, rede, processamento de imagem, etc. *Cinder* está disponível em: http://libcinder.org/features/

3.5 Técnicas de interação de RV

As interfaces de terceira geração possuem uma vasta gama de técnicas de interação. No contexto desta proposta de desenvolvimento foram escolhidos algumas que se encaixavam nos requisitos (Capítulo 4), que eram correlatas à experiência do autor, encontradas em artigos de manipulação de dados, e em interfaces pesquisadas na etapa de design. Assim, nesta seção apresentam-se as técnicas de interação que serão utilizadas na proposta de interface do Capítulo 5.

3.5.1 Modelos 3D

Seres humanos estão acostumados a visualizar elementos em três dimensões, então uma forma significativa de criar interação é utilizar o representações tridimensionais, que espelham a realidade do mundo real. Profundidade, escala e distância passam a se tornar elementos do *feedback loop* que o usuário pode observar e manipular.

Uma biblioteca que pode ser utilizada para geração de espaços 3D é o OpenGL(WOO et al., 1999), ela é uma API baseada na linguagem C, cuja função é esconder a complexidade de comunicação com os drivers das placas de vídeo, através de comandos que realizam as operações.

Os objetos 3D, podem ser modelos criados num software de autoria, como 3DSMax[8], Blender[9], Google SketchUp[10]. Estes modelos são sólidos e possuem as seguintes propriedades: (FOLEY et al., 1990)

- Pivô: distância do objeto a um sistema de coordenadas próprio.
- Dimensão: um sólido realizável, intrinsicamente tridimensional, porém se duas dimensões são muito grandes este aparenta bidimensional, se três dimensões são muito grandes este aparenta tridimensional.
- Representação: formas de visualização do modelo

  Aramada: os objetos são descritos pelo seu conjunto de arestas das bordas do objeto.

  Por faces: os objetos são descritos pelas suas superfícies limitantes, supostamente fechadas e orientáveis.

[8]Link para download: http://www.autodesk.com/products/autodesk-3ds-max/overview
[9]Link para download: http://www.blender.org/
[10]http://www.sketchup.com/

A utilização das cores, segundo Foley et al. (1990), exerce uma ação tríplice: a de impressionar, a de exercer e a de construir. A cor é vista quando esta impressiona fisicamente a retina, é sentida através da emoção, e seu significado simbólico é construído. O OpenGL permite descrever as cores através dos parâmetros de intensidade dos canais de cores RGB e da sua transparência, chamado de canal alfa.

O oF possui um componente chamado ofGraphic, responsável por lidar com as transformações, e possui um componente chamado ofAssimpModelLoader, um wrapper da biblioteca ASSIMP(NOBLE, 2009), responsável por lidar com os modelos de objetos e realizar animações.

3.5.2 Audio

A experiência primária de interface com computadores é a visual, contudo não é a única função cognitiva e de percepção que os seres humanos possuem. A audição é um sentido que é geralmente ignorado no design de interfaces com o usuário. Para Noble (2009), a sonorização de uma aplicação deve ser casada com o elemento visual, de forma a facilitar a fixação do conceito mental e o entendimento do ambiente ao redor da imagem.

O som pode influenciar a percepção dos objetos e informações, e podem prover *feedbacks* diretos como:

- sinal de reconhecimento ou ícone sonoro: Uma ação que não é possível de ser indicada visualmente. Ex.: som de clicks ao "pressionar"um botão numa tela touchscreen.
- uma mensagem associativa ou têxtil: Um som, ou jingle que reflita uma assinatura musical.
- uma ajuda à simulação do mundo real: Sons podem transmitir informações espaciais. Quando um usuário é inserido num ambiente virtual ele pode experimentar ecos, posições, e outros fatos que auxiliam na obtenção de informação.
- um produto ou uma ação indireta: Ferramentas que provêm uma visualização do som, permitindo criar desenhos e músicas.
- uma forma de contar uma informação sobre o objeto fazendo som: esta é parecida com a ajuda à simulação, porém neste caso é quando a fonte de

som é indicada no próprio som. Ex.: Garfo batendo no vidro, fazendo um som de batido no vidro.

- um disparo emocional: Trilha sonora utilizada para conduzir o clima da aplicação e das interações, e como elas dramaticamente mudam de acordo com a natureza do som e da informação que representam.

Noble (2009) lista as maneiras mais comuns de utilizar som como um driver de interação:

- O usuário cria um som que outra pessoa pode ouvir.
- O usuário cria um som de entrada utilizando uma ferramenta.
- O usuário cria um som que é a entrada de um sistema.
- O usuário conversa ou diz palavras para criar a entrada de um sistema.

oF possui uma classe chamada ofSoundPlayer que oferece facilidades de execução de sons. Esta é compatível com diversos formatos de som (WAV, MIDI, MP3, XMA e OGG) e permite posicionar o som em 3D, fornecendo um posicionamento tridimensional da fonte de som para o usuário. Outra facilidade explorada pelo oF é a utilização do sistema de fala sintética do sistema operacional ou de um software instalado, como Balabolka[11] , para realizar leituras dos procedimentos ou narrar a utilização do sistema.

3.5.3 Múltiplas visões e mapas

Na vida real, assim como nos jogos, são utilizados diversos pontos de vistas simultâneos, como exemplo, tem-se uma interface para uma cabine de avião, nela o piloto olha para a cena espacial do mundo, e para os diversos pontos de vistas dos equipamentos. A Figura 3.9 ilustra possiblidades de visões diferentes com (a)(d) e (c) do *Microsoft Flight Simulator* e (d) do *Google Earth Flight Simulator*.

Da mesma forma, mapas e mini-mapas provêm visões macro do sistema, contextualizando toda a situação e possibilitando a navegação através dos objetos da cena. A metáfora de mapas, como na abordagem *world-in-miniature* (STOAKLEY et al., 1995), é inspirada em mapas e globos. Mini-mapas representam a visão atual do mundo virtual no qual o usuário está inserido, replicando e correspondendo aos objetos que podem ser selecionados e manipulados. (KIM, 2005)

[11]Link para download: http://www.cross-plus-a.com/balabolka.htm

(a) HUD View of Microsoft Flight Simulator

(b) Pilot View of Google Earth Flight Simulator

(c) 3th Person View of Microsoft Flight Simulator

(d) Map View of Microsoft Flight Simulator

Figura 3.9 - Exemplo de múltiplas visões.
Fonte: Cerqueira et al. (2013)

Aplicativos de software permitem a montagem de múltiplas visões do mesmo elemento em uma única interface-usuário. Ferramentas de CAD oferecem diferentes câmeras de visualização. Em computação gráfica essas visões são chamadas de *viewport*. *Viewports* são retângulos em 2D que definem uma área da superfície da tela, que gerará a cena. (FOLEY et al., 1990)

No oF é possível criar *viewports* em diferentes pontos da janela de exibição. No oF o *viewport* é tratado como um objeto do tipo ofRectangle, que é passado como argumento na hora de desenhar os elementos na tela, como mostrado no exemplo da Figura 3.10.

Figura 3.10 - Exemplo de viewports do oF

3.5.4 *RDT - Reconfigurable Disk Tree*

As interações tridimensionais com grandes conjuntos de dados exige uma estrutura que mantenha a estrutura semântica que represente a hierarquia dos elementos. Eid (2005) expõe que vários algoritmos e layouts em 2D são eficientes, contudo, observando a capacidade de exibição de nós de dados, é extremamente aconselhada uma visualização em 3D. A seguir estão exemplos dos layouts utilizados para visualização de grandes conjuntos de dados em 3D, catalogados por Eid (2005):

- Árvores hiperbólicas: mapeia estruturas hierárquicas em uma função hiperbólica ou num espaço hiperbólico. Figura 3.11a
- Mapa de árvores: mapeia estruturas em nós em formatos de retângulos que são subdivididos recursivamente. Figura 3.11b
- Árvores cônicas: mapeia estruturas hierárquicas numa árvore tridimensional. Figura 3.11c.

Jeong e Pang (1998) apresentaram uma técnica de visualização chamada RDT (*Reconfigurable Disk Trees*), nesta, cada nó é associado com um centro de um disco com seus nós filhos espalhados pela circunferência; Jeong e Pang (1998) discutem que esta técnica reduz a quantidade de nós ocludidos, aceleram o processamento computacional, o tempo de pesquisa do usuário, facilitam projeção planar da estrutura e a manipulação em 3D.

Figura 3.11 - Alguns tipos de distribuição de dados em árvores. (a) hiperbólica, (b) mapa e (c) árvore cônica.
Fonte: Adaptado de Eid (2005)

Nós de uma RDT podem ter seus filhos expandidos ou reduzidos, para isso deve-se repetir o procedimento a partir da etapa numero dois. A Figura 3.12a e 3.12b mostra um diagrama apresentado no artigo de dois casos, um com a somatória dos raios das circunferências dos filhos maior que o raio da circunferência do nó pai, e outro com a somatória menor. No algoritmo proposto por Jeong e Pang (1998), da Figura 3.12c é mostrado o ponto onde essa informação é levada em conta para decidir o posicionamento dos nós filhos no raio do disco pai.

3.5.5 Painéis

A tendencia das interfaces é o minimalismo de elementos de tela. As ações sobre os objetos devem ser contextualizadas ao objeto selecionado. A resposta a esta ação, em grandes quantidades de dados deve ser um elemento que seja mais padronizado e autoconfigurável possível, de forma a eliminar menus. Alguns sistemas operacionais

estão adotando áreas padronizadas de informação, que são customizadas ao contexto, como o *Windows 8 Charm bar*, mostrado na Figura 3.13.

(a) Caso onde os somatório dos raios filhos é maior que o raio do pai.

(b) Caso onde o somatório dos raios filhos é menor que o raio do pai.

Procedure $Calculate_radius_3D(i)$

Input node i

Output radius rad_i and outer-radius out_rad_i for node i

1 if $num_child_i = 0$
  $out_rad_i = M_rad$
  $rad_i = 0$
  return
2 for each child k of node i, execute $Calculate_radius_3D(k)$
3 $inner_sum = 2\sum_k out_rad_k$
4 $max_child_radius = max_k(out_rad_k)$
5 if $inner_sum < 2\pi \cdot max_child_radius$
  $rad_i \approx max_child_radius$ /* case 2 */
else
  $rad_i \approx inner_sum/2\pi$ /* case 1 */
6 $out_rad_i \approx rad_i + max_child_radius$

End Procedure

(c) Algoritmo de calculo dos raios.

Figura 3.12 - Árvore de discos reconfiguráveis
Fonte: Jeong e Pang (1998)

O oF possui um add-on para o desenvolvimento de painéis, chamado ofxUI[12], mostrado na Figura 3.14. Este possui elementos para visualização de texto, de listas, *sliders* com valores, caixas de seleção, gráficos de elementos e botões que podem ser aninhados de forma a construir blocos de interação para cada elemento selecionado na tela.

[12]Disponível em: https://github.com/rezaali/ofxUI

(a) Windows Context (b) Music Player Context (c) Netflix Context

Figura 3.13 - Exemplo de painéis com informações sensíveis ao contexto

Figura 3.14 - Exemplo dos elementos de interação de um painel montado com o componente ofxUI

4 O SIMULADOR SIMC3

Este capítulo descreve o estudo dos simuladores operacionais desenvolvidos no INPE e detalha a forma de modelagem adotada e os requisitos de interface definidos para o Simulador dos Satélites CBERS 3&4. Além disso, o capítulo apresenta e discute as características do ANALIX, uma versão do simulador desenvolvida para validar conceitos a serem adotados no SIMC3, aqui usado como referência.

4.1 Simuladores operacionais de satélites do INPE

O INPE possui um histórico de desenvolvimento de simuladores operacionais, utilizados para prover cenários que poderiam acontecer durante a fase de operação do satélite. Os simuladores operacionais desenvolvidos no INPE no período de 1991 a 2005 são apresentados em (AMBROSIO et al., 2006). Os primeiros simuladores operacionais atenderam as seguintes missões espaciais do INPE:

- missão de coleta de dados dos SCDs - foi desenvolvido o simulador SIMS, iniciado em 1991, contendo informações do ambiente espacial, estações terrenas e simulações dos subsistemas do satélite.
- missão de imageamento da Terra do primeriro satélite CBERS-1 - foi desenvolvido o SIMCS, iniciado em 1997, para dar suporte à atividade de treinamento dos operadores, com detalhes como: telecomandos de atitude e orbita.
- a missão do micro satélite científico da parceria França (CNES)-Brasil (INPE), foi projetado o FBMSIM, iniciado em 2002, porém passou por diversos desafios de especificação do comportamento, e acabou descontinuado junto com a missão FBM.
- Atualmente o INPE desenvolve o simulador SIMC3, um simulador de alta fidelidade. A justificativa para o desenvolvimento deste foi a necessidade de avaliar procedimentos operacionais nominais e anormais antes da execução no satélite real. O simulador apoia as atividades operacionais e ajuda na análise de comportamentos que podem ocorrer durante o ciclo de vida do satélite após seu lançamento. O escopo de uso deste simulador cobre a reprodução das reações dos subsistemas dada a influência de outros subsistemas e componentes ambientais.(TOMINAGA et al., 2012). Este simulador inclui a reprodução em software, do satélite, das estações terrenas e do ambiente espacial.

A Figura 4.1, mostra um histórico dos simuladores apontando a época do desenvolvimento, a linguagem utilizada, o grau de fidelidade e a satisfação do usuário com relação ao sua utilidade.

Figura 4.1 - Resumo das características dos simuladores.
Fonte: Adaptado de Ambrosio et al. (2006).

4.2 Especificação do comportamento do satélite

No SIMC3, o comportamento dos subsistemas é documentado de uma forma que os engenheiros especialistas e a equipe de desenvolvimento do simulador os entendam. O modelo comportamental elétrico do satélite, foi divido em subsistemas. Cada subsistema possui seu próprio documento, chamado *Subsystem Model Specification*, o qual segue um padrão de especificação (TOMINAGA; AMBROSIO, 2012) e uma rotina de validação. A Figura 4.2, mostra o fluxo de desenvolvimento aplicado na elaboração da especificação dos modelos do SIMC3. Os documentos usados como fonte de informação sobre o Subsistema são lidos pelos modeladores e os engenheiros responsáveis são entrevistados para dirimir dúvidas e acrescentar informações ás vezes omissas sobre o comportamento do subsistema. Os modelos, devidamente documentados, são avaliados em reuniões que incluem o arquiteto de simulação, os modeladores, os engenheiros responsáveis pelo subsistema e o gerente da missão. Alterações necessárias são realizadas e revisadas novamente. No aceite, o modelo é repassado para o grupo de desenvolvimento. (TOMINAGA et al., 2012)

O documento de especificação do comportamento do subsistema de cada subsistema segue as definições dadas em (TOMINAGA; AMBROSIO, 2012), o qual compreende três níveis: subsistema, equipamento e componente; e possui três naturezas de representação:

Figura 4.2 - Fluxo de desenvolvimento do documento de especificação do modelo. Fonte: Tominaga et al. (2012).

- Natureza Física: Descreve o subsistema (S/S) em termos dos equipamentos (EQ) que o constitui, unidades, partes e a localização de montagem. Os equipamentos são mostrados em duas formas: primeiro em uma árvore , seguido de um diagrama de blocos interligando os equipamentos.

- Natureza Lógica: Descreve os estados lógicos e transições que definem o estado de um subsistema em um dado momento. Esta é representada por um diagrama lógico que apresenta chaves lógicas (SW),os telecomandos (TC) e as telemetrias (TM). Para expressar o conjunto de estados das chaves e expressar os modos de operação (OM) do subsistema, foi definido estado de trabalho para os níveis de subsistema e equipamento.

- Natureza de Potência: Esta descreve o subsistema em termos de distribuição de potência e balanço energético. Esta é subdividida em elétrica (PW), térmica (TH) e comunicação (RF) (irradiação para transmissão de dados).

A Figura 4.3, resume os elementos descritos pelo *Subsystem Model Specification*.

Figura 4.3 - Elementos que descrevem o subsistema.
Fonte: Tominaga et al. (2012).

A dinâmica do modelo é representada por um conjunto de regras causais, baseadas em causas, pré-condições e consequências; formando expressões lógicas entre parâmetros de entrada e saída, que disparam eventos. Os parâmetros de entrada do comportamento podem ser telecomandos ou eventos externos de outros modelos (EX), as saídas são a telemetria, o estado de trabalho, e os valores das naturezas de potência. A Figura 4.4 apresenta os elementos que compõe o comportamento. As referencias (TOMINAGA; AMBROSIO, 2011), (TOMINAGA et al., 2013), (RAMAZOTTI; AMBROSIO, 2013) e (TORRES et al., 2013) exemplificam a especificação do comportamento dos subsistemas.

4.3 Estudo dos requisitos do simulador SIMC3

Esta seção apresenta os resultados do estudo dos requisitos do Simulador SIMC3 (AMBROSIO, 2012), a partir do qual foi possível extrair o contexto de utilização ,

remontar casos de uso e consequentemente e entender as funcionalidades da interace-usuário. Foram também identificados os requisitos que indicam formas como os elementos devem possibilitar interação.

Figura 4.4 - Comportamento dos elementos do modelo.
Fonte: Tominaga et al. (2012).

Na análise do documento de especificação de requisitos (AMBROSIO, 2012), foram identificados quatro classes de informações principais relacionadas à interação com usuário:

- Parâmetros, seus atributos, regras e nomenclaturas;
- Parâmetros em evolução no tempo de simulação;
- Agenda das tarefas pré-programadas, configurações e alterações de parâmetros ao longo da simulação;
- Registro (log) dos resultados da simulação (parâmetros e eventos ocorridos).

4.3.1 Stakeholders e contexto

O documento de requisitos (AMBROSIO, 2012) descreve que o simulador deve apoiar as atividades de engenharia e a operação dos satélites, sendo os stakeholders interessados do simulador: engenheiros de sistemas, engenheiros de operação, operadores, engenheiros especialistas e gerente da missão.

Outros stakeholders, como descrito em Tominaga et al. (2012), são a equipe de desenvolvimento do simulador: arquitetos de simulação, modeladores e equipe de programação; bem como possíveis utilizações para apresentações, treinamento de novos funcionários da equipe CBERS e uma versão simplificada para o museu do INPE (CERQUEIRA et al., 2013).

A Figura 4.5 representa os stakeholders descritos em (AMBROSIO, 2012) e e os stakeholders levantados no decorrer desta dissertação (TOMINAGA et al., 2012) (CERQUEIRA et al., 2013).

Figura 4.5 - Stakeholders interessados na simulação do CBERS.
Fonte: Adaptado de Tominaga et al. (2012) e Cerqueira et al. (2013).

O documento de requisitos (AMBROSIO, 2012) divide a utilidade do simulador SIMC3 em dois ambientes, ou modos operacionais, que são previstos para o simulador, chamando-os: análise e operação.

No ambiente de análise o simulador não se conecta com o Sistema do CCS. Os comandos são ativados pela interface-usuário ao invés de serem pelo sistema do CCS. O condutor geralmente é um engenheiro de satélite, especialista em operação, ou engenheiros correlatos responsáveis pela missão. A Figura 4.6 traz um diagrama deste ambiente.

No ambiente de operação o simulador conecta-se com o sistema do CCS, e é con-

Figura 4.6 - Ambiente de Análise para o SIMC3.
Fonte: Ambrosio (2012)

trolado via o sistema de monitoramento e controle do CCS do INPE. O condutor da simulação interage com o simulador enquanto que os operadores do CCS interagem com o sistema de controle, como se estivessem operando o satélite real. Uma interface-usuário pode ser utilizada somente para acompanhar as modificações de comportamento. A Figura 4.7 traz um diagrama deste ambiente

Figura 4.7 - Ambiente de Operação para o SIMC3.
Fonte: Ambrosio (2012)

4.3.2 Casos de uso

A partir dos requisitos (AMBROSIO, 2012), foram gerados cinco casos de usos representando as interações da interface-usuário conforme solicitados nos requisitos. Os

casos de uso ilustrados indicam cada requisito associado a ele. O software *Enterprise Architect* foi usado para edição e representações dos casos de uso.

4.3.2.1 Caso de Uso: Geral

O primeiro caso de uso, como mostrado na Figura 4.8, descreve as interações entre o usuário e o simulador em quatro classes: parâmetros (*Parameter Interaction*), evolução dos parâmetros (*Evolution Interaction*), a agenda (*Schedule Interaction*) e o registro dos eventos (*Log Interaction*), cuja descrição encontra-se nos próximos itens.

Figura 4.8 - Caso de uso geral.
Fonte: Requisitos retirados de Ambrosio (2012)

4.3.2.2 Caso de uso: Evolução dos valores de parâmetros - *Evolution Interaction*

O caso de uso *Evolution Interaction* é referente à evolução dos valores parâmetros, sendo as funções deste caso responsáveis por controlar quais parâmetros são amostrados do simulador e como será mostrada a evolução histórica deste parâmetro na rodada de simulação. A Figura 4.9 mostra os caso de uso para interação com a evolução dos valores dos parâmetros e os requisitos correspondentes.

Figura 4.9 - Caso de uso da interação com os parâmetros em evolução.
Fonte: Requisitos retirados de Ambrosio (2012)

4.3.2.3 Caso de uso: Parâmetros - *Parameter Interaction*

O caso de uso *Parameter Interaction* é referente as funções de manipulação com os parâmetros do modelo simulado, alterando valores de parâmetros ou grupo de parâmetros, restrições de acesso, inspecionando valores e alarmes em caso de alguma

inconsistência. A Figura 4.10 mostra as funções de software referentes a este caso e os requisitos correspondentes.

Figura 4.10 - Caso de uso da interação com parâmetros.
Fonte: Requisitos retirados de Ambrosio (2012)

4.3.2.4 Caso de uso: Log - *Log Interaction*

O caso de uso *Log Interaction* é referente ao registro de dados e funções de configuração, revisão de eventos e exibição em tempo de execução dos eventos, a forma como deve ser gerado o arquivo de log e a possibilidade de usar um log como fonte de dados para a interface ao invés do simulador. A Figura 4.11 mostra os casos de uso para interação com o log e os requisitos correspondentes.

Figura 4.11 - Caso de uso da interação com log.
Fonte: Requisitos retirados de Ambrosio (2012)

4.3.2.5 Caso de uso: Agenda - *Schedule Interaction*

O caso de uso *Schedule Interaction* é referente à gestão da execução da simulação e scripts, nestes estão as interação de controle de execução como inciar, parar, pausar, mudar a velocidade, o passo de simulação; e as interações dos scripts como: cenário inicial (configuração), lista de eventos e lista de falhas. Os artefatos consistem de arquivos que contêm as descrições e os tempos dos eventos desejados na rodada de

simulação. A Figura 4.12 mostra o caso de uso para interação com a agenda e os requisitos correspondentes.

Figura 4.12 - Caso de uso da interação com a agenda de execução e de eventos.
Fonte: Requisitos retirados de Ambrosio (2012)

4.4 Requisitos para o layout da interface

Dos sessenta e dois requisitos de interface-usuário do documento de requisitos (AMBROSIO, 2012), apenas dezesseis descrevem como as funções devem ser implementadas em uma interface-usuário. Cabe observar que não são especificados os elementos de interface a serem utilizados, e este foi o fator a ser explorado por neste trabalho, permitindo então, a exploração da utilização de uma interface de terceira geração. A seguir seguem os requisitos que indicam algum layout para a interface-usuário, exatamente como estão especificados em (AMBROSIO, 2012):

1 SIMRq401 O simulador deve prover botões na tela para o condutor iniciar e encerrar uma rodada de simulação.

2 SIMRq402 O simulador deve prover botões na tela para o condutor interromper temporariamente e continuar em seguida uma rodada de simulação.

3 SIMRq403 O simulador deve prover facilidades para o condutor visualizar a variação dos valores de TM, de parâmetros isolados e de grupos de parâmetros, em forma tabular.

4 SIMRq404 O simulador deve prover facilidades para o condutor visualizar a variação dos valores de TM, de Parâmetros e de grupos, em forma gráfica, através de curvas geradas pela variação dos valores de parâmetros, durante uma rodada de simulação.

5 SIMRq406 O simulador deve prover facilidades para o condutor visualizar, em qualquer momento durante uma rodada de simulação: a hora corrente, a hora de simulação, as opções de controle de execução para iniciar/encerrar interromper/continuar a simulação.

6 SIMRq407 O simulador deve prover facilidades para o condutor visualizar os estados do satélite, subsistemas e equipamentos, em uma forma sintética com um desenho que dê uma visão geral.

7 SIMRq408 O simulador deve exibir em tela, o modo de operação dos equipamentos de cada subsistema do Satélite, em tempo de execução.

8 SIMRq409 O simulador deve exibir em tela, um resumo dos dados da configuração corrente usada na rodada de simulação.

9 SIMRq410 O simulador deve prover facilidades para o condutor visualizar os telecomandos especificados para o satélite sendo simulado e comandos

especiais para mudança de modo de operação do satélite com relação ao consumo de potência.

10 SIMRq411 O simulador deve prover facilidades para o condutor visualizar a "curva do camelo"antes e durante uma rodada de simulação, conforme a data/hora estabelecida no arquivo TLE (*TBC*).

11 SIMRq412 O simulador deve prover facilidades para o condutor visualizar valores correntes de conjuntos de parâmetros simultaneamente em telas pré-definidas.

12 SIMRq414 O simulador deve prover facilidades para o condutor visualizar a ocorrência de mudança do estado de iluminação do satélite: iluminado/-sombra (eclipse), durante uma rodada de simulação.

13 SIMRq415 O simulador deve prover facilidades para o condutor visualizar todo o log de eventos, após uma determinada rodada de simulação.

14 SIMRq416 O simulador deve prover facilidades para o condutor visualizar mensagens de log de eventos, em tempo de execução.

15 SIMRq441 O simulador deve prover facilidades para o condutor definir telas de visualização associando grupos de parâmetros que devem ser visualizados juntos. O simulador deve prover facilidades para o condutor alterar parâmetros durante a rodada de simulação (alteração "*on-line*") de modo que o parâmetro e o valor escolhidos sejam alterados imediatamente. O Apêndice B apresenta detalhes sobre a alteração de parâmetros. O simulador deve prover facilidades para o condutor alterar parâmetros durante uma rodada de simulação, de modo que o parâmetro e o valor sejam escolhidos antes da rodada de simulação (falha pré-programada) e a alteração ocorra no instante programado, durante a simulação.

16 SIMRq461 O simulador deve exibir TMs e parâmetros de TMs organizadas por equipamentos e subsistemas.

4.5 Analix: análise da interface-usuário

Analix (AMBROSIO et al., 2007) (AMBROSIO et al., 2009) é uma versão do Simulador de Satélites CBERS3, constituído de duas aplicações: o próprio simulador e um visualizador dos dados que são armazenados numa rodada de simulação.

O propósito deste simulador é oferecer aos engenheiros e operadores um auxílio da análise do comportamento do satélite CBERS na fase de operação. Com este simulador é possível monitorar os valores de algumas telemetrias e parâmetros do satélite gerados durante uma simulação.

O Analix é um software de interação de segunda geração, da taxonomia de Botega e Cruvinel (2009), construído em C++/Qt.

Sua janela principal contém: (i) uma barra de menus com funcionalidades de configuração, visualização, organização de janela e ajuda; (ii) uma barra de ícones, com as funcionalidades principais do simulador; (iii) uma barra de tempo, com o tempo real e o tempo simulado; (iv) uma barra de informação, com a configuração, nome do satélite, estado de iluminação e dados de inicio da simulação; (v) e uma área de log. A Figura 4.13 ilustra a janela principal do Analix indicando os conteúdos citados.

Figura 4.13 - Janela principal do Analix.

A interface-usuário do Analix é formada por telas de monitoração, onde é possível monitorar os valores dos parâmetros do satélite e as telemetrias geradas em formato de tabelas ou gráficos.

A monitoração via tabelas consta de uma listagem dos parâmetros variáveis a serem confirgurados e monitorados de um determinado modelo do simulador; nesta opção é possível inspecionar o valor instantâneo em determinado instante da simulação. O procedimento para encontrar um valor desejado passa na seleção de monitoração via tabelas, num ícone na barra de ícones, passando por uma tela inicial de monitoração, onde o usuário escolhe qual modelo vai monitorar (Figura 4.14a) com seus diversos parâmetros, como visto na tabela de monitoração da Figura 4.14b.

Figura 4.14 - Navegação para monitoração via tabelas. (a) escolha do contexto (modelo) para visualização. (b) janela com a lista de variáveis do modelo em evolução.

A monitoração gráfica é realizada para o entendimento da evolução temporal do valor de um parâmetro, telemetria, e/ou um conjunto dos mesmos. O procedimento para acompanhar a evolução do valor de um parâmetro é semelhante à monitoração por tabelas; seleciona-se o ícone de monitoração por gráficos na barra de ícones, abrindo uma janela para escolha do modelo (Figura 4.15a), para então abrir a janela de gráficos (ver Figura 4.15b). Nela é dada uma lista de variáveis; a variável escolhida é atribuída ao gráfico. O eixo e escala do gráfico no qual os valores serão plotados devem ser escolhidos também. (No início ou no fim do gráfico, respectivamente).

Figura 4.15 - Navegação para monitoração via gráficos. (a) escolha do contexto (modelo) para visualização. (b) janela com a lista de variáveis do modelo em evolução.

No Analix é possível enviar telecomandos usando um ícone na barra de ícones. Quando escolhida esta opção, abre-se uma janela com alguns cenários operacionais do satélite. O usuário escolhe sua opção e o Analix compõe os múltiplos telecomandos necessários para executar a ação, como ilustrado na Figura 4.16. A confirmação de envio é exibida apenas na janela de Log, se esta não estiver visível o usuário não tem confirmação da execução da ação.

Figura 4.16 - Janela de escolha de telecomando a ser enviado.

5 PROPOSTA DA INTERFACE-USUÁRIO COM TÉCNICAS DE REALIDADE VIRTUAL

Este capítulo descreve o design e o projeto de software da interface-usuário de terceira geração utilizada nesta dissertação. O projeto de uma interface interativa com usuário, de terceira geração, utiliza-se de conceitos que aproximam o usuário da temática do problema, melhoram a aparência da informação apresentada na tela, facilita o aprendizado e dá coerência para o uso do software. O projeto de software inclui a descrição da metáfora, a utilização da técnica de *storyboard*s e o mapeamento das técnicas de interação (múltiplas visões e mapas, modelos 3D, RDT e painéis) em classes (de acordo com uma linguagem orientada a objeto).

5.1 Design da interface

O design de uma interface interativa conta com a sua metáfora e os sketches usados para avaliar a sua interatividade.

Metáfora

O objeto real sendo simulado é um satélite, desta forma a metáfora de apresentação da interface deve ser semelhante ao manuseio de um satélite.

Do estudo do SIMC3, foi compreendido que o alvo da modelagem do comportamento do satélite é a apresentação dos modos operacionais dos: satélites, subsistemas, equipamentos e estações; e da manipulação dos parâmetros que levam à estes modos.

Para manipular um modelo de satélite e seus comportamentos, foi buscado na ficção cientifica exemplos de possíveis interfaces, como mostrado nas Figuras 5.1a e 5.1b do mundo ficcional da Marvel (MARVEL, 2014) e na Figura 5.1c do mundo Tron (DISNEY, 2014). Estas interfaces trazem uma área de manipulação central, com mostradores, ícones, árvores e painéis auxiliares ao redor.

Uma metáfora, tirada dos exemplos ficcionais, sugere a manipulação direta com modelo 3D do objeto simulado, no caso o satélite, através da metáfora de manipulação de toques, o usuário toca num elemento de cena e recebe uma informação contextualizada. A utilização de diagramas esquemáticos, gráficos, utilização de sub-áreas ao redor para manter dados fora da área de interação e utilização de painéis corroboram com as ideias de interface dos exemplos. Nota-se que essa linha de design para interfaces já está sendo utilizada em sistemas operacionais, como a interface Metro do Windows 8.

Figura 5.1 - Exemplos de interfaces da ficção. (a) Exemplo com área central de manipulação com auxiliares ao redor. Utilização de árvore. (b) Exemplo com área de manipulação central e botoes ao redor, interação com toque no objeto. (c) Exmeplo com mapa, árvores, áreas de manipulações espalhadas, não possui uma área central.

Storyboard

O *storyboard* da interface proposta, começa com o *sketch* da Figura 5.2. No storyboard, o primeiro sketch é a apresentação do posicionamento das cenas. Na Figura 5.2, são mostrados os espaços reservados para a área principal de interação, as três áreas auxiliares, o local de registro de log, a linha do tempo de eventos e o local dos controles de execução. Este design exprime o conceito da metáfora, área central de interação com áreas de informação complementares ao redor.

Analisando os Casos de Uso descritos no Capítulo 4 e os conceitos de interação apresentados no Capítulo 3, foram definidas sete *Views* responsáveis por concentrar as atividades das quatro classes de informações: parâmetros, visualização de evolução, gestão de eventos, e log. Cada *View* foi rascunhada para mostrar quais elementos podem sofrer interação. No rascunho da Figura 5.3, as interações com:

Figura 5.2 - Sketch da tela principal, mostrando as áreas de interação (*Views*).

- Parâmetros podem ser do tipo manipulação de elementos em 3D e em árvores do tipo RDT - podendo gerar views do satélite em órbita e sua relação com o sol e estações terrenas (*Space View*), a visão dos equipamentos internos do satélite (*Sat View*) e a distribuição dos elementos em uma árvore RDT (*RDTView*);

- Visualizações da evolução dos valores dos parâmetros podem ser realizadas nas views Space, Sat e RDT, na forma de mudança de características visuais, como mudança de cor, transparência, e efeitos sonoros. Entretanto, o entendimento da evolução temporal dos eventos ainda recai sobre a utilização de gráficos (*Graph View*).

- Gestão do tempo podem ser realizadas num mostrador do tempo real e tempo de simulação (*Timeline View*).

- Log permitem ao usuário visualizar, salvar ou limpar o log utilizando uma View de interação, a *Log View*.

- Controle de interface podem ser implicitos ou explicitos em botões, para isso foi utilizada uma view com os controles (*System View*).

Figura 5.3 - Relação das *Views* com os Casos de Uso.

A Figura 5.4a mostra a *Space View* com um rascunho de planeta, um satélite orbitando e as informações de órbita; com possibilidade de ajuste de foco e seleção. Também são mostrados o sol e pontos de interesse como estações terrenas. Os elementos da *Sat View* são apresentados na Figura 5.4b, onde o satélite é dividido em subsistemas selecionáveis, que mostram o modo operacional do satélite e subsistema através da cor atual e permitem a visualização e alteração de parâmetros num painel de informações. As árvores RDTs são manipuladas na *RDT View*(ou *Param View*). No sketch da Figura 5.4c, são exibidos os nós, elementos do sistema simulado (satélite, subsistemas, equipamentos, estações terrenas, planeta), estes nós indicam o modo operacional vigente com cores e possibilitam seleção para maiores informações e edições. O acompanhamento da evolução dos eventos pode ser por inspeção instantânea nas cores dos elementos 3D, ou nos nós da RDT. Porém, para acompanhar a evolução dos valores dos parâmetros num determinado período de tempo, ainda é indicado o uso de gráficos. O *sketch* da Figura 5.4d apresenta uma proposta para a *Graph View*, indicando uma possibilidade de gráficos tridimensionais, onde um dos eixos é o tempo, e nos outros dois eixos os parâmetros para acompanhamento.[1]

Por *storyboard* também é possível fazer um *sketch* das relações, a Figura 5.5a traz

[1] A *GraphView* não foi implementada pois este trabalho não traz nenhuma inovação ao uso de gráficos.

Figura 5.4 - *Sketch* das *Views*.

um exemplo das relações entre os elementos de cena das *Views* que fazem a transição entre uma *View* para outra, e a Figura 5.5b traz um exemplo da transição do surgimento de um painel, e como ele é tratado numa narrativa do tipo *storyboard*, no caso ciclicamente.

Do trabalho intelectual de design da interface (metáfora e *storyboard*) foram retirados os conceitos de interação desejados (ver Capítulo 3) de forma que possam ser implementados, como mostrado na seção seguinte de mapeamento das técnicas de interação utilizadas em RV.[2]

[2]Obs.: Num desenvolvimento de interface, os times que montam o design da interação (chamado *UX Designer*) e que fazem a implementação da interface (chamado *UI Developer*) são diferentes (NOBLE, 2009) (SAFFER, 2008) (MELBOURNE, 2011), contudo nesta dissertação foram realizados pela mesma pessoa.

(a) Diagrama de relação entre os elementos que provocam mudança das views de interação.

(b) Sketch do comportamento de habilitação e desabilitação dos painéis.

Figura 5.5 - *Sketches* de comportamento.

5.2 Mapeamento das técnicas de interação

O desenvolvimento da interface foi baseado em formas (modelos 3D, árvore, painéis) diferentes de acessar os dados do modelo. A Figura 5.6 apresenta uma simplificação do diagrama de classes, onde o aplicativo da interface é composto por três componentes: o controle do modelo, a tela e o controle de interação. O controle do modelo

é composto pela hierarquia da estrutura de dados do modelo, e pelos métodos de comunicação com o simulador. A tela é composta pelos múltiplos métodos de acesso ao modelo das Views e painéis que o usuário pode usar para ver e interagir. O controle de interação é responsável por mapear uma interação ao que é exibido na tela e provocar uma mudança no modelo/simulador.

Figura 5.6 - Diagrama de classe com uma simplificação da relação dos objetos da interface.

Nesta seção são mostradas as técnicas de interação que compõem o protótipo da interface iSIMC3, os requisitos e as classes implementadas.

5.2.1 Multiplas visões e mapas

5.2.1.1 Aplicação da metáfora ao contexto

O design da interface-usuário indica a interação em uma área de trabalho principal e áreas secundárias de suporte. Desta forma, a área de manipulação do software, ou tela de interface, deve ser subdividida para conter cada *View*. Do *storyboard*, o conceito para as áreas de interação para esta proposta segue subdivisão inicial da Figura 5.7.

As múltiplas visões são exploradas nas *Views* da interface-usuário. A *View* principal de interação é utilizada para a exploração e navegação dos dados, as *Views* em posições auxiliares são usadas como mini mapas para manterem a coesão e contexto

da informação da *View* principal.

Figura 5.7 - Conceito do *layout* das *Views* para esta proposta

5.2.1.2 Projeto de Software

Esta interação não está ligada diretamente a um requisito, mas sim a maneira de funcionamento da metáfora/storyboard e sua relação com interfaces naturais e virtuais. Algo que é muito adotado em jogos (CERQUEIRA et al., 2013).

Em relação à implementação em software, cada *View* interativa é uma especialização de uma classe abstrata iView, que faz parte dos componentes da janela de interação do software. No oF é possível definir um *viewport* para selecionar uma certa região de interação. Então para cada iView está associado sua informação de posicionamento por uma classe iViewport. A Figura 5.8 mostra as classes iView e iViewport.

A Figura 5.9 mostra as especializações das *views*, nas views 3Ds (*Space*, *Param*, *Sat* e *Graph*) e as 2D (*Timeline*, *System* e *Log*).

O controle da *View* disponibiliza para a área principal de interação, a *View* escolhida. Para entender essa transição foi feito um pequeno estudo dos eventos de transição e os possíveis estados de interação, resultando na simplificação da Figura 5.10, onde é possível fazer o rearranjo entre duas *Views*, abrir e fechar um painel de inspeção e dependendo da opção do painel, mudar de View.

As *Views* 2d não mudam de local, cada uma tem um painel fixo com os componentes de interface-usuário.

Figura 5.8 - Classes iView e iViewport que controlam as áreas de interação.

Figura 5.9 - Diagrama de classes simplificado com as especializações da classe iView.

5.2.2 Modelos 3D

5.2.2.1 Aplicação da metáfora ao contexto

Para aproximar a metáfora de manipulação ao análogo da manipulação e navegação do elemento real, é utilizado neste estudo de caso modelos 3D dos elementos reais.

Figura 5.10 - Fluxo de mudanças da View ativa para interação.

O satélite utilizado nesta simulação é o CBERS3, para isto foi cedido pelo grupo os modelos mecânicos 3D construídos no software *SolidWorks*. O modelo foi exportado do *SolidWorks* em partes, totalizando 626(seissentos e vinte e seis) arquivos binários STL (ENNEX CORPORATION, 2014), este representa os faces do modelo 3D em triângulos, sem especificação de cor ou textura, e seus pivôs no objeto e no mundo. Sendo considerado um arquivo "limpo" para utilização em softwares de CAD. Como o software *SolidWorks* é uma solução paga, neste estudo de caso é utilizado o software 3DSMax, na sua versão para estudantes. Esta versão permite a utilização de todas as funcionalidades, desde que, sua utilização não seja comercial, o que atendeu à esta proposta.

Os modelos em STL foram importados, um a um, para o 3DSMax, possibilitando os primeiros *drafts* contidos da Figura 5.11.

O primeiro tratamento realizado sobre os modelos foi o agrupamento em subsistemas, isso foi possível de acordo com o nome dos arquivos. Este agrupamento possibilitou a separação funcional do modelo, apresentado na Figura 5.12.

No segundo tratamento foram adicionadas texturas para introduzir o realismo metálico. As Figuras 5.13, e 5.14, apresentam a renderização com texturas, do modelo

do satélite aberto e fechado, respectivamente.

Figura 5.11 - Primeiras renderizações do modelo do satélite.

Figura 5.12 - Satélite com equipamentos agrupados por subsistema.

A descrição do storyboard indica dois usos para o modelo do satélite: como uma caixa na *SpaceView* e explodido na *SatView*. No modelo caixa, para a *SpaceView* só foram utilizados os elementos externos; no modelo explodido, para a *SatView* cada subsistema foi dividido e seu pivô, em relação ao centro do satélite, foi anotado.

Figura 5.13 - Renderização do modelo aberto com texturas metálicas.

Figura 5.14 - Renderização do modelo fechado com texturas metálicas.

5.2.2.2 Projeto de Software

A utilização de modelos 3D, compreende os requisitos SIMRq407, SIMRq408, SIMRq414, fornecendo uma metáfora de navegação, semelhante ao manuseio do satélite real, mostrando se em eclipse ou iluminado, com alteração de cores claras e cores escuras; e indicar os modos/estados operacionais do satélite, subsistema ou equipamento com alteração de cores de acordo com o comportamento.

A descrição do modelo em XML (Apêndice B) contempla a informação de coloração para indicação de possíveis cenários de operação com a tag Appearance, indicando que em:

- *OFF*: desligado ficando numa tonalidade branca;
- *STANDBY*: em espera ficando numa tonalidade cinza;
- *FUNCTIONAL*: operando, ficando numa tonalidade verde;
- *FAILURE*: falha, ficando numa tonalidade vermelha;

O oF traz em seu *core* a biblioteca ASSIMP, com o *wrapper* ofxAssimpModelLoader. Esta biblioteca possibilita a utilização de diversos tipos de arquivos de modelos 3D. Para este estudo de caso o modelo foi padronizado para o formato COLLADA (KHRONOS GROUP, 1989), exportado pelo 3DSMax.

A classe Node3D foi escrita para manter os modelos 3D. Os dados dos modelos são inseridos no momento de leitura da descrição XML, e ao fim é realizado o carregamento do modelo 3D, esta função inicializa o modelo 3D na memória. Quanto maior o modelo, mais tempo demora a carregar na memória. A Figura 5.15, apresenta a estrutura do cabeçalho da classe Node3D, importante notar que só existem métodos para o tratamento da visualização dos modelos 3D.

5.2.3 RDT

5.2.3.1 Aplicação da metáfora ao contexto

Num ambiente tridimensional, segundo Jeong e Pang (1998), a melhor forma de ver uma hierarquia é a utilização de árvores cônicas.

A RDT reproduz a ação da variação de modos operacionais utilizada nos modelos 3D, contudo é possível analisar via uma hierarquia. Nos modelos 3D é possível analisar a relação dos modos e o posicionamento de maneira como se estivesse inspecionando um satélite real, no RDT essa análise está atrelada à hierarquia.

Na metáfora da árvore desta proposta, o nó raiz é a Terra, e nele estão alocados as estações terrenas e o satélite. No nó satélite estão alocados os subsistemas, que poderiam ser mais fragmentados em equipamentos e componentes. A Figura 5.16 mostra uma árvore do nó Terra até os nível dos nós equipamentos. É possível visualizar a quantidade de equipamentos de cada subsistema.

Figura 5.15 - Cabeçalho da classe Node3D

5.2.3.2 Projeto de Software

A escolha por árvores RDT atende aos requisitos SIMRq407, SIMRq408 e SIMRq461, que pedem uma visão sintetizada, hierárquica e em tempo de execução da variação dos modos operacionais.

Com a arquitetura dos elementos de simulação sendo disponibilizada numa árvore, o usuário condutor visualiza a relação de hierarquia e encontra os itens que deseja analisar de maneira mais rápida, acelerando a navegação do simulador. Numa interface normal de menus, como adotado pelo Analix, são necessárias várias telas e saber o significado do que condiz com o nome da variável. Na versão desta proposta

a navegação é realizada em apenas uma seleção e o entendimento de onde está o elemento do modelo é atrelado com sua posição na hierarquia. Acha-se o nó, faz-se inspeção instantânea da coloração do nó, para obter seu modo operacional, ou desejando acompanhar parâmetros, seleciona-o abrindo um painel para obter mais informações.

Figura 5.16 - Exemplo de RDT com equipamentos.

A hierarquia montada pelo software corresponde à hierarquia dos *links* dos nós do modelo da estrutura XML, a partir deste inter-relacionamento é criada a árvore. Outras modalidades de organização de hierarquia podem ser incluídas no futuro, neste estudo de caso só é implementada a hierarquia por linkagem de nós estrutura sistema->subsistema->equipamento.

Para montar a árvore RDT, foi necessário agregar um decorador de classe ao nó do modelo que incluísse as definições de posicionamento e exibição, como mostrado no cabeçalho da classe NodeRDT na Figura 5.17.

A RDT View é a *View* que conduz a interação na árvore RDT. No código foi criada a classe ParamView, herdada de iView. A *ParamView* contém métodos para desenhar e encontrar os nós para interação, e chama os métodos da classe de controle do modelo para recalcular a árvore em caso de alguma mudança de visualização. A

Figura 5.17 - Cabeçalho da classe NodeRDT.

Figura 5.18, mostra o cabeçalho da classe *ParamView*.

Figura 5.18 - Cabeçalho da classe ParamView.

5.2.4 Painéis

5.2.4.1 Aplicação da metáfora ao contexto

A navegação para encontrar um parâmetro ou a avaliação contextualizada de todo um sistema pode ser atingida em interface-usuários com o uso de modelos 3D e RDTs. A inspeção e alteração de parâmetros específicos podem ser realizadas com

componentes de metáforas semelhantes à utilização de equipamentos reais de medição, como um teste desta interface publicado (MORRIS; ROTHWELL, 2002), contido na Figura 5.19, onde é utilizado um transferidor como representação do movimento de multi-touch para alteração na inclinação de uma órbita.

Figura 5.19 - Teste de interação natural *multi-touch* em *SmartBoard*.
Fonte: Cerqueira et al. (2012)

Contudo, para generalizar a interação dos parâmetros da hierarquia da arquitetura de documentos XML, ao invés de criar uma interação metafórica para cada parâmetro foi escolhido a utilização de controles em painéis contextuais.

Nesta proposta, os painéis foram utilizados para representar quatro tipos de informação: conteúdo dos nós, configurações de views, log e controles do sistema. A Figura 5.20, mostra alguns exemplos destes usos, como informações de conteúdo, log e painéis de controle.

5.2.4.2 Projeto de Software

A utilização de painéis corrobora com os requisitos: SIMRq403, SIMRq410, SIMRq412, SIMRq414 e SIMRq441. Estes requisitos, de maneira geral, indicam que parâmetros precisam ser inspecionados, avaliando seus valores instantâneos e possibilitando sua mudança pela interface. E são equivalentes a inspeção de parâmetros em tabelas no simulador de referência Analix.

Figura 5.20 - Exemplo da utilização de multiplos painéis para interação de controle, de log e com conteúdo.

A implementação dos painéis utiliza o componente ofxUI do oF, este permite a abertura dos painéis e utilização de componentes de interface-usuários. Os componentes da ofxUI são orientados para utilização em interfaces *touch-screen*, e suas áreas de interação são grandes com pouca inserção direta de valor.

Cada *View* possui sua implementação de um método abstrato da classe iView, para definir o painel e recolher seus eventos, as configurações foram escritas no software. Os painéis fixos da *LogView*, do *TimelineView* e do *SystemView* também são escritas no software. Os painéis de conteúdo dos nós são dinâmicos e sensíveis ao conteúdo clicado, sendo alocados no momento de execução de acordo com a estrutura do modelo em XML. Tanto o nó quanto o parâmetro da estrutura de dados do modelo são decorados com uma instância de uma classe concreta que monta o conteúdo do painel, como visto no recorte do diagrama na Figura 5.21.

5.3 Controle da interação

A interface física de uso adotada nesta proposta permite apenas a manipulação 2D, como: mouse, telas touchscreens e smartboards. Assim, o toque do usuário se dá numa interface física plana, sem profundidade e não no objeto 3D posicionado de maneira tridimensional.

Neste estudo de caso, foram utilizados os eventos de mouse: movimento, clicks de

Figura 5.21 - Classes decoradoras dos nós e parâmetros para criação dos painéis.

início e fim; como a fonte de interação.

A interação com mouse se dá em quatro etapas: movimentação do mouse, aperto de um botão, movimento do mouse com o botão apertado e liberação do botão.

A cada movimento do mouse, o oF faz as três etapas internamente e devolve para um método as posições do mouse. Cabe a descoberta do que está próximo ao ponteiro no momento da interação a uma função que primeiro teste se o mouse está sobre um painel, ou sobre uma *View*. Estando sobre um painel o componente ofxUI fica responsável pela interação; estando sobre uma *View* é necessário descobrir o objeto mais próximo a cada interação de movimentação.

No nível de entendimento da interação para a interface-usuário, a seleção acontece no aperto inicial do mouse, fornecido pelo oF. Nesta etapa é registrada a posição do *mouse* e o objeto próximo (descoberto pelo método de movimento do *mouse*).

Na liberação do botão, a posição atual dos objetos próximos são registrados, iniciando a etapa de liberação reativa. Na liberação há a verificação se o mouse não está sobre um painel, depois verifica se o aperto do mouse registrou algum objeto, a seguir testa se as Views são as mesmas, se sim e se passou um tempo de *hold* é chamado o painel de configuração da View, se não se realiza um *swap* entre *Views*. Caso haja algum objeto próximo ao mouse no momento de início da interação e

houve um tempo de *hold*, este objeto é identificado e o painel é chamado.

As interações realizadas sobre um objeto do painel são manuseadas pelo próprio componente ofxUI.

5.3.1 Projeto de Software

O controle de interação no software ocorre via a classe iAction, que interliga a aplicação, o modelo e os elementos de tela. Para cada evento (movimentação, aperto e liberação) foi definido uma instância da classe mouseClick, vista na Figura 5.22, que registra as informações do mouse e procura na *View* onde se encontra o mouse, o elemento do modelo que está mais próximo.

Figura 5.22 - Relação das classes de controle de interação com a aplicação, tela e modelo.

A Classe iAction é responsável por analisar todos os eventos de interação, desta forma a classe da aplicação fica limpa, podendo ser substituída com facilidade, modularizando e componentizando a arquitetura da aplicação.

Os cabeçalhos das classes são mostrados na Figura 5.23, onde é possível ver os métodos de seleção, início e fim de ação na classe iAction e os atributos dos eventos de interface na classe mouseClick. Caso fosse considerado mais um modal de interação, uma nova classe de interação deveria ser agregada à iAction.

Figura 5.23 - Cabeçalhos das classes de controle de interação iAction e do modal de interação mouseClick.

5.4 Integração entre modelo, elementos de interface e comunicação

A integração dos projetos de software deste capítulo foram incrementais, contudo foram desenvolvidos estudos de casos unitários separados para aprendizado e testes. Os casos unitários que formaram o aprendizado de cada componente foram:

- Casos de Viewports: Teste de posicionamento e interação de viewports na tela. Aprendizado sobre como funciona a distribuição de áreas de interação numa janela de software;
- Caso de controle de interação: Teste de captura de posicionamento de um elemento 3D com interação 2D. Aprendizado sobre os métodos de projeção do mundo virtual para a janela de software, e captura das coordenadas do mouse;
- Caso de Modelos 3D: Leitura e coloração de modelos COLLADA. Aprendizado sobre posicionamento de modelos num espaço 3D, controle de pivôs, mudança de cor do objeto em software, utilização de canais de transparência e movimentação da câmera virtual;

- Caso de Painéis: Abertura e fechamento de painéis, interação com os controles. Aprendizado de utilização do componente de interação em 2D (ofxUI), utilização dos controles e geração de eventos dos controles;
- Caso de RDT: Teste da explosão da arvore RDT. Aprendizado do algoritmo de formação da arvore;
- Caso do Modelo: O modelo foi reaproveitado do simulador simplificado (Apêndice B), e como é uma estrutura de dados simples, não foi realizado um estudo e sim a construção direta.

A Figura 5.24 mostra o diagrama de classe completo. Os métodos e atributos são omitidos da figura por razão de clareza. Em verde claro, é representado a estrutura de dados que contém os nós dos modelos e seus parâmetros. Os nós e os parâmetros do modelo agregam as classes (em azul claro) com informações de interação (modelos 3D, posicionamento e painéis). Em azul são representadas as classes que implementam as *views*, que agregam o seu posicionamento via classe Viewport e são agregadas a classe iScreen, que controla o que é exibido. As classes iAction e a classe do modal de interação com mouse, em laranja, fazem a ponte de interface entre o modelo e as *Views*. Em roxo, é representado a classe que faz a conexão com o simulador [3].

[3] A explicação dessa conexão não faz parte do escopo deste trabalho, e por isso é omitida.

Figura 5.24 - Diagrama de classes completo da interface iSIMC3.

5.5 Fluxo de execução

Visando ilustrar a interface-usuário proposta, esta seção apresenta o fluxo de navegação através das facilidades disponibilizadas.

Um simulador simplificado (descrito em detalhes no Apêndice B) foi implementado para permitir a avaliação da interface-usuário. Desta forma, a variação dos valores dos parâmetros e as respostas às opções requisitadas na interface-usuário são similares aquelas fornecidas por um simulador operacional.

Como infraestrutura de simulação foi utilizado um notebook com processador i5, placa de vídeo *Nvidia GeForce GT*630, 6Gb de RAM DD3 com Windows 8.1. , Visando ilustrar a execução da simulação para demonstração da interface foi proposto um cenário no qual um modelo do subsistema de energia, um modelo representativo de todo o satélite, um modelo de ambiente, e a interface possui os controles para a evolução do simulador.

Ao iniciar a execução do Simulador simplificado, contendo a interface-usuário proposta, a primeira tela exibida é aquela mostrada na Figura 5.25, a qual mostra a SpaceView na área de interação principal. Antes da conexão, a interface pode ser explorada para aprendizado ou para demonstração (por exemplo: museu).

Figura 5.25 - Tela inicial da interface proposta.

O simulador simplificado tem uma configuração padrão, carregada a partir dos arquivos XML, sendo esta a condição suficiente para explorar a interface.

Tomando-se como referência o Analix, ao iniciar a execução de uma simulação, dois trajetos são possíveis: (i) selecionar um modelo simulado e visualisar seus parâmetro,e visualizar seu valor corrente num dado instante e (ii) selecionar um modelo simulado e visualisar seus parâmetros em um gráfico.

No interace, o usuário já possui a visualização das alterações macros de cada nó/modelo, já que as cores representam o modo de operação do satélite e dos subsistemas e equipamentos em cada instante.

Para visualização de um valor de parâmetro instantâneo, são dadas duas opções de navegação: utilizar os modelos 3D ou utilizar a árvore RDT, caso a opção desejada não esteja na área de interação, a mesma deve ser arrastada para a área principal.

A Figura 5.26 mostra os modelos 3D na área, primeiro sem nada selecionado e consequentemente as partes do modelo em aramado (*wireframe*).

Figura 5.26 - SatView sem nenhum subsistema selecionado.

A movimentação do mouse sobre os modelos 3D faz com que eles sejam preenchidos, indicando para o usuário a possibilidade de execução de uma ação, como mostrado na Figura 5.27.

Figura 5.27 - SatView com subsistema selecionado.

Uma vez escolhido o objeto, o condutor apertando e segurando com o mouse seleciona uma ação. A ação é a abertura de um painel contextualizado com as condições instantâneas do objeto selecionado, como mostrado na Figura 5.28.

Figura 5.28 - Painel contextualizado aberto com as condições instantâneas.

O mesmo processo é realizado na busca na árvore, como mostrado na Figura 5.29,

onde nenhum elemento está selecionado.

Figura 5.29 - Tela com a árvore RDT aberta.

Quando o mouse está sobre um nó da árvore este é indicado por uma linha amarela do mouse até o ponto desejado, como visto na Figura 5.30.

Figura 5.30 - Tela com a árvore RDT aberta e indicando uma nó selecionado.

O processo para abrir um painel é semelhante ao utilizado na vista 3D, com o objeto selecionado, segura e libera um botão do mouse. O resultado é a abertura do painel com os parâmetros do respectivo nó, como visto na Figura 5.31.

Figura 5.31 - Tela com a árvore RDT e com o painel aberto.

Para abrir a inspeção de parâmetro, caso a *View* desejada esteja na área de interação, basta uma operação, de aperto sobre um elemento, que representa de maneira significativa a metáfora do objeto simulado real. Esta representação, segundo a bibliografia (DENNIS et al., 2005) (NOBLE, 2009) (NIELSEN, 1993) (NORMAN, 1993) (KIM, 2005), permite que uma interface de software seja de fácil aprendizado, fácil memorização e de maneira eficaz e eficiente.

Para acompanhar a evolução de parâmetros, além da forma mental, de ir memorizando os valores instantâneos e mentalmente montar a evolução; a forma proposta pelo Analix é a utilização de gráficos. Esta forma continua sendo a melhor para este tipo de análise, não foi encontrada literatura que indicasse uma forma de visualização melhor.

5.6 Extensão e troca de artefatos

Neste trabalho, foram desenvolvidos dois produtos de software: a interface-usuário e um simulador simplificado. Ambos os produtos gerados podem ser reutilizados e estendidos sem necessidade de recompilação.

- Como tomamos como base para estudo um satélite real e complexo, o CBRES3&4, foi feita uma simplificação do modelo do satélite coim poucos documentos de modelos. Uma das atividades seria completar a arquitetura de documentos com todos os modelos necessários para simular com maior ou menor fidelidade.

- A modelagem atual do satélite do estudo de caso foi realizada em documentos que especificam o comportamento de subsistemas. Apesar de não entrar no nível de descrição de equipamentos a arquitetura projetada permite a granularização ou concentração da hierarquia de documentos em XML. Esta decisão afeta o simulador simplificado e a interface, pois a estrutura de modelos é semelhante, na hora de extensão deve-se escolher a granularidade de acordo com o necessário.

- A fidelidade do modelo pode ser ajustada de acordo com a quantidade de parâmetros e expressões que descrevem, porém, deve-se pensar no fato de que muitas expressões aumentam a quantidade de processamento, reduzindo o tempo de cada ciclo de avaliação da rodada de simulação.

- Cada nó possui um campo que especifica os modelos 3D que devem ser carregados, e os atributos geométricos que os modificam para encaixar-se no pivot do modelo 3D do satélite. Estes modelos 3D podem ser substituídos por outros com maiores ou menores detalhes de acordo com o uso e granularidade desejada.

- Outro ponto de alteração é o arquivo TLE localizado na estrutura de arquivos do simulador, este pode ser alterado por um arquivo mais novo.

Uma alteração completa do satélite acarreta na substituição de todos os artefatos: modelos de comportamento na estrutura de documentos XML, arquivo TLE de descrição da orbita e os modelos 3D.

Uma reutilização mais complexa envolveria a recompilação, como o caso de uma distribuição para outra arquitetura (*Linux/Mac/iOS/Android*), o reuso dos componentes individuais ou para melhorias/expansões de funcionalidade, mas nesta seção não são abordados. Estes casos tornam-se alvo de trabalhos futuros.

5.6.1 Utilização de software livre e desacoplamento do código, da estrutura e do conteúdo

Neste trabalho foram utilizadas ferramentas livres, as quais só podem ser usadas para o desenvolvimento de aplicações não comerciais, como o *Visual Studio C++ Express 2012* para programação, *3D Studio Max 2014* para trabalhar com os modelos 3D e a ferramenta totalmente livre Notepad++ como editor dos arquivos XML.

Para modelagem e simulação do simulador simplificado, foram utilizados: o *openModelica*, que é um ambiente de simulação multi-domínio, e nesta dissertação foi usado para modelagem do comportamento de um circuito simplificado, análogo ao de uma bateria; e o *SciLab*, que é um ambiente de simulação baseado em programação, foi usado para modelar e simular as funções de carga e descarga da bateria. Foi utilizado dois softwares de modelagem, pois no openModelica é possível "montar" o circuito e simular, e salvar as curvas para analise do comportamento do circuito. As curvas geradas foram usadas para validar as funções criadas no SciLab. As funções foram usadas no modelo do simulador simplificado descrito no Apêndice A.

O framework escolhido para desenvolvimento foi o openFrameworks, ele é open-source sob licença MIT, permitindo uso comercial ou não, com distribuição de código ou não. Contudo algumas bibliotecas utilizadas como a SpaceTrack restringem que o código deva ser distribuído e aberto. Este fator pode levar os desenvolvedores a a escolher soluções que deixarão o código fechado, contudo, no escopo desta dissertação, como tentativa de evitar esta característica, o software foi separado do conteúdo. Desta forma, o software poderá ter o código fonte distribuído, reutilizado e melhorado além desta dissertação e ainda assim mantendo o conteúdo sigiloso seguro.

Neste trabalho foi elaborado um esquema para representar o desacoplamento de ideias em camadas. Este desacoplamento permite dividir o trabalho para grupos especializados específicos. Foram definidas três camadas: infra-estrutura, estrutura e conteúdo:

- Camada de infra-estrutura: contém o código das aplicações, partes que não são mudadas, e que foram a base das aplicações, no caso da interface e do simulador simplificado. Nesta camada, é executado trabalho de programação para interligar a estrutura, posicionar e interpretar conteúdos. Nesta camada, o código pode ser distribuído e open-source pois não é sigiloso.

- Camada de estrutura: contém os arquivos que montam a hierarquia dos objetos da simulação (nós da hierarquia e seus parâmetros). Nesta camada, é executado trabalho de identificação e modelagem da estrutura a ser exibida/simulada, levantamento dos parâmetros. Nesta camada, a hierarquia de XML pode ser uma representação fiel do sistema, podendo ser sigilosa, porém uma representação simples pode não ser sigilosa.

- Camada de conteúdo: contém os conteúdos indicados pela estrutura para a interface (modelos 3D) e para a simulação (comportamentos e TLE). Nesta camada, é executado o trabalho de modelagem 3D e modelagem do comportamento (relações entre os diversos parâmetros). Nesta camada, os modelos podem ser uma representação fiel, podendo ser sigilosos, porém uma versão simples possa não ser.

A Figura 5.32 ilustra as camadas de desacoplamento do código, da estrutura e do conteúdo. A extensão e troca dos artefatos pode ser realizada nas camadas de estrutura e conteúdo, substituindo os arquivos XML, os modelos 3D, modelos de comportamento e o TLE.

Figura 5.32 - Desacoplamento do código, da estrutura e do conteúdo.

6 AVALIAÇÕES

Este capítulo apresenta a avaliação da interface-usuário desta dissertação. A avaliação foi baseada em um questionário aplicado a dois grupos distintos e realizada em duas etapas. Na primeira etapa, o grupo de avaliadores foi composto por alunos do curso de Engenharia e Tecnologia Espaciais, com perfil jovem. Na segunda etapa, o grupo de avaliadores foi composto por representates dos engenheiro, desenvolvedores, modeladores e operadores envolvidas nos projetos satélites do INPE, portanto pessoas com perfil mais velho, com grande conhecimento técnico em simuladores de satélites - o grau de representatividade desta amostra é grande, pois são pessoas experientes e com alto envolvimento nos projetos espaciais. Os resultados indicaram que para o perfil mais novo, onde este tipo de interface é mais comum, a experiência de uso é mais natural, fazendo mais sugestões. Para o perfil mais velho, a experiência numa interface 3D trouxe maiores facilidades de observação de itens que estavam acostumados na simulação. Dos diferentes perfis profissionais, cada um focou em uma característica da interface, e diante o atendimento das diversas sugestões dadas pelos entrevistados, resultou-se que a utilização das técnicas de interação desta proposta são desejáveis.

6.1 Questionários para avaliação da interface

A avaliação foi realizada com base em questionário aplicado a pessoas voluntárias que manipularam a interface e responderam as questões.

A avaliação foi realizada em duas etapas, cada uma com um questionário:

- Primeira etapa: visa avaliação preliminar, realizada no meio do desenvolvimento por alunos do curso ETE. O questionário dessa etapa abrangiu aspéctos simplificados.
- Segunda etapa: visa avaliação dos elementos final, realizada no final do desenvolvimento por pessoas experientes e alto envolvimento no desenvolvimento e utilização de simuladores operacionais. O questionário dessa etapa abrangeu mais pontos de análise e possibilitou respostas abertas dos avaliadores.

6.1.1 Questionário da primeira etapa

O questionário da primeira etapa foi elaborado levando em conta os fatores de avaliação estabelecidos por Kirner e Kirner (2013), próprios para avaliação de interação

com interfaces. A Tabela 6.1 mostra as perguntas elaboradas, pelo autor desta dissertação, para uso desta dissertação e sua classificação. Este questionário foi aplicado utilizando a interface de formulários do *Google Docs*[1]. Estas questões visaram avaliar preliminarmente: a interface, os aspéctos visuais, a motivação de uso, o comportamento e acessibilidade.

Tabela 6.1 - Questionário da primeira etapa.

Dimensões	**Pergunta**
Interação	(1) Quão fácil é encontrar um equipamento?
	(2) A representação visual é compatível?
	(3) Como foi o tempo de resposta da interação?
Interface	(4) A informação está posicionada adequadamente?
	(5) A interface é atrativa?
	(6) Foi fácil de aprender os elementos?
Aspectos Visuais	(7) As cores estão em harmonia?
	(8) Os textos estão legíveis?
	(9) Qual a qualidade dos objetos 3D?
Motivação	(10) É difícil de manter a atenção?
	(11) A associação de conteúdo está correta?
	(12) Qual o grau de satisfação?
Comportamento	(13) Como são os feedbacks dos objetos próximos?
	(14) Quão adaptável para diferentes usuários?
Acesso	(15) É simples de carregar?
	(16) é simples acessar as views?
	(17) É fácil de encontrar parâmetros?

6.1.2 Questionário da segunda etapa

O questionário da segunda etapa, elaborado pelo autor desta dissertação, foi baseado em tópicos derivadas do artigo (MARTINS et al., 2013) que define dimensões e critérios utilizados para avaliar a usabilidade em interfaces de realidade aumentada, mas possibilita a extensão para avaliação de interfaces que utilizam metáforas naturais de uso. Além das questões diretas, foi deixado um espaço para respostas livres, sugestões, proposições e justificativas complementares às questões diretas.

Cada uma das dimensões mede características de facilidade do entendimento e uso

[1]Disponível em: drive.google.com

da interface. A Tabela 6.2 apresenta a dimensão e a pergunta elaborada no contexto desta dissertação.

Tabela 6.2 - Questionário da segunda etapa.

Dimensões	**Pergunta**
Interação com Sistema	**Perguntas em relação ao manuseio da interface e o hardware de interação, medindo a navegabilidade via mouse e se este é de fácil aprendizado.** (1) É fácil a manipulação dos elementos da tela? (2) É fácil entender a manipulação do mouse sobre uma área de interação? (3) É fácil entender a manipulação do mouse para transitar entre área de interação? (4) A interface transmite uma percepção de profundidade? (5) É possível entender que o mouse "toca"elementos 3D, com profundidade, mesmo que o mouse trabalhe em 2D? (6) A quantidade de opções de interação do mouse é suficiente para acompanhar a simulação?
Interface de aplicação	**Perguntas em relação à apresentação da informação, se esta é atrativa e fácil de usar e se os feedbacks são consistentes.** (7) Quão atrativa é a interface? (8) Qual a qualidade dos modelos 3D utilizados? (9) Quão fácil é manipular as áreas de interação na segunda utilização? (10) É necessário um manual para entender o funcionamento deste simulador com interface 3D? (11) O modelo 3D representa corretamente o objeto real? (12) A árvore de parâmetros reproduz a hierarquia corretamente? (13) As cores utilizadas são consistentes para representar o comportamento do satélite e dos subsistemas? (14) É fácil identificar o modo operacional funcional? (15) É fácil identificar o modo operacional de falha? (16) É fácil identificar o modo operacional desligado?

Continua da próxima página.

Tabela 6.2 – *Continuado da página anterior.*

Dimensões	Pergunta
Sensorial	**Perguntas relacionadas a capacidade de participação do usuário, instintividade e clareza, verificando a cognição experimental.** (17) Ao clicar sobre um objeto a resposta é conforme a esperada? (18) Ao arrastar um objeto a resposta é conforme a esperada? (19) É clara a possibilidade de modificar parâmetros de simulação através do modelo 3D do satélite e dos subsistemas? (20) É clara a possibilidade de modificar parâmetros de simulação através da árvore de parâmetros? (21) É intuitiva a modificação de parâmetros na tela? (22) É possível ver com clareza os valores dos parâmetros na tela?
Associação espacial	**Perguntas relacionadas a navegação da tela, da compatibilidade do conhecimento e do mapeamento das informações.** (23) É fácil encontrar um parâmetro? (24) É fácil alterar um modo de operação? (25) É fácil alterar o valor de um parâmetro? (26) É fácil perceber que o satélite está em orbita?
Representação	**Perguntas relacionadas a aparência e disposição dos elementos da tela, aspectos artísticos e de realismo, da sobrecarga de informação da tela, da relevância e fidelidade das representações.** (27) As disposições das áreas de interação estão boas? (28) O posicionamento dos equipamentos exprime o realismo da situação? (29) É interessante ver o comportamento expresso através de mudança de cor nos modelos 3D dos equipamentos do satélite? (30) O posicionamento dos objetos da árvore exprime o realismo da relação entre os objetos reais?

Continua da próxima página.

Tabela 6.2 – *Continuado da página anterior.*

Dimensões	**Pergunta**
	(31) É interessante ver o comportamento expresso através de mudança de cor na árvore? (33) A quantidade de áreas de interação na tela consegue representar o que está sendo simulado? (área de interação do espaço, área de interação do satélite e área de interação da árvore) (34) É mais fácil encontrar o caminho do parâmetro por onde? (Árvore ou Modelo 3D).
Motivação e esforço	**Perguntas relacionadas a percepção da importância do tema, da satisfação do usuário, e da capacidade de promover a utilização do tipo de interface.** (35) O uso deste tipo de interface facilita acompanhar uma rodada de simulação de um simulador operacional? (36) A utilização de elementos 3D possibilita o entendimento instantâneo do estado do satélite e seus subsistemas? (37) A utilização da árvore possibilita o entendimento instantâneo do estado do satélite e seus subsistemas? (38) Uma interface com interação 3D motiva o uso de um simulador operacional? (39) Este tipo de interface poderá substitui a interface tradicional de um simulador operacional, no futuro?

Para cada questão, de ambos os questionários, os entrevistados tiveram cinco respostas,de acordo com a escala Likert (JAMIESON, 2004): 5 - Excelente, 4 - Muito Bom, 3 - Satisfatório, 2 - Ruim, 1 - Pobre. Com exceção da pergunta trinta e quatro que pede ao avaliador para escolher entre os dois métodos de navegação de dados (árvore ou modelo 3D).

6.2 Resultados da primeira etapa da avaliação

A primeira avaliação foi realizada aproximadamente no meio do período de desenvolvimento da interface, para validar a metáfora de interação e a estética das técnicas e da distribuição dos elementos de interfaces usados.

O grupo de teste foi uma amostra de conveniência de vinte pessoas, estudantes de pós-graduação do INPE, com idade aproximada de trinta anos. Para esta avaliação a interface foi oferecida sem nenhum conteúdo exato, apenas para avaliação, os modelos 3D eram caixas que representavam posições aleatórias. Para cada entrevistado foi realizado uma breve demonstração e explicação e possibilidade de manuseio. O questionário foi realizado com auxilio da ferramenta de formulário do GoogleDrive[2] e para cada entrevistado foi utilizado não mais que quinze minutos.

O resultado da análise das respostas é mostrado no gráfico da Figura 6.1, contendo o resultado de satisfação de pobre a excelente, baseada numa escala Likert, dos fatores perguntados.

Figura 6.1 - Resultados da avaliação preliminar.

A conclusão da pesquisa da primeira etapa foi que, para esta faixa etaria, a interface apresentada facilitaria encontrar equipamentos, o sistema responde num tempo adequando, as poucas informações estavam posicionadas de maneira correta. A interface apresentou cores em harmonia, com textos legíveis e objetos 3D com qualidade. A forma como foi implementada o feedback de objetos próximos indicou facilidade de manuseio. Os entrevistados consideraram a interface atrativa, porém difícil de manter a atenção com a quantidade de elementos, e estes consideraram que atenderia a tipos diferentes de usuários (especialistas e não-especialistas).

[2]Disponível em https://drive.google.com/

6.3 Resultados da segunda etapa da avaliação

O segundo teste foi realizado após a conclusão da interface proposta. O objetivo da avaliação foi capturar a sensação da experiência de usuários acostumados a trabalhar com simulação em softwares baseados em WIMP.

O grupo de avaliadores foi uma amostra de conveniência formada por profissionais, engenheiros, analistas de software e operadores de satélites. Este grupo compreendeu sete pessoas voluntárias, funcionários do INPE. O perfil foi definido com relação a atividade (ou papel que a pessoa já assumiu no desenvolvimento de um simulador de satélite) e a idade. Para cada entrevistado foi apresentado a interface-usuário proposta, mostrando as funcionalidades e explicando o funcionamento do mouse e elementos de cena da interface, seguido pelo manuseio da interface pelo entrevistado. O questionário foi entregue impresso, a ser preenchido com a utilização, cada entrevista levou em média duas horas. Quatro perfis diferentes foram usados nesta avaliação:

- perfil modelador: entrevistado A e B,
- perfil desenvolvedor: entrevistado C e D,
- perfil operador: entrevistado E e F,
- perfil engenheiro: entrevistado G.

A idade dos entrevistados variou de uma primeira faixa de aproximandamente trinta anos e de uma segunda faixa de aproximadamente cinquenta anos.

As médias de cada perfil entrevistado são mostradas no gráfico da Figura 6.2.

Cada perfil apresentou um comportamento de respostas diferente, que indica o seu gosto diante ao que esperar de uma interface deste tipo.

O perfil **modelador** apresentou maior interesse em visualizar o resultado do modelo apresentado na tela. Como a interface foi feita alinhada aos dados que o modelo fornece, a avaliação deste grupo apresentou bons resultados devido ao conhecimento da forma de pensamento do modelo. A Figura 6.3 apresenta os resultados para o grupo de modeladores.

O perfil **desenvolvedor** apresentou maior interesse nas formas de interação direta com os parâmetros, isto é, no *layout* de disposição dos dados. Houve diversas su-

Figura 6.2 - Médias das respostas de cada perfil.

Figura 6.3 - Resultados para o grupo de modeladores.

gestões de navegação dos parâmetros e melhorias nas formas de agrupamento dos dados nos painéis. Este grupo não apresentou conhecimento do realismo visual do satélite. A Figura 6.4 apresenta os resultados para o grupo de desenvolvedores.

O grupo do perfil **operador** apresentou maior interesse pela utilização da árvores de dados, pois considerou esta, a maior contribuição da nova interface. Como mais uma forma de interação, foi sugerida a diagramação de blocos funcionais 2D (chaves), para demonstrar a correlação dos parâmetros internamente aos equipamentos, ao invés da utilização dos painéis. Este grupo não apresentou conhecimento do realismo visual do satélite. A Figura 6.5 apresenta os resultados para o grupo de operadores.

Figura 6.4 - Resultados para o grupo de desenvolvedores.

Figura 6.5 - Resultados para o grupo de operadores.

O grupo do perfil **engenheiro** apresentou maior interesse pelos modelos 3D, gostou da diagramação em árvore, e sugeriu a inclusão de variações de layout que seriam úteis. Foi pedida a exploração de gráficos de forma mais elaborada, pois este é o principal uso no ponto de vista do grupo de engenheiros, e que não foi explorado neste trabalho. A Figura 6.6 apresenta os resultados para o grupo de engenheiros.

A separação do grupo amostrado por idade (aproximadamente trinta e cinquenta), permitiu notar uma característica interessante: o grupo de menor idade tem maiores exigencias para uma interface como a proposta, enquanto o grupo de maior idade apresentou poucas exigências, talvez devido pouca familiaridade com o mundo dos

Figura 6.6 - Resultados para o grupo de engenheiros.

jogos e/ou softwares interativos. A Figura 6.7 apresenta os resultados para a amostra dividida por idade.

Figura 6.7 - Resultados para idades de aproximadamente trinta e cinquenta anos.

Computou-se a média de cada dimensão perguntada. O gráfico da Figura 6.8 ilustra as médias das dimensões do questionário de usabilidade aplicado.

Com o resultado da segunda etapa de avaliação e de acordo com a média de cada dimensão perguntada entre os vários perfis e idades observamos que:

1 de maneira geral as interações do mouse com a interface foram claras, tanto na navegabilidade e quanto na manipulação,

Figura 6.8 - Médias das dimensões do questionário de usabilidade aplicado.

2 a forma como foram implementados os painéis de contexto não foram satisfatórias, e precisam ser reformuladas (visto respostas da dimensão sensorial),

3 a associação e representação da metáfora de utilização com o objeto real talvez não seja determinante para o grupo amostrado já que a maioria dos entrevistados não tem conhecimento da localização dos equipamentos no satélite, com esceção do grupo de engenheiros.

4 das observações de motivação, todos os grupos apresentaram expectativas para o uso de interfaces 3D, minimalistas, com interação baseada em contexto e com manipulação com mão, ou outros artifícios, e estes serão uma realidade no futuro da simulação operacional, como apresentada nesta dissertação,

5 da divisão de grupos por idade, observou-se maior criteriosidade da manipulação da interface pelo perfil de aproximadamente trinta, e maior entusiasmo em relação as possibilidades futuras pelo perfil de aproximadamente cinquenta.

6 das dimensões analisadas, a dimensão **sensorial** foi a que obteve menor resultado, e esta teve por objetivo inspecionar a instintividade e clareza, mostrando que a mudança de paradigma não é natural para os avaliadores, o que reforça a necessidade de um pequeno manual e treinamento. Contudo os avaliadores afirmaram que num segundo uso, essas formas de interação se tornam obvias, e que interface-usuários com técnicas de naturais de

interação terão espaço ampliado, como mostra o resultado da dimensão **motivação**.

Das técnicas avaliadas (árvores, modelos 3d, multiplas visões, utilização de audio e painéis) é possível elencar das respostas livres quais devem ser aplicadas inicialmente, devido à predileção dos avaliadores.

A primeira sugestão de aplicação é a utilização das árvores (vide pergunta 34 do segundo questionário). Segundo todos os avaliadores, esta forma de representação da hierarquia dos componentes é de fácil utilização, manipulação e verificação dos estados. A segunda sugestão é a utilização dos painéis contextualizados, porém, de maneira melhorada. A terceira sugestão é a utilização de multiplas visões, possibilitando pontos de vistas diferentes. A quarta sugestão é a utilização de gráficos, mesmo que não implementada nesta proposta foi a técnica mais comentada, e de vital importancia e plausível de utilização das mesmas técnicas de manipulação das vistas 3D. A quinta sugestão é a utilização de visões 3D utilizando modelos 3D, mesmo produzindo facilidades de visualização contextualizada com a metáfora do problema (satélite em orbita e vista do posicionamento real dos equipamentos) esta produziu menor efeito nos avaliadores. Seguem algumas das observações:

- Árvores: Facilidades de alteração de critério de layout, nós selecionáveis e com controle de expansão e salvamento de vistas específicas.
- Modelos 3D: As vistas 3D (SpaceView e SatView): Facilidades de visualização dos parâmetros de orbita, mostrando a elipse, e os pontos que caracterizam a orbita.
- Múltiplas Visões: Permitir alteração em tempo de execução do posicionamento das visões, facilidades de habilitar e desabilitar as visões.
- Painéis: Facilidades de agregação dos parâmetros em caixas simples, somente com o nome e valor, e ir expandindo de acordo com a necessidade de visualização ou edição. Prover métodos de organização dos parâmetros por nome, ou outra caracteristicas e um forma de buscá-los. Facilidades de fixar um painél na tela e deslocá-lo da posição padrão.

A Tabela 6.3 resume as sugestões de sequência de utilização das técnicas apresentadas nesta proposta, com a métrica do custo computacional de processamento e da aceitação dos avaliadores.

Tabela 6.3 - Resumo e sugestão da ordem de utilização

Ordem de uso	**Técnica**	**Processamento**	**Aceitação**
1	Árvores	Muito-baixo	Alto
2	Painéis	Baixo	Médio
3	Múltiplas visões	Médio	Médio
4	Gráficos interativos	Baixo	Alto
5	Vistas 3D	Alto	Baixo

7 CONCLUSÕES

Este trabalhou apresentou a proposta de uma interface-usuário para representar dados de um simulador operacional de satélite em um ambiente de realidade virtual, utilizando metáforas de uso diferentes: múltiplas visões simultâneas, modelos 3D, RDTs e painéis de interação. As metáforas foram realizadas com a utilização de softwares livres e do *framework* de desenvolvimento de interfaces interativas openFrameworks e avaliado através de testes com usuários.

O desenvolvimento de simuladores, de maneira geral, e em particular os simuladores operacionais, visa principalmente provar um modelo, de modo que as questões relativas a formas de interação do condutor com o software são negligenciadas. Porque a interface de usuário é deixada de lado?

Como complemento à uma abordagem de união de computação gráfica com simulação computacional foi adicionado ao desenvolvimento de um simulador, aspectos de interação utilizando RV. A simulação passa, então a ser visual-interativa, a qual corresponde a extensão do conceito de "simulação operacional", que significa que o usuário é capaz de operar o modelo simulado através de uma interface interativa, e este trabalho mostrou a viabilidade de construção de um simulador operacional de satélites utilizando estas técnicas.

As interfaces-usuário dos softwares simuladores são, na sua maioria, baseadas em WIMP. A avaliação da interface realizada tanto na etapa preliminar quanto na etapa final com a apresentação do protótipo com estrutura e conteúdo definidos, mesmo parecendo obvia, mostra que a utilização de interações com metáforas de uso similares à utilização do objeto simulado potencializa a utilização do simulador operacional. A utilização das técnicas de realidade virtual facilitam a cognição no momento da interpretação dos dados simulados, atendendo a especialistas e não-especialistas na hora da visualização do comportamento e de dados técnicos. O que leva a pergunta do porque esses sistemas não são projetados desta maneira?

Uma possível resposta é a falta de normatização, disponibilização de um *framework* de interação veloz e portável que possa ser atrelado com os padrões de simulação operacional de satélites. Pesquisas neste sentido estão evoluindo rapidamente.

Este capítulo apresenta as (i) contribuições conceituais, (ii) contribuições para os projetos de desenvolvimento e (iii) conclui indicando alguns trabalhos futuros.

7.1 Contribuições em conceitos

Esta seção discute as contribuições resultantes da realização da pesquisa desta dissertação.

7.1.1 Visualização Interativa em Simulação

Na seção 2.2 foi apresentada uma maneira de integrar a simulação com a computação gráfica. Segundo Lee (1999), a simulação visual consiste na adição de um fluxo contínuo de visualização aos dados de simulação, contudo Lee (1999) não contempla interação. Como resultado desta dissertação, propõe-se adicionar no diagrama da Figura 2.2, um bloco de interação para a manipulação da evolução do modelo, tornando a simulação visual-interativa. A Figura 7.1, mostra o novo bloco de interação e mostra que a interação pode: (i) mudar a forma como uma simulação é vista na fase de renderização, (ii) ser um fator facilitador nas tomadas de decisão da evolução do comportamento do modelo e (iii) controlar o gestor de tempo.

Figura 7.1 - Inserção da interação na simulação visual.
Fonte: Adaptado de Lee (1999)

A simulação visual-interativa mapeia a camada de interface-usuário com os dispositivos e drivers, para ter acesso às ações e sensações humanas do usuário (CARDOSO et al., 2012) e melhorar a usabilidade de simuladores espaciais.

7.1.2 Adição de elementos na taxonomia de interação

Wenzel et al. (2003) propôs duas taxonomias das técnicas de visualização para apresentar dados de simulação: uma do ponto de vista dos elementos gráficos e outra relativa à informação da simulação. Quando se acrescenta o conceito de interação à simulação, consequentemente, a taxonomia de Wenzel et al. (2003) deve ser ajustada para incluir o novo conceito. A Tabela 7.1 mostra os critérios já definidos, sobre simulação (laranja), visualização(azul), interação (verde) e os novos elementos, em branco, correspondem à natureza de interação, o feedback da interação, a forma de manipulação, e a interface de interação.

Tabela 7.1 - Adaptação com novos elementos de interação à taxonomia proposta por Wenzel et al. (2003).

Critério	Especificações						
Dimensão	1D	2D	2 ½ D	3D			
Representação	Simbólica/ Caracter	Simbólica/ Abstrato	Icônica/ Estilizada	Icônica/ Realista	Foto-realista		
Formato da exibição	Texto	Tabela	Gráfico	Diagrama	Fotográfico	Modelos 3D estáticos	Modelos 3D animados
Escala da visualização	Nenhuma	Linear	Logaritmica	Exponencial	Categorizada		
Escala de tempo	Nenhuma	Linear	Logaritmica	Exponencial	Categorizada		
Projeção geométrica	Nenhuma	Ortogonal	Oblíqua	Perspectiva			
Sistema de coordenadas da visualização	Da tela	Local	Global				
Dimensão temporal do modelo	Nenhuma	Discreto	Contínuo				
Velocidade de apresentação	Estático	Sem relação	Lento	Tempo Real	Acelerado		
Local de Interação	Nenhum	Na navegação	Nos modelos gráficos	Nos modelos de simulação	Nos parâmetros dos modelos		
Natureza de interação	Presencial	Geo-localizada	Característica	Temporal			
Feedback de interação	Visual	Sonoro	Táctil				
Manipulação	Direta	Comandos	Menus	Dependente de estado			
Interface de interação	Realidade Virtual	Realidade Aumentada	Realidade Cruzada	Artefato Tangível	Tela		

Wenzel et al. (2003) tinha definido realidade virtual e realidade aumentada como

forma de exibição, e utilizando a definição de interface de interação os dois tipos de interfaces foram reposicionados. Esses novos elementos expandem a taxonomia, de ”elementos gráficos” para ”elementos de interação”.

7.2 Contribuições para o INPE

O desenvolvimento desta dissertação incluiu um estudo do simulador operacional em desenvolvimento no INPE (Caítulo 4), que usa um satélite real de grandes dimensões e de conhecimento do INPE, para realizar as avaliações com os usuários de forma mais fidedigna. Como resultado deste estudo foi possível apontar algumas contribuições técnicas que poderão ser adotadas em projetos do INPE:

- Do estudo dos requisitos gerais do SIMC3, foram identifcados e documentados casos de uso, os quais não faziam parte dos documentos do SIMC3 (Capítulo 4).
- Da revisão detalhada dos requisitos de interface-usuário, foi observado que os requisitos de interface descritos não especificam a forma como a interface com usuário deve ser realizada, possiblitando a implementação, desde linha de comando até interações naturais.
- Proposta de estruturação do software de um simulador de satélite usando (i) hierarquia de nós com parâmetros, descritos em XML e (ii) um analisador matemático para interpretas as expressões do modelo nos documentos XML, externos ao corpo principal do software do simulador. Estas soluções melhoraram o reuso de código e as múltiplas fidelidades de um modelo em um mesmo simulador;
- Proposta de interface com navegação por parâmetros dispostos em duas hierarquias: árvore RDT e modelo geométrico real;
- Utilização de navegação intuitiva e visão macro do sistema.

7.3 Contribuições de desenvolvimento

Como contribuições desta dissertação resultantes da execução da implementação, destaca-se:

- A implementação de RDT como forma de navegação em simulação operacional de satélites é única, já que não foram encontrado nenhuma indicação de utilização da teoria aplicada a simuladores;

- A utilização de critérios de avaliação de usabilidade em estado da arte para avaliação de interface.
- Aplicação do openFrameworks como base para construção de interfaces-usuário de simuladores com técnicas de RV;
- Um componente de software, denominado ofxSpacetrack foi desenvolvido no contexto desta dissertação e disponibilizado para a comunidade do openFrameworks;

7.4 Trabalhos futuros

Como trabalhos futuros desta dissertação, destaca-se:

- Utilização de uma infraestrutura de simulação padronizada, como o HLA, para distruibuição da interface e dos modelos de simulação.
- Revisão das sugestões dadas pelos entrevistados da segunda etapa e realização de uma implementação completa da vista de gráficos.
- Inclusão de uma forma de reorganização dinâmica da hierarquia da árvore.
- Utilização de outras modalidades de interação como movimentação com mãos.
- Revisão e comparação com um *engine* de jogos para verificar ganho de velocidade dado a complexidade dos modelos 3D de um satélite de grande porte, das interações possíveis e da velocidade de desenvolvimento.
- Avaliação com um maior número de usuários, e outros perfis.
- Compilação de uma versão da interface para outras arquiteturas de *desktops* e dispositivos móveis.
- Estudo das formas de validação das representações para com o conteúdo representado.
- Comparação da estrutura de dados de comportamento de subsistemas do satélite com base nas normas dos padrões ECSS e CCSDS.

REFERÊNCIAS BIBLIOGRÁFICAS

AMBROSIO, A. M. **Especificação geral de requisitos do simulador SIMC3 para os satélites CBERS3&4**. São José dos Campos: INPE, 2012. (RT-SRS-1021/00). Documento interno. 39 p. 44, 45, 46, 47, 48, 49, 50, 51, 52, 53

AMBROSIO, A. M.; CARDOSO, P. E.; NETO, J. B. Brazilian satellite simulators: previous solutions trade-off and new perspectives for the cbers program. In: 8TH INTERNATIONAL CONFERENCE ON SPACE OPERATIONS, 2006-06-19/23, Rome, Italy. **Proceedings...** [S.l.], 2006. p. 7. Acesso em: 14 abr. 2014. 2, 41, 42

AMBROSIO, A. M.; HOFFMANN, L. T.; BARRETO, J. **Relatório técnico do simulador CBERS3**: versão treinamento de operadores - protótipo 2. São José dos Campos: [s.n.], 2009. (RTD-TRP-1080/00). Documento interno. 42 p. 54, 120

AMBROSIO, A. M.; LÉO, S. C.; COSTA, M. **Interface usuário do simulador CBERS3**: versão treinamento de operadores. São José dos Campos: [s.n.], 2007. (RTD-SRS-1004/00). Documento interno. 14 p. 54, 120

ANALYTICAL GRAPHICS. **Systems Tool Kit (STK)**. 2014. Online. Disponível em: <http://www.agi.com/>. Acesso em: 04/01/2014. 12, 14, 15

AZEVEDO, D. N. R.; HOFFMANN, L. T.; AMBROSIO, A. M.; PERONDI, L. F. Analysis of the simulation model platform adoption in the context of inpe simulators. In: WORKSHOP ON SIMULATION & EGSE FACILITIES FOR SPACE PROGRAMMES (SESP), 25-27 September 2012, Noordwijk. **Proceedings of ...** 2012. Disponível em: <http://urlib.net/dpi.inpe.br/plutao/2012/11.28.13.56.51>. Acesso em: 03 fev. 2014. 2

BERG, I. **muparser - fast math parser library**. 2014. Online. Disponível em: <http://muparser.beltoforion.de/>. Acesso em: 27/01/2014. 119, 132, 141

BOTEGA, L. C.; CRUVINEL, P. E. Realidade virtual: Histórico, conceitos e dispositivos. **Aplicações de Realidade Virtual e Aumentada.**, SBC-Sociedade Brasileira de Computação, Porto Alegre, p. 8–30, 2009. 4, 17, 19, 55

CARDOSO, A.; JúNIOR, E. A. L.; OLIVEIRA, C. K. adn Luciene Chagas de; MARQUES, L.; LOPES, R. A. Visão geral de um framework para realidade aumentada em diferentes níveis de abstração. In: WORKSHOP DE REALIDADE VIRTUAL E AUMENTADA 2012 - WRVA. **Anais do ...** [S.l.]: SBC, 2012. 106

CELESTIA DEVELOPMENT TEAM. **Celestia**. 2010. Online. Disponível em: <http://216.231.48.101/celestia/>. Acesso em: 04/01/2014. 12, 14

CERQUEIRA, C. S.; KIRNER, C. Introdução à utilização de openframeworks para o desenvolvimento de aplicações de rva. **Tendências e Técnicas em Realidade Virtual e Aumentada**, v. 3, p. 250–278, Maio 2013. Disponível em: <http://plutao.sid.inpe.br/dpi.inpe.br/plutao/2013/05.31.19.24.23>. Acesso em: 14 abr. 2014. 153

CERQUEIRA, C. S.; SANTOS, W. A. dos; AMBROSIO, A. M. Development of an interface to a spacecraft simulator empowered by virtual reality. **SBC Journal on 3D Interactive Systems**, v. 3, n. 3, p. 37–44, 2012. Disponível em: <http://seer.ufrgs.br/jis/article/view/32103>. Acesso em: 14 abr. 2014. 30, 31, 75, 153

_____. Serious game interaction techniques applied to an operational satellite simulator. In: SIMPÓSIO BRASILEIRO DE JOGOS E ENTRETENIMENTO DIGITAL, 12., 2013. **Proceedings ...** SBC, 2013. p. 437–445. ISBN 2179-2259. Disponível em: <http://www.sbgames.org/sbgames2013/proceedings/artedesign/53-dt-paper.pdf>. Acesso em: 14 abr. 2014. 20, 35, 46, 66, 153

CRAWFORD, C. **Chris Crawford on interactive storytelling**. Berkeley, CA: New Riders, 2004. Disponível em: <http://www.peachpit.com/bookstore/product.asp?isbn=0321278909&rl=1>. 25

DENNIS, A.; WIXOM, B.; TEGARDEN, D. **Systems analysis and design with UML version 2.0**: an object-oriented approach. John Wiley & Sons, 2005. ISBN 9780471348061. Disponível em: <http://books.google.com.br/books?id=eelQAAAAMAAJ>. 26, 86

DEPARTMENT OF DEFENSE (DOD). **DoD Modeling and Simulation Glossary**. 2010. Disponível em: <http://www.msco.mil/files/Draft_MS_Glossary_March_B_version.pdf>. Acesso em: 04/01/2013. 9, 10, 14, 15

DISNEY. **Tron Legacy**. 2014. Online. Disponível em: <http://www.disney.pt/tron/>. Acesso em: 27/01/2014. 59

DOMINGUES, D. G. **O uso de metáforas na computação**. Dissertação (Mestrado) — Escola de Comunicações e Artes, Universidade de São Paulo, 2001. Disponível em: <EscoladeComunicaç~oeseArtes,UniversidadedeS~aoPaulo>. Acesso em: 27/01/2014. 27

EICKHOFF, J. **Simulating spacecraft systems h**. Springer-Verlag Berlin Heidelberg, 2009. (Springer aerospace technology). ISBN 9783642012761. Disponível em: <http://books.google.com.br/books?id=uHFizM24E5sC>. 1

EID, P. M. **TViz**: a taxonomy visualization tool. Dissertação (Mestrado) — Concordia University, 2005. Disponível em: <http://spectrum.library.concordia.ca/8250/>. 36, 37

ENNEX CORPORATION. **Digital Asset and FX Exchange Schema**. 2014. Online. Disponível em: <http://www.ennex.com/~fabbers/StL.asp>. Acesso em: 27/01/2014. 68

EUROPEAN COOPERATION FOR SPACE STANDARDIZATION (ECSS). **ECSS-E-ST-70C - Space engineering - ground systems and operations**. Noordwijk, The Netherlands: ESA-ESTEC, 2008. Disponível em: <http://www.ecss.nl/>. Acesso em: 04/01/2013. 2, 3

_____. **ECSS-E-TM-10-21A - Space engineering - system modelling and simulation**. Noordwijk, The Netherlands: ESA-ESTEC, 2010. Disponível em: <http://www.ecss.nl/>. Acesso em: 04/01/2013. 1, 13, 16

FOLEY, J. D.; DAM, A. van; FEINER, S. K.; HUGHES, J. F. **Computer graphics**: principles and practice. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1990. ISBN 0-201-12110-7. 11, 32, 33, 35

GRUDIN, J. Consistency, standards, and formal approaches to interface development and evaluation: A note on wiecha, bennett, boies, gould, and greene. **ACM Trans. Inf. Syst.**, ACM, New York, NY, USA, v. 10, n. 1, p. 103–111, jan. 1992. ISSN 1046-8188. Disponível em: <http://doi.acm.org/10.1145/128756.128760>. 24

HOFFMANN, L. T.; PERONDI, L. F. Estudo de simuladores computacionais aplicados ao ciclo de desenvolvimento de plataformas orbitais. In: **Anais do ...** [S.l.]: INPE, 2010. 1, 143

_____. Increasing the user interactivity in spacecraft simulations. In: SIMULATION AND EGSE FACILITIES FOR SPACE PROGRAMMES (SESP 2012), 25-27 September 2012, Noordwijk. **Proceedings of ...** [S.l.]: SESP 2012, 2012. 3

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. **ISO 9241-11**: Ergonomic requirements for office work with visual display terminals (vdts) – part 9: requirements for non-keyboard input devices. [S.l.], 2000. 24, 25

JAMIESON, S. Likert scales: how to (ab)use them. **Medical Education**, Blackwell Science Ltd, v. 38, n. 12, p. 1217–1218, dez. 2004. ISSN 0308-0110. Disponível em: <http://dx.doi.org/10.1111/j.1365-2929.2004.02012.x>. 95

JEONG, C.-S.; PANG, A. Reconfigurable disc trees for visualizing large hierarchical information space. In: IEEE SYMPOSIUM ON INFORMATION VISUALIZATION, 1998. **Proceedings of ...** [S.l.], 1998. p. 19–25, 149. 36, 37, 38, 71

KHRONOS GROUP. **The StL Format**. 1989. Online. Disponível em: <https://collada.org/>. Acesso em: 27/01/2014. 71

KIM, G. **Designing virtual reality systems**: the structured approach. Springer, 2005. (Designing Virtual Reality Systems: The Structured Approach, v. 1). ISBN 9781852339586. Disponível em: <http://books.google.com.br/books?id=_rPieb32VtMC>. 27, 28, 34, 86

KIRNER, C. Prototipagem rápida de aplicações interativas de realidade aumentada. In: **Tendências e Técnicas em Realidade Virtual e Aumentada**. Porto Alegre: SBC, 2011. p. 29–54. Disponível em: <http://ckirner.com/download/capitulos/Prot-Rapida-RA-CK.pdf>. 21, 22

KIRNER, C.; CERQUEIRA, C. S.; KIRNER, T. G. Using augmented reality cognitive artifacts in education and virtual rehabilitation. In: **Virtual Reality in Psychological, Medical and Pedagogical Applications**. [S.l.]: InTech, 2012. p. 247–270. ISBN 9789535107. 21, 22

KIRNER, C.; KIRNER, T. Development of online educational games with augmented reality. In: 5TH INTERNATIONAL CONFERENCE ON EDUCATION AND NEW LEARNING TECHNOLOGIES. **Proceedings ...** [S.l.]: IATED, 2013. (5th International Conference on Education and New Learning Technologies), p. 1950–1959. ISBN 978-84-616-3822-2. ISSN 2340-1117. 91

LANDAY, J. A.; MYERS, B. A. Sketching interfaces: Toward more human interface design. **Computer**, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 34, n. 3, p. 56–64, mar. 2001. ISSN 0018-9162. Disponível em: <http://dx.doi.org/10.1109/2.910894>. 27

LEE, G. S. Towards an integration of computer simulation with computer graphics. In: WESTERN COMPUTER GRAPHICS SYMPOSIUM. **Proceedings of ...** Banff, AB, Canada, 1999. p. 102–108. 11, 12, 13, 106

MAGALHÃES, R. O. d. **Estudo de avalanche térmica em um sistema de carga e descarga de bateria em satélites artificiais**. 171 p. Tese (Doutorado) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2012-02-15 2012. Disponível em: <http://urlib.net/sid.inpe.br/mtc-m19/2012/01.16.14.31>. Acesso em: 30 jan. 2014. 122

MARTINS, V. F.; KIRNER, T. G.; KIRNER, C. Estado da arte de avaliação de usabilidade de aplicações de realidade aumentada no brasil. In: WORKSHOP DE REALIDADE VIRTUAL E AUMENTADA 2013 - WRVA. **Anais ...** [S.l.]: SBC, 2013. 92

MARVEL. **Marvel's Iron Man**. 2014. Online. Disponível em: <http://marvel.com/ironman3>. Acesso em: 27/01/2014. 59

MEEUS, J. H. **Astronomical algorithms**. [S.l.]: Willmann-Bell, Incorporated, 1991. ISBN 0943396352. 119, 132, 136

MELBOURNE, B. **The difference between a UX Designer and UI Developer**. 2011. Online. Disponível em: <http://asinthecity.com/2011/11/10/the-difference-between-a-ux-designer-and-ui-developer/>. Acesso em: 27/01/2014. 17, 63

MICROSOFT. **Chapter 4: Designing the Hilo User Experience**. 2014. Online. Disponível em: <http://msdn.microsoft.com/en-us/library/windows/desktop/ff800706.aspx>. Acesso em: 27/01/2014. 27, 29

MILGRAM, P.; TAKEMURA, H.; UTSUMI, A.; KISHINO, F. Augmented Reality: A Class of Displays on the Reality-Virtuality Continuum. In: SPIE CONFERENCE ON TELEMANIPULATOR AND TELEPRESENCE TECHNOLOGIES. **Proceedings of the ...** Boston, Massachusetts, USA, 1995. (Proceedings of SPIE, v. 2351), p. 282–292. 20

MORRIS, C.; ROTHWELL, D. Operational spacecraft simulations: present and future. **Simulation and Modelling of Satellite Systems**, p. 9–9(0), 2002. Disponível em: <http://digital-library.theiet.org/content/conferences/10.1049/ic_20020069>. 15, 75

MYERS, B. A. A brief history of human-computer interaction technology. **interactions**, ACM, New York, NY, USA, v. 5, n. 2, p. 44–54, mar. 1998. ISSN 1072-5520. Disponível em: <http://doi.acm.org/10.1145/274430.274436>. 19

NAUGHTER, P. J. **AA+ v1.57 A class framework for Computational Astronomy**. 2013. Online. Disponível em: <http://www.naughter.com/aa.html>. Acesso em: 27/01/2014. 119, 132, 136

NIELSEN, J. **Usability engineering**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993. ISBN 0125184050. 17, 19, 24, 25, 86

NOBLE, J. **Programming interactivity**: a designer's guide to processing, arduino, and openframeworks. 1st. ed. [S.l.]: O'Reilly Media, Inc., 2009. ISBN 0596154143, 9780596154141. 25, 26, 29, 33, 34, 63, 86

NORMAN, D. A. **Things that make us smart**: Defending human attributes in the age of the machine. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1993. ISBN 0-201-62695-0. 4, 17, 22, 23, 25, 26, 86

OH, J.-M.; LEE, Y. S.; MOON, N. Towards cultural user interface generator principles. In: MULTIMEDIA AND UBIQUITOUS ENGINEERING (MUE), 2011 5TH FTRA INTERNATIONAL CONFERENCE. **Proceedings of ...** [S.l.], 2011. p. 143–148. 19

OPEN SOURCE INITIATIVE. **The MIT License**. 2014. Online. Disponível em: <http://opensource.org/licenses/MIT>. Acesso em: 27/01/2014. 30

RAMAZOTTI, R.; AMBROSIO, A. M. **SYSC Model Technical Specification - CBERS3&4**. São José dos Campos: INPE, 2013. (RTD-SRS-1012/00). Documento interno. 31 p. 44

REGGESTAD, V.; PECCHIOLI, M.; MERRI, M. Virtual reality for real operations. **ESA Bulletin**, v. 148, p. 43–51, November 2011. Disponível em: <http://www.esa.int/About_Us/ESA_Publications/ESA_Publications_Bulletin/ESA_i_Bulletin_i_148_November_2011>. 2

SAFFER, D. **The Disciplines of User Experience**. 2008. Online. Disponível em: <http://www.kickerstudio.com/2008/12/the-disciplines-of-user-experience/>. Acesso em: 27/01/2014. 17, 18, 63

SCHROEDER, R. Defining virtual worlds and virtual environments. **Journal of Virtual Worlds Research**, v. 1, n. 1, 2008. Disponível em: <http://journals.tdl.org/jvwr/article/view/294>. 22

SCHWEIGER, M. **Orbiter Space Flight Simulator**. 2012. Online. Disponível em: <http://orbit.medphys.ucl.ac.uk/>. Acesso em: 04/01/2014. 12, 14

SOUZA, M. L. de Oliveira e; TRIVELATO, G. da C. Simulators and simulations: their characteristics and applications to the simulation and control of aerospace vehicles. In: INTERNATIONAL CONGRESS AND EXPOSITION OF THE MOBILITY TECHNOLOGY, 12. (SAE), 18-20 nov. 2003, São Paulo. **Anais...** São Paulo: SAEBrasil, 2003. p. 9. Acesso em: 04/01/2013. 9, 10

STEPHENSON, N. **Snow crash**. [S.l.]: Random House Publishing Group, 1992. ISBN 9780553898194. 19

STOAKLEY, R.; CONWAY, M. J.; PAUSCH, R. Virtual reality on a wim: Interactive worlds in miniature. In: SIGCHI CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS. **Proceedings of the ...** New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1995. (CHI '95), p. 265–272. ISBN 0-201-84705-1. Disponível em: <http://dx.doi.org/10.1145/223904.223938>. 34

SUTHERLAND, I. E. Sketch pad a man-machine graphical communication system. In: SHARE DESIGN AUTOMATION WORKSHOP. **Proceedings of the ...** New York, NY, USA: ACM, 1964. (DAC '64), p. 6.329–6.346. Disponível em: <http://doi.acm.org/10.1145/800265.810742>. 19

TOMINAGA, J.; AMBROSIO, A. M. **DCS Model Technical Specification - CBERS3&4**. São José dos Campos: INPE, 2011. (RTD-SRS-1006/00). Documento interno. 19 p. 44

_____. **CBERS3&4 Simulator Models Technical Specification Guidelines**. São José dos Campos: INPE, 2012. (RT-SRS-1022). Documento interno. 9 p. 42

TOMINAGA, J.; CERQUEIRA, C. S.; KONO, J.; AMBROSIO, A. M. Specifying satellite behavior for an operational simulator. In: SIMULATION AND EGSE FACILITIES FOR SPACE PROGRAMMES (SESP 2012), 25-27 September 2012, Noordwijk. **Proceedings ...** SESP 2012, 2012. Disponível em: <http://congrexprojects.com/docs/12c09_docs/s7_1400_ambrosio.pdf?sfvrsn=2>. Acesso em: 14 abr. 2014. 2, 41, 42, 43, 44, 45, 46, 153

TOMINAGA, J.; RAMAZOTTI, R.; AMBROSIO, A. M. **MWT Model Technical Specification - CBERS3&4**. São José dos Campos: INPE, 2013. (RTD-SRS-1016/00). Documento interno. 53 p. 44

TORRES, L. C.; AMBROSIO, A. M.; LÉO, S. C. **Especificação do Modelo EPSS para o Simulador dos Satélites CBERS 3&4**. São José dos Campos: INPE, 2013. (RTD-SRS-1023/00). Documento interno. 18 p. 44, 120

USABILITY BODY OF KNOWLEDGE. **Storyboard**. 2014. Online. Disponível em: <http://www.usabilitybok.org/storyboard>. Acesso em: 27/01/2014. 27, 28

VALLADO, D. A.; CRAWFORD, P.; HUJSAK, R.; KELSO, T. S. Revisiting Spacetrack Report #3. In: AIAA ASTRODYNAMICS SPECIALISTS CONFERENCE AND EXHIBIT. **Proceedings of...** [S.l.], 2006. 119, 132, 133

WEISER, M. Hot topics—ubiquitous computing. **Computer**, v. 26, n. 10, p. 71–72, 1993. ISSN 0018-9162. 22

WENZEL, S.; BERNHARD, J.; JESSEN, U. A Taxonomy of visualization techniques for simulation in production and logistics. In: CHICK, S.; SANCHEZ, P. J.; FERRIN, D.; MORRICE, D. J. (Ed.). **Proceedings ...** [S.l.], 2003. p. 729–736. xv, 11, 12, 107

WIKIPEDIA. **List of widget toolkits**. 2014. Online. Disponível em: <http://en.wikipedia.org/wiki/List_of_widget_toolkits>. Acesso em: 27/01/2014. 28

WOO, M.; NEIDER, J.; DAVIS, T.; SHREINER, D. **OpenGL programming guide**: the official guide to learning opengl version 1.2. 3rd. ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1999. ISBN 0201604582. 32

APÊNDICE A - SIMULADOR SIMPLIFICADO

Este apêndice descreve o simulador simplificado que foi construído para prover dados para a interface. Este simulador "Simula um satélite em órbita, que pode variar os modos de operação e estar susceptível a presença e ausência do sol e ter visibilidade ou não de estações terrenas". A simulação do ambiente espacial é baseada nas bibliotecas *SpaceTrack*(VALLADO et al., 2006) e AA++ (MEEUS, 1991) (NAUGHTER, 2013), já a simulação do satélite é realizada com uso biblioteca matemática muParser(BERG, 2014). A base do software foi criada sobre o openFrameworks, mesmo que sem interface gráfica, este fornece facilidades de programação em C++ que facilitam o desenvolvimento.

A.1 Modelagem do simulador simplificado

O simulador simplificado, batizado de *CBERS Slim*, consiste de duas partes: modelo do ambiente e modelo do satélite.

A.1.1 Modelo do Ambiente

O modelo do ambiente gera os dados externos ao satélite como:

- Posição no espaço, em referência ao centro da Terra, com coordenadas cartesianas (x, y e z) e em coordenadas geográficas (latitude, longitude e altitude) e o fornece o raio da Terra.
- Posição do sol, em referência ao centro da Terra, com coordenadas cartesianas (x, y e z) e o raio do Sol.
- Posição da estação terrena, em referência ao centro da Terra, com coordenadas cartesianas (x, y e z) e em coordenadas geográficas (latitude, longitude e altitude) e a presença de um satélite na visada.
- Posição do satélite, em referência ao centro da Terra, com coordenadas cartesianas (x, y e z) e em coordenadas geográficas (latitude, longitude e altitude), a visibilidade de uma estação terrena e a presença do sol.

Estes dados são utilizados pelo modelo do satélite para cálculo de parâmetros e para compor a representação visual espacial da interface com usuário. A implementação deste modelo fez uso de componentes de software prontos, como o *SpaceTrack*(VALLADO et al., 2006) e o AA++ (MEEUS, 1991) (NAUGHTER, 2013) que foram adaptados para utilização nesta dissertação.

A.1.2 Modelo do Satélite

O modelo do satélite é composto por dois modelos simplificados representando o comportamento de dois subsistemas: o subsistema de suprimento de energia e um modelo representando o comportamento geral de consumo de energia do satélite. de satélite. Foram escolhidos os modelos do subsistema de suprimento de energia e de um modelo representativo de consumo de todo o satélite. Como fonte de inspiração, os modelos simplificados foram baseados nos documentados (TORRES et al., 2013), (AMBROSIO et al., 2007) e (AMBROSIO et al., 2009).

Estes modelos são majoritariamente movidos por eventos, isto é, os modos de operação são mudados por consequência dos eventos.

A.1.2.1 Modelo do subsistema suprimento de energia

O subsistema de suprimento de energia, do inglês *Electrical Power Supply Subsystem* (EPSS), é responsável pela aquisição e manutenção das fontes de energia elétrica do satélite, controlando os painéis solares e as baterias.

De forma a simplificar este subsistema, o subsistema foi reduzido à presença de dois painéis solares, de uma bateria, e de duas chaves lógicas que controlam os estados de interligação destes equipamentos, como mostrado na Figura A.1.

Figura A.1 - Representação do modelo simplificado para controle de potência.

As chaves e os dois somadores representam as possibilidades de combinações para prover energia para o barramento principal. Assim como o satélite, cada subsistema tem os seus modos de operação. As combinações das chaves representam os modos de operação do subsistema de suprimento de energia, a saber:

- **Sun/Charging** - Satélite está recebendo energia solar e carregando a bateria.
- **Sun/Discharging** - Satélite está recebendo energia solar, mas os painéis solares não são suficientes para suprir energia, desta forma a bateria é utilizada para compor a energia para os equipamentos.
- **Eclipse/Discharging** - Satélite não está recebendo energia, assim a bateria é responsável por toda a energia do satélite.

Estes três modos operacionais podem ser tabelados, em função da existência de sol, e da potência requerida pelo barramento de forma a compor as posições das duas chaves e o respectivo modo operacional, como mostrado na Tabela A.1:

Tabela A.1 - Modos operacionais do subsistema de suprimento de energia.

hasSun?	**MainBus**	**SWD01**	**SWD02**	**OM**
Não	x	4	0	OMD04
Sim	Ibus < Ipn2	1	1	OMD02
Sim	Ibus < Ipn2 + Ipn1	2	0	OMD02*
Sim	Ibus > Ipn2 + Ipn1	3	0	OMD03

* Como simplificação, neste modo de operação, a bateria não é utilizada, nem carregada.

Os painéis solares foram modelados como duas fontes de corrente, Ipn1 e Ipn2, constantes. Dependentes apenas da iluminação para estar fornecendo energia para o barramento principal e/ou para carga da bateria. Para isto foram considerados os dados fornecidos de potência e tensão. Destes são encontrados a corrente e a resistência:

$$Ipn_x = \frac{Ppn_x}{Vpn_x} e \tag{A.1}$$

$$Rpn_x = \frac{Vpn_x}{Ipn_x} \tag{A.2}$$

Neste modelo foi feita uma normalização da potência, para que haja a mesma corrente numa tensão menor. Assim, para uma tensão de 30[V] e uma corrente de 19[A], a potência do painel que carrega a bateria (PN1) é modelada para 570[W], com uma resistência interna de 1,57[Ω]. Da mesma forma o painel PN2: 1062[W] e 0.85[Ω].

A bateria pode ser modelada por um circuito RC (MAGALHÃES, 2012). A documentação indica que a bateria possui uma capacidade de Amperes-hora e uma tensão máxima, para ser modelada num circuito RC, para isto toma-se a capacidade da bateria (Cap_{bat}), convertendo em carga elétrica (Q_{bat}) (1 coulomb é a quantidade de carga elétrica carregada pela corrente de 1 ampère durante 1 segundo):

$$Q_{bat} = 3600 * Cap_{bat} \tag{A.3}$$

Um capacitor é descrito pela capacitância, que é em função da carga e da tensão, assim a capacitância do modelo da bateria (C_{bat}) é dado por:

$$C_{bat} = \frac{Q_{bat}}{V_{bat}} \tag{A.4}$$

A resistência interna da bateria é dada pela situação desejada de máxima transferência de potência do painel solar para a bateria, logo $R_{bat} = Rpn_1$.

Ciclo de Carga

No ciclo de carga, o painel solar modelado é uma fonte de tensão (Vpn_1) com uma carga (Rpn_1) e a bateria modelada o um capacitor (C_{bat}) e uma carga (R_{bat}), como mostrado na Figura A.2 o modelo, e A.3 o resultado de uma simulação utilizando o openModelica[1] para 100.000 segundos ($5 * RC$).

[1]Disponível em https://www.openmodelica.org/

Figura A.2 - Modelo de carga no openModellica.

Figura A.3 - Curva de carga no openModellica.

No ciclo de carga, a bateria é carregada pela corrente gerada pelo PN1. Assim, as equações de carga para tensão e corrente são respectivamente A.5 e A.6.

$$Vcurrent = Vmax + (Vcurrent - Vmax) * e^{\frac{-t}{RC}} \tag{A.5}$$

$$Icurrent = \left(\frac{(Vmax - Vcurrent}{Rpn_1 + Rbat}\right) * e^{\frac{-t}{RC}} \tag{A.6}$$

Onde:

- t é o tempo em segundos transcorrido durante o ciclo de carga;
- RC[s] é a constante de carga $(Rpn_1 + Rbat) * Cbat$;
- $Vmax$ é a máxima tensão que a bateria pode atingir, no caso o valor de PN1;
- $Vcurrent$ é a tensão do capacitor no momento t do ciclo de carga;
- $Icurrent$ é a corrente do capacitor no momento t do ciclo de carga.

No SciLab [2] as equações para cálculo de tensão e corrente de carga da bateria são mostradas na Listagem A.1, necessitam do tempo no momento de carga (t) e da tensão atual (v_current) . Os parâmetros PN1_V, PN1_R, BAT_R e BAT_RC_-charge são constantes.

Listing A.1 - Código SciLab do modelo de Carga da Bateria

```
1  function [Vcurr, Icurr]=loadBattery(t, v_current)
2
3      if BAT_isCharging ~= 1 then
4        global("BAT_isCharging"); BAT_isCharging = 1;
5        global("BAT_opTime"); BAT_opTime = t;
6        global("BAT_opValue"); BAT_opValue = v_current;
7      end
8
9      //Calculate Current V
10     Vcurr = PN1_V + (BAT_opValue - PN1_V)*((%e)^(-(t - BAT_opTime)/
         BAT_RC_charge));
11     global("BAT_V_Current"); BAT_V_Current = Vcurr;
```

15

[2]Disponível em http://www.scilab.org/

```
12
13      //Calculate Current I
14      Icurr = ((PN1_V - BAT_opValue)/(PN1_R+BAT_R))*((%e)^(-(t -
            BAT_opTime)/BAT_RC_charge));
16      global("BAT_I_Current"); BAT_I_Current = Icurr;
17
18  endfunction
```

A Figura A.4 ilustra as curvas de tensão e corrente para o código de carga de capacitor.

Figura A.4 - Resposta do SciLab à função de carga.

Ciclo de descarga

O ciclo de descarga da bateria é também um circuito RC, com o barramento representado pelo resistor R_{bus}, que requer uma potência de P_{bus}. Nota-se que a potência P_{bus} "puxa" a corrente disponível na bateria, assim a corrente e R_{bus} são respectivamente:

$$I_{bus} = \frac{P_{bus}}{V_{bus}} e \tag{A.7}$$

$$R_{bus} = \frac{V_{bus}^2}{P_{bus}} \tag{A.8}$$

O P_{bus} é dado pela somatória dos valores de potência elétrica dos subsistemas eletricamente dependentes (variando de acordo com o budget elétrico do modo operacional), e V_{bus} é constante. Para este modelo a descarga de até 26[V] é considerada para o funcionamento correto do satélite.

A Figura A.5 esquematiza o circuito do modelo com o modelo RC da bateria e o resistor representando a carga do barramento. O resultado da simulação de descarga para um cenário de consumo de 1250[W] é apresentado na Figura A.6.

Figura A.5 - Modelo de descarga no openModellica.

Figura A.6 - Curva de descarga no openModellica

No ciclo de descarga, a bateria é drenada pelos equipamentos representados por R_{bus}. Assim, as equações de carga para tensão e corrente são respectivamente A.9 e A.10.

$$Vcurrent = (Vcurrent) * e^{\frac{-t}{RC_2}} \tag{A.9}$$

$$Icurrent = \left(\frac{(Vcurrent}{R_{bus} + R_{bat}}\right) * e^{\frac{-t}{RC_2}} \tag{A.10}$$

Onde:

- t é o tempo em segundos transcorrido durante o ciclo de descarga;
- RC_2[s] é a constante de descarga $(R_{bus} + R_{bat}) * C_{bat}$;
- $Vcurrent$ é a tensão do capacitor no momento t do ciclo de descarga;
- $Icurrent$ é a corrente do capacitor no momento t do ciclo de descarga.

No SciLab as equações para cálculo de tensão e corrente de descarga da bateria são mostradas na Listagem A.2.

Listing A.2 - Código SciLab do modelo de Descarga da Bateria

```
function [Vcurr, Icurr]=dischargeBattery(t, v_current)

    if BAT_isCharging ~= 2 then
            global("BAT_isCharging"); BAT_isCharging = 2;
            global("BAT_opTime"); BAT_opTime = t;
            global("BAT_opValue"); BAT_opValue = v_current;
    end
    //Calculate Current V
    Vcurr = (BAT_opValue)*((%e)^(-(t - BAT_opTime)/BAT_RC_discharge));
    global("BAT_V_Current"); BAT_V_Current = Vcurr;
    //Calculate Current I
    Icurr = -((BAT_opValue)/(BUS_R + BAT_R))*((%e)^(-(t - BAT_opTime)/
        BAT_RC_discharge))
    global("BAT_I_Current"); BAT_I_Current = Icurr;
endfunction
```

A Figura A.7 ilustra as curvas de tensão e corrente para o código de carga de capacitor.

Figura A.7 - Resposta do SciLab à função de descarga.

Simulação conjunta de carga e descarga

Combinando as funções de carga e descarga, num ciclo de uso, demonstra a utilização deste modelo simplificado.

O programa a Listagem A.3 modela o comportamento deste subsistema, e testa um período de carga completo, seguido por uma descarga de 10% e uma nova carga completa. Os valores iniciais de, $V_{curr} = 0$[V], $Vpn_1 = 30$[V], $Ppn_1 = 532$[W], $V_{bat} = 30$[V] e $BatCap = 507$[Ah], foram simulados em até oito vezes a constante de carga

Listing A.3 - Código da simulação de carga e descarga acopladas.

```
1  // RBD MODEL VARIABLES
2
3  //SAG2
4  global PN2_P; PN2_P = 992;                          //INPUT
5  global PN2_V; PN2_V = 30;                          // INPUT
6  global PN2_I; PN2_I = PN2_P/PN2_V;
7  global PN2_R; PN2_R = ((PN2_V)^2)/PN2_P;
8
9  //SAG1
10 global PN1_P; PN1_P = 570;                          // INPUT
11 global PN1_V; PN1_V = 30;                          // INPUT
12 global PN1_I; PN1_I = PN1_P/PN1_V;
13 global PN1_R; PN1_R = ((PN1_V)^2)/PN1_P;
14
15 //MAINBUS
16 global BUS_P; BUS_P = 706;                          // INPUT
17 global BUS_V; BUS_V = 30;
18 global BUS_I; BUS_I = BUS_P/BUS_V;
19 global BUS_R; BUS_R = ((BUS_V)^2)/BUS_P;
20
21 //BATTERY
22 global BAT_CAP; BAT_CAP = 50;                          // INPUT
23 global BAT_V; BAT_V = 30;                          // INPUT
24 global BAT_C; BAT_C = (3600*BAT_CAP)/BAT_V;
25 global BAT_R; BAT_R = PN1_R;
26 global BAT_RC_charge; BAT_RC_charge = (PN1_R + BAT_R)*BAT_C;
27 global BAT_RC_discharge; BAT_RC_discharge = (BUS_R + BAT_R)*BAT_C;
28 global BAT_V_Current; BAT_V_Current = 0;
29 global BAT_I_Current; BAT_I_Current = 0;
30 //BAT MODEL CONTROL
31 global BAT_isCharging;    // 1 = charging, 2 = discharging, 0 = inert
32 global BAT_opTime;
33 global BAT_opValue;
```

```
34
35 //INITIAL CONDITION
36 resVactual = 0;
37 BAT_opTime = 0;
38 BAT_isCharging = 0;
39
40 for(n = 1: 100: (8*int(BAT_RC_charge)))// (5*int(BAT_RC_charge) + 1))
41
42     if n < 5*int(BAT_RC_charge) then
43         [resV(n),resI(n)] = loadBattery(n,resVactual)
44     elseif n < ( 5*int(BAT_RC_charge) + int(BAT_RC_charge)/10 ) then
45         [resV(n),resI(n)] = dischargeBattery(n,resVactual)
46     else
47         [resV(n),resI(n)] = loadBattery(n,resVactual)
48     end
49
50     resVactual = resV(n)
51     t(n) = n/1;
52 end
53
54 clf;
55 plot(t,resV,"*r")
56 plot(t,resI,"*g")
57 xtitle('Tensão e corrente de carga no tempo','tempo [100s]')
58 legends(['Verde - Corrente [A] ','Vermelho - Tensão [V] '],[1,1],opt="lr
    ")
```

Para aproximar o tempo de carga e descarga daquele usado no simulador Analix, que não é linear e modela outros elementos do suprimento de energia, foi retirado o tempo que o simulador indica para a carga e descarga de 10%, desta forma para o modelo foram alterados os valores da capacitância da bateria, para a fase de carga, em 100[F] e para a fase de descarga, em 8000[F]. A curva da Figura X mostra a condição de carregamento da bateria, do zero ao valor total (7*RC_charge) em 2200[s], da utilização com 1500[W], com descarga de 10% (RC_discharge/10) em 1800[s] e restabelecimento da carga completa em 1000[s].

A Figura A.8 mostra a nova onda com a correção dos tempos.

A.1.2.2 Subsistema representativo de todo o satélite

Este subsistema representa, de maneira simplificada, o consumo dos subsistemas em seis modos de operação do satélite:

Figura A.8 - Curva com capacitâncias corrigidas para ficar semelhante ao tempo real de carga e descarga.

- Fotografando com sol (OM01) - Neste modo ocorre a captura de imagens durante um período de iluminação solar.
- Fotografando em eclipse (OM02) - Neste modo ocorre a captura de imagens durante um período de eclipse.
- Gravando com sol (OM03) - Neste modo ocorre a gravação das imagens durante um período de iluminação solar.
- Gravando em eclipse (OM04) - Neste modo ocorre a gravação das imagens durante um período de eclipse.
- Stand-by (OM05) - Neste modo o satélite está em modo de espera.
- Playback (OM06) - Neste modo são realizadas as transmissões dos dados para as estações terrenas.[3]

A Tabela A.2, representa a combinação dos modos operacionais com os valores de potência consumida e um modo operacional para cada subsitema apresentado,

[3]Obs.: Este modo não consta da documentação disponibilizada, mas está presente no Analix.

isto foi feito apenas para ilustrar as mudanças do comportamentos. Esses resultados evolutivos dos estados (modos operacionais) e dos parâmetros dependentes do tempo (potências, posições) são colhidos pela interface que transformam dados brutos de simulação em representações visuais.

Tabela A.2 - Relação do modo operacional, com consumo e modos operacionais dos subsistemas.

OM	PW	SS1	SS2	SS3	SS4	SS5	SS6	SS7	SS8	SS9	Serviço
01	1478	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON
02	1478	OFF	ON	OFF	OFF	ON	ON	OFF	ON	ON	ON
03	1097	ON	ON	ON	ON	OFF	OFF	OFF	ON	ON	ON
04	1097	OFF	OFF	OFF	OFF	ON	ON	ON	ON	ON	ON
05	0784	OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON	ON	ON
06	1232	OFF	OFF	OFF	OFF	ON	ON	ON	ON	ON	ON

A comunicação dos modelos de ambiente e do satélite, neste simulador simplificado, ocorre de duas formas:

- Via modelo simplificado do subsistema de suprimento de energia, através da consulta ao parâmetro que indica se o satélite está recebendo luz solar.
- Via modelo representativo de todo o satélite, como a consulta, do modo operacional de playback, utilizando o parâmetro que indica se o satélite está em visada de alguma estação terrena.

A interface com o usuário utiliza a evolução dos valores dos parâmetros do modelo no tempo para habilitar a visualização com o realismo numérico do modelo.

Os dados de órbita, posicionamento do satélite e estações terrenas são obtidos do modelo do ambiente e os dados específicos do comportamento do satélite são obtidos do modelo representativo do satélite.

A.2 Implementação do simulador simplificado

Para implementar o modelo do ambiente foram utilizados dois componentes de software, o AIAA SpaceTrack (VALLADO et al., 2006) e o AA++(MEEUS, 1991) (NAUGHTER, 2013). Para implementar o modelo do satélite foi utilizado um componente de software de resolução matemática, o muParser (BERG, 2014).

A.2.1 Simulação do ambiente espacial

A simulação do ambiente espacial calcula a posição e velocidade orbital do satélite, incluindo a visibilidade de uma dada estação e iluminação.

A.2.1.1 Órbita do satélite

Para o cálculo da órbita do satélite, foi utilizado a biblioteca disponibilizada pelo propagador do site Celestrak , chamada de SpaceTrack (VALLADO et al., 2006). A biblioteca SpaceTrack é disponibilizada gratuitamente para as linguagens C++, Matlab, Fortran e Pascal.

A Spacetrack lê como fonte de dados o TLE de uma determinada data e propaga para uma data desejada. Não é a ênfase do trabalho o estudo profundo do funcionamento dos algorítmos de propagação, contudo, foi realizada uma análise do funcionamento da biblioteca, e esta realiza a leitura do arquivo TLE, processa-o, verifica sua validade, ajusta a data para leitura do propagador, chama o propagador e este devolve com as componentes de posição e velocidade. Como mostrado no fluxograma da Figura A.9.

Figura A.9 - Fluxo de execução da biblioteca Spacetrack
Fonte: Adaptado de Vallado et al. (2006)

Para ser utilizada neste trabalho, a Spacetrack foi adaptada para orientação a objeto e encapsulada utilizando os conceitos do oF, num *wrapper* ofxSpacetrack, e distribuída gratuitamente na Internet em: https://github.com/bodusb/ofxSpacetrack.

No *wrapper* foram criadas funções, para entrar com o arquivo TLE, iniciar o processamento, fazer a gestão do tempo, do multiplicador de tempo, controlar a propagação, capturar as posições em coordenadas cartesianas e em geográficas e manusear os parâmetros orbitais. O cabeçalho da ofxSpacetrack com as principais funções está contido na Listagem A.4.

Listing A.4 - Cabeçalho do wrapper ofxSpacettrack

```
1  #pragma once
2  #include "ofMain.h"
3
4  // SATELLITE LIBRARIES
5  #include "aiaa/SGP4ext.h"
6  #include "aiaa/SGP4UNIT.h"
7  #include "aiaa/SGP4io.h"
8  // http://celestrak.com/publications/AIAA/2006-6753/
9
10
11 #define deg2rad (PI / 180.0)
12 #define rad2deg (180.0 / PI)
13 #define omega_E 1.00273791        // Earth rotations per sidereal day
14 #define secondsDay                86400.0 // 24 * 60 * 60
15 #define minutesDay                1440.0  // 24 * 60
16 #define e_R                               6378.135        // Radius of
       the Earth (km)
17 #define f                 (1.0 / 298.26)  // Ellipticity of the Earth
18 #define dAstronomicalUnit     149597870.7         // In km
19
20
21 typedef struct YMD{
22 int year;
23 int mon;
24 int day;
25 int hr;
26 int minute;
27 double sec;
28 }YMD;
29
30
31 class ofxSpacetrack{
32
33 public:
34         ofxSpacetrack();
35         ~ofxSpacetrack();
36 // Handle TLE
```

```
37          void     setFileTLE(string value){this->fileTLE = value;};
38   string          getFileTLE(){return this->fileTLE;};
39   bool   processTLE();
40
41 // Time
42   YMD    getCurrentYMD();
43   double          getCurrentJ2000();
44   double          getCurrentMFE();
45   double          convertYMDtoJ2000( YMD *value );
46   YMD    convertJ2000toYMD( double value );
47          void     setTimezone(int value);
48          int      getTimezone();
49   void   setTimeMultiplier(double value);
50   double          getTimeMultiplier();
51   void   doSimulatedTime(bool value);
52   bool   isSimulatedTime();
53 // Propagator control
54   bool   startPropagatorNow(); //start propator
55   bool   stopPropagatorNow(); //stop propagator
56   void   update();
57          void     updateTo(double toJ2000);
58 // Positions
59          ofPoint  getCurrentPoint();
60          double   getCurrentX();
61          double   getCurrentY();
62          double   getCurrentZ();
63          ofVec3f  getCurrentVelocity();
64          double   getLatitude();
65          double   *getLatitudeAddress();
66          double   getLongitude();
67          double   *getLongitudeAddress();
68          double   getAltitude();
69          double   *getAltitudeAddress();
70 // Info - http://en.wikipedia.org/wiki/Orbital_elements
71          void     setEccentricity(double v_ecc);
72          double   getEccentricity();
73          void     setSemimajorAxis(double v_a);
74          double   getSemimajorAxis();
75          void     setInclination(double v_incl_in_rad);
76          double   getInclination();
77          void     setLongAscendingNode(double v_ascNode_in_rad);
78          double   getLongAscendingNode();
79          void     setPeriapsisArgument(double v_argp_in_rad);
80          double   getPeriapsisArgument();
81          void     setMeanAnomaly(double v_mean_in_rad);
```

```
82          double   getMeanAnomaly();
83 // sat number
84          long int          getSatNumber();
85 };
```

O resultado visual desta biblioteca é visto na Figura A.10, com o correto posicionamento do satélite sobre o planeta. Este posicionamento foi validado com o próprio site Celestrak e com a posição calculada pelo simulador Analix.

Figura A.10 - Exemplo de saída do ofxSpacetrack

A.2.1.2 Posição do Sol

Para realizar o cálculo da posição do sol e saber se o satélite está iluminado ou em eclipse, foi utilizado um algoritmo astronômico do livro *Astronomical Algorithms* de Jean Meuus (MEEUS, 1991), que foi disponibilizado em C++, sob a alcunha de AA++ (NAUGHTER, 2013).

A biblioteca AA++ possibilita o cálculo da declinação e da ascensão reta da eclíptica de um astro ao redor da Terra, com estes valores, e utilizando o objeto do tipo ofQuartenium foi possível calcular a posição geográfica e cartesiana do sol. Para calcular se o satélite está iluminado ou em eclipse, foi realizada uma simples operação geométrica.

Considerando a projeção da sombra cilíndrica, mostrada na Figura A.11, foi possível

modelar uma distância relativa a uma posição do vetor sombra.

Figura A.11 - Elementos que definem uma situação de eclipse.

Desta forma, o tamanho do vetor maxDist, da Figura A.11 é dada pela Equação A.11.

$$maxDist \cong 2 * (e_R + altitude) * tan\Big(\frac{asin\Big(\frac{e_R}{e_R + altitude}\Big)}{2}\Big) \tag{A.11}$$

Onde, e_R é o raio da Terra, altitude é a altitude do satélite em relação ao raio da Terra.

Das equações disponibilizadas pelo AA+, do vetor sol normalizado, invertido e multiplico por e_R + altitude é calculado o ponto simétrico (com dimensão e_R+ altitude) ao sol, na região de sombra. A Equação A.12, é a componente que auxilia a calcular a condição de eclipse. Caso a distância do satélite até o ponto pointShadow for menor que maxDist A.11, o satélite está em eclipse.

$$pointShadow = -(e_R + altitude) * \Big(\frac{\overline{vetorSol}}{|vetorSol|}\Big) \tag{A.12}$$

Essas operações foram adicionadas na rotina de update do ofxSpacetrack, assim, a cada evolução é recalculado se o mesmo está iluminado ou em eclipse. Ao cabeçalho do ofxSpacetrack foram adicionadas métodos para capturar a posição e raio do sol, se ele é integrado no update, e se o satélite está em eclipse, com visto na Listagem A.5.

Listing A.5 - Métodos adicionados no ofxSpacetrack para disponibilizar interface aos dados do sol e iluminação.

```
1  ...
2  ofPoint  getSunPosition();
3  double   getSunRadius();
4  void     setSunCalc(bool value);
5  bool     getSunCalc();
6  bool     isSunPositionAvailable();
7
8  bool     isOnEclipse();
9  double   getHasSUN();
10 ...
```

A.2.1.3 Relação com as estações terrenas

A operação de cálculo de visada da estação terrena seguiu um procedimento similar ao cálculo de eclipse. Para visada do satélite, foi definido que ele pode comunicar-se com uma estação, com altitude GSAltitude, até o ponto de um horizonte de contato, podendo calcular um raio de visibilidade dado pela Equação A.13.

$$dist_{satHorizonte} = \sqrt{\left(e_R + altitude\right)^2 - \left(e_R + GSAltitude\right)^2} \tag{A.13}$$

Se a relação da posição atual do satélite com a posição da estação terrena for menor que $dist_{satHorizonte}$, este está em visada. A Figura A.12 apresenta uma representação geométrica do calculo da visada do satélite à uma estação terrena.

Ao ofxSpacetrack foi adicionado um vetor de estações terrenas, utilizando o container vector do STL, com a estrutura da Listagem A.6.

Listing A.6 - Estrutura de dados para manipular estações terrenas.

```
1  typedef struct groundStation{
2          int          id;
3          string       name;
4          double       latitude;
5          double       longitude;
6          double       altitude;
```

Figura A.12 - Elementos que definem uma situação de visibilidade entre satélite e estação terrena.

```
7          ofPoint geoPosition;
8          bool            enabled;
9          double          hasSAT;
10 }groundStation;
```

Ao cabeçalho da ofxSpacetrack foram adicionadas funções para configurar estações terrenas

As funcionalidades de cálculo de órbita, posição e visada do sol e das estações compõem a simulação do ambiente de operação do satélite.

A Figura A.13, exemplifica todo o funcionamento da simulação ambiental. Onde a linha amarela representa o vetor sol, a linha roxa representa o vetor sombra, a esfera roxa representa o *pointShadow*, a linha verde indica que o satélite está iluminado e mostra o vetor Sol-Satélite. Foram colocadas três estações terrenas (São José dos Campos, Cuiabá e Alcântara) e nenhuma delas está em visada, vetores com linhas em vermelho.

Adiantando o tempo, é possível encontrar um momento em que todas as estações veem o satélite, vetores com linhas em verde, e o mesmo está numa condição de

Figura A.13 - Exemplo do modelo ambiental, com satélite iluminado e fora da visada das três estações.

eclipse, vetor Sol-Satélite em vermelho, visto na Figura A.14.

Figura A.14 - Exemplo do modelo ambiental, com satélite em eclipse e com visada das três estações.

A.2.2 Simulação do comportamento do satélite

Os modelos do satélite são carregados a partir da estrutura de arquivos em XML, nesta estrutura constam em cada nó do modelo: parâmetros e expressões. Os parâmetros são variáveis e as expressões são as equações de dinâmica do modelo.

Para simular a evolução dos parâmetros através das equações carregadas em tempo de execução, foi incorporado um componente de software de análise de expressões matemáticas. Para isto foi utilizado o muParser, que é uma biblioteca escrita em C++, e possui wrappers para C e para C#.

O muParser foi escolhido pois é otimizado para velocidade, e possui operadores binários, pós e pré fixação, principais funções matemáticas, testes lógicos e possibilidade de extensão e acoplamento de novas funções. (BERG, 2014)

Durante a leitura dos nós, cada nó recebe uma instância do muParser, e aos parâmetros, sejam eles DATAVALUE ou OPERATIONALMODE, são atribuídos variáveis dentro do contexto do muParser; e às expressões são atribuídas como regras pelo muParser.

Para isso, foi construída uma classe de interface para o muParser, de forma a facilitar as operações básicas desejadas de adicionar variável, adicionar expressão, adicionar constante, listar os itens e avaliar a expressão. Como mostrado na Figura A.15.

Figura A.15 - Cabeçalho mostrando principais métodos de interface com o analisador léxico matemático muParser.

Nota-se que a adição de variável é realizada através de um ponteiro. Este ponteiro

aponta para a variável da estrutura do nó do modelo, quando as expressões são avaliadas, elas mudam o valor da variável apontada.

APÊNDICE B - ARQUITETURA DE DADOS - XML

Este apêndice apresenta uma forma de estruturar as informações de um simulador em forma de XML, de forma que seja compreendido por um sistema computacional. XML é um documento baseado em nós com tags de abertura e fechamento de contexto, ex.: <tag> </tag>. Isto torna possível a leitura por diferentes aplicações, desde que se conheça o vocabulário utilizado. Para implmentação foi utilizado o add-on do oF chamado ofxXmlSettings, pois no escopo desta dissertação, esta foi a forma mais rápida, devido ao conhecimento prévio do autor, para criação deste simulador simplificado, porém a literatura indica (HOFFMANN; PERONDI, 2010) outras possibilidades que não foram exploradas no trabalho.

B.1 Arquitetura de arquivos

Cada modelo foi descrito com uma hierarquia de arquivos. O modelo é dividido em nós, cada nó representa um bloco particular do modelo. Estes nós estão estruturados numa árvore que partem do nó inicial, por exemplo, do nó Terra, e ramifica em estações, satélites; estes ramificam em subsistemas e equipamentos.

B.1.1 Nó da estrutura

Para descrever os **nós** do modelo foi utilizada uma estrutura de arquivos que especifica o perfil, elementos filhos, os parâmetros, modelos 3D, e os comportamentos dos parâmetros. A estrutura do arquivo é apresentada na Listagem B.1.

Listing B.1 - Estrutura do nó.

```
1  <?xml version ="1.0" encoding = "UTF-8"?>
2
3  <SIMModel>
4          <SIMNodeProfile>
5          </SIMNodeProfile>
6
7          <SIMNodeLinks>
8          </SIMNodeLinks>
9
10         <SIMNodeParameters>
11         </SIMNodeParameters>
12
13         <SIMNode3DModel>
14         </SIMNode3DModel>
15
16
```

```
17          <SIMNodeExpressions>
18          </SIMNodeExpressions>
19 </SIMModel>
```

A estrutura de **descrição de perfil do nó** *SIMNodeProfile* indica as informações relativas à descoberta do perfil, como nome, sigla, código, tipo e texto informativo. A Listagem B.2 apresenta a representação da hierarquia das informações de perfil.

Listing B.2 - Estrutura do perfil do nó.

```
1 <SIMNodeProfile>
2          <id>  </id>
3          <name> </name>
4          <acronym>  </acronym>
5          <uniqueCode>  </uniqueCode >
6          <type> </type>
7          <info> </info>
8 </SIMNodeProfile>
```

O campo de tipo na descrição do perfil de um nó indica o tipo de elemento. Nessa versão os tipos permitidos são:

- **EARTH**: indica que o nó consiste no planeta Terra, é esperado, mas não obrigatório, que a raiz da simulação comece com este nó;
- **GROUNDSTATION**: indica que o nó consiste de uma estação terrena;
- **SATELLITE**: indica que o nó consiste de um satélite;
- **SUBSYSTEM**: indica que o nó consiste de um subsistema de um satélite;
- **EQUIPMENT**: indica que o nó consiste de um equipamento de um subsistema.

A **estrutura de Links** *SIMNodeLinks* indica os nós subsequentes na hierarquia do modelo. A Listagem B.3 apresenta uma representação da hierarquia de links subsequentes.

Listing B.3 - Estrutura para indicar subnós.

```
1 <SIMNodeLinks>
2          <Link> </Link>
3          <Link> </Link>
4 </SIMNodeLinks>
```

A **estrutura de parâmetros** *SIMNodeParameters* descreve cada parâmetro utilizado no nó do modelo, podendo ser de 0 à n parâmetros. A Listagem B.4 apresenta uma representação da hierarquia das estruturas de parâmetros.

Listing B.4 - Estrutura para definição de parâmetros.

```
1 <SIMNodeParameters>
2         <NodeParameter type = "">
3         </NodeParameter>
4         <NodeParameter type = "">
5         </NodeParameter>
6 </SIMNodeParameters>
```

B.1.2 Parâmetros

Cada **parâmetro** tem uma composição genérica e seu conteúdo específico, indicado pelo atributo type na abertura da *tag* NodeParameter. A Listagem B.5 apresenta a representação genérica para escrever parâmetros.

Listing B.5 - Estrutura para definir um parâmetro.

```
1 <SIMNodeParameters>
2         <NodeParameter type = "">
3                 <ParameterProfile>
4                 </ParameterProfile>
5                 <!-- Specific info -->
6         </NodeParameter>
7 </SIMNodeParameters>
```

O **perfil do parâmetro** *ParameterProfile* é similar ao perfil do nó, com campos de identificação, como apresentados estruturalmente na Listagem B.6.

Listing B.6 - Estrutura do perfil de um parâmetro

```
1 <ParameterProfile>
2         <ID> </ID>
3         <Name> </Name>
4         <Acronym></Acronym>
5         <uniqueCode>  </uniqueCode>
6         <info> </info>
7 </ParameterProfile>
```

Os tipos de nós são colocados na *tag* de início do parâmetro (NodeParameter type = " "), podendo ser:

- **DATAVALUE**: este tipo de parâmetro define uma variável quantitativa

interna do nó do modelo;

- **OPERATIONALMODE**: este tipo de parâmetro define uma variável qualitativa interna do nó do modelo.
- **EXTERNALEVENT**: este tipo de parâmetro define uma variável que aponta para um parâmetro de outro nó.

Cada tipo de nó especifica um conjunto de informações específicas.

B.1.2.1 Parâmetro do tipo DATAVALUE

O **parâmetro do tipo DATAVALUE** é interno ao nó do modelo e pode assumir os seguintes subtipos:

- **TELECOMMAND**: usado para indicar um dado do tipo telecomando;
- **TELEMETRY**: usado para indicar um dado do tipo telemetria;
- **THERMALPOWER**: usado para indicar um dado de potência dissipada em calor;
- **COMMUNICATIONPOWER**: usado para indicar um dado de potência dissipada em comunicação;
- **ELECTRICPOWER**: usada para indicar um dado de potência elétrica consumida;
- **SWITCH**: usado para indicar uma chave lógica do modelo;
- **SIMULATIONHELPER**: usado para indicar um parâmetro auxiliar para a simulação;
- **USERDEFINED**: usado para indicar um parâmetro definido pelo usuário que não se enquadra nas outras definições.

Estes subtipos não se diferenciam entre si, apenas ajudam a indicar o tipo correto do dado do parâmetro. Outras informações são o valor inicial, o máximo e mínimo threshold e a unidade do parâmetro. A Listagem B.7 apresenta uma representação da hierarquia das estruturas básicas que compõe a informação específica do parâmetro tipo DATAVALUE.

Listing B.7 - Estrutura da informação específica de um parâmetro do tipo DATAVALUE.

```
1 <DataType> </DataType>
2 <initialValue> </initialValue>
3 <MaxThreshold> </MaxThreshold>
4 <MinThreshold> </MinThreshold>
5 <ValueUnit> </ValueUnit>
```

B.1.2.2 Parâmetro do tipo OPERATIONALMODE

O **parâmetro do tipo OPERATIONALMODE** é uma variável qualitativa que indica qual o modo operacional está ativo, dentre as opções possíveis de modos. Assim é necessária uma lista de possíveis modos. Como apresentado na hierarquia da Listagem B.8.

Listing B.8 - Estrutura da informação específica de um parâmetro do tipo OPERATIONALMODE.

```
1 <initialValue> </initialValue>
2 <Modes>
3         <ModeObject>
4         </ModeObject>
5         <ModeObject>
6         </ModeObject>
7 </Modes>
```

Cada modo foi escrito como se fosse um parâmetro, reutilizando o perfil de parâmetro e um elemento (*Appearance*) que indica a natureza da aparência do modo operacional, dividido em:

- **OFF**: indica que neste modo o elemento do nó está desligado;
- **STANDBY**: indica que neste modo o elemento do nó está em stand-by;
- **FUNCTIONAL**: indica que neste modo o elemento do nó está funcionando;
- **FAILURE**: indica que neste modo o elemento do nó está em falha.

E numa versão futura:

- **OVERHEATED**: indica que neste modo o elemento do nó está superaquecendo;

- **OVERCOOLED**: indica que neste modo o elemento do nó está super-resfriado.

A Listagem B.9 apresenta uma representação da hierarquia das estruturas básicas que compõe a informação específica do parâmetro tipo OPERATIONALMODE.

Listing B.9 - Estrutura da informação de um modo operacional.

```
1 <ParameterProfile>
2         <ID> </ID>
3         <Name> </Name>
4         <Acronym></Acronym>
5         <uniqueCode>  </uniqueCode>
6         <info> </info>
7 </ParameterProfile>
8 <Appearance> </Appearance>
```

B.1.2.3 Parâmetro do tipo EXTERNALEVENT

O **parâmetro do tipo EXTERNALEVENT** é uma variável quantitativa que aponta para o valor de uma variável em outro nó. Para fazer esse link de parâmetros é necessário indicar o código único do nó e do parâmetro (dois elementos para acelerar a busca). A Listagem B.10 apresenta uma representação da hierarquia das estruturas básicas que compõe a informação específica do parâmetro tipo EXTERNALEVENT.

Listing B.10 - Estrutura da informação específica de um parâmetro do tipo EXTERNALEVENT

```
1 <NodeCode> </NodeCode>
2 <ParameterCode> </ParameterCode>
```

B.1.3 Expressões

Para externalizar a compilação do simulador, esta estrutura considera a leitura das expressões matemáticas que compõe o modelo, inclusive permitindo a inclusão de testes de verificação.

As expressões matemáticas devem conter somente os nomes dos parâmetros que foram especificados no nó.

A Listagem B.11 apresenta uma representação da hierarquia das estruturas básicas que compõe as expressões matemáticas e em Xb uma exemplificação do seu vocabulário

Listing B.11 - Estrutura de definição de expressões da dinâmica do modelo

```
1 <SIMNodeExpressions>
2         <Expression> </Expression>
3         <Expression> </Expression>
4 </SIMNodeExpressions>
```

B.1.4 Modelo 3D

Este é um trecho utilizado somente pela interface, mas compõe a estrutura do vocabulário do arquivo XML, para leitura de modelos 3D utilizados para ilustrar a interface.

Cada nó pode ter um modelo 3D, e para carregá-lo são necessários, o nome do modelo, o endereço do arquivo, rotação, escala, e o no caso dos modelos internos do satélite é necessário o braço do pivot em relação ao centro do modelo. A Listagem B.12 apresenta a hierarquia da estrutura.

Listing B.12 - Estrutura de definição de modelos 3D.

```
1 <SIMNodeModel>
2         <name></name>
3         <link></link>
4         <rotation></rotation>
5         <scale></scale>
6         <satPivot></satPivot>
7 </SIMNodeModel>
```

B.2 Leitor de XML

O oF possui duas formas diferentes de fazer leitura de arquivos XML, instanciando um tipo ofXml, ou utilizando o add-on ofxXmlSettings.

O ofXml é um *wrapper* de uma funcionalidade da biblioteca POCO , em particular Poco::XML::DOM, já o ofxXmlSettings é um complemento adicional similar ao TinyXML possuindo uma semântica de utilização mais simples do que a ofXml da POCO. Esta simplicidade motivou a escolha.

ofxXmlSettings permite operações de:

- Mudança de contexto: com as funções pushTag() e popTag();
- Leitura de arquivo: com a função loadFile();
- Leitura de valores de tags: com a função getValue();

- Leitura de valores de atributos de tags: com a função getAttribute();
- Salvamento de arquivos: com as funções addTag(), addAtribute(), setValue(), saveFile().

B.2.1 Projeto de Software

Para armazenar a estrutura de nós do modelo no simulador e na interface, foi desenvolvida uma estrutura de classes, orientada de forma que fosse genérica. A classe fábrica de modelos, chamada SIMModel, contém o link para a árvore de nós SIMModelNode. Cada nó SIMModelNode tem agregada uma classe SIMModelProfile, pode ter links subsequentes de nós associados e parâmetros do tipo SIMNodeParameter que são especializados nos 3 tipos de parâmetros: DATAVALUE com a classe SIMParameterValue, OPERATIONALMODE com a class SIMParameterOperationalMode e EXTERNALEVENT com a classe SIMParameterExternal. A classe SimParameterOperationalMode possui vários objetos do tipo SIMOperationalModeObject que representam cada um de seus objetos. A classe SIMParameterExternal agrega um link para um objeto SIMModelParameter, sendo que este não está no mesmo nó. A Figura B.1 mostra a estrutura de dados utilizada para a composição dessa hierarquia de elementos lidos a partir da estrutura de documentos em XML.

A Figura B.2 mostra o diagrama da estrutura com os acoplamentos ambientais, de resolução matemática, e mais o controle e objeto de script, para criação de uma configuração inicial e ou modificações que possam ser programadas para ocorrer durante a rodada de simulação.

Figura B.1 - Diagrama de classes da estrutura de dados.

Figura B.2 - Diagrama de classes simplificado da estrutura de dados do modelo, com o acoplamento do modelo ambiental e da análise matemática.

APÊNDICE C - PUBLICAÇÕES REALIZADAS DURANTE A ELABORAÇÃO DESTA DISSERTAÇÃO

Este apêndice apresenta os artigos publicados durante o desenvimento desta dissertação, são eles:

- *Specifying Satellite Behavior for an Operational Simulator* - apresentado e publicado no evento internacional Simulation and EGSE Facilities for Space Programmes, 2012, na ESTEC/ESA. - este artigo trata da forma de modelagem e apresentação dos documentos que descrevem os subsistemas do simulador operacionial SIMC3. (TOMINAGA et al., 2012)

- *Development of an Interface to a Spacecraft Simulator Empowered by Virtual Reality* - publicado no periódico especial da Sociedade Brasileira de Computação, o SBC Jornal on 3D Interactive Systems, 2012 - este artigo trata da exploração do uso de metáforas naturais e da estruturação da simulação utilizando SOA. (CERQUEIRA et al., 2012)

- *Introdução à utilização de openFrameworks para o desenvolvimento de aplicações de RVA* - Tendências e Técnicas em Realidade Virtual e Aumentada, 2013 - este capítulo de livro e minicurso foi publicado e apresentado no evento XV Symposium on Virtual and Agumented Reality visando introduzir o uso de openFrameworks para a comunidade de desenvolvimento. (CERQUEIRA; KIRNER, 2013).

- *Serious Game Interaction Techniques Applied to an Operational Satellite Simulator* - publicado e apresentado no evento SBGames, 2013 - este artigo trata da demonstração dos resultados da primeira etapa de avaliação deste trabalho para um público de desenvolvimento de jogos. (CERQUEIRA et al., 2013)

A primeira página de cada uma das publicações citadas encontra-se, respectivamente, a seguir, nas Figuras C.1, C.2, C.3 e C.4

Specifying Satellite Behavior for an Operational Simulator

Jun Tominaga[(1)], Christopher Cerqueira[(1)], Janio Kono[(1)], Ana Ambrosio[(1)]

Brazilian Institute for Space Research - INPE[(1)]
Av. dos Astronautas, 1758
12.227-010 – São José dos Campos – SP - Brazil
E-mail: jun@ccs.inpe.br , christophercerqueira@gmail.com , kono@dss.inpe.br , ana.ambrosio@inpe.br

ABSTRACT

To write a satellite simulator specification for a relatively simple satellite may be simple as the information can be shared among the small group of hardware experts, system engineers and software programmers. However, this becomes extremely difficult as the number of people involved increases. This paper focuses on questions related to how to write effective simulator specification documents to describe the behavior of complex satellites, in such a manner that the specification be comprehensive and clear to all the people involved in the simulator project development. Subsystem designers, system architects, electrical, mechanic and software engineers, as well as mathematicians and programmers with a diverse assortment of backgrounds and different points of view are supposed to work together towards a common goal, which is the development of an operational satellite simulator. Consequently, the greatest challenge to build a satellite simulator is to find a common language, understandable and acceptable by every stakeholder, yet powerful enough to describe the satellite behavior in sufficient detail. The current existing standards for satellite simulators were found to lack some important details concerning how to write specifications of the models, in a precise manner. This paper presents the representation of the knowledge about the subsystems behavior in every its states through well-organized tables. This solution aimed to reduce the gap in the language to represent the satellite behavior in the Specification of the Operational Simulator of the CBERS 3&4 Satellite being developed at INPE.

INTRODUCTION

One of the greatest challenges in developing an operational satellite simulator consists of specifying the behavior of the satellite to be represented in the final software product. A compromised solution must be achieved from a tradeoff between a high fidelity model with extreme complexity and a low-cost simpler implementation with a poor performance.

Currently, INPE is engaged in the development of an operational simulator of the CBERS-3&4 satellites, named SIMC3 [1] [2]. The specifications of the SIMC3 include requirements for a high-fidelity model for the electrical power consumption and the communication interfaces with the Brazilian Satellite Control System. Taking into account the difficulties in providing the huge amount of specifications and the hard work to manage the communication interfaces required for the development of this operational satellite simulator, associated with a short timeframe, the adopted development philosophy was based on prototyping and incremental coding. Once the software architecture and the kernel of the simulator are well defined, the

Figura C.1 - *Specifying Satellite Behavior for an Operational Simulator*

Development of an Interface to a Spacecraft Simulator Empowered by Virtual Reality

Christopher Shneider Cerqueira[1], Walter Abrahão dos Santos[2] and Ana Maria Ambrosio[1]
Engineering and Space Technology[1], Applied Computing Laboratory[2]
National Institute for Space Research - INPE
São José dos Campos - Brazil
christophercerqueira@gmail.com, walter.abrahao@lac.inpe.br, ana.ambrosio@inpe.br

Abstract — Due to its large territory, Brazil relies on space systems to perform a myriad of supporting activities. Space systems design requires strong modeling and simulations techniques for achieving high performance. This article describes the development of a goal-driven user interface (UI) for spacecraft distributed simulations using a service oriented paradigm and supporting different space missions. From usability problems with traditional simulators UIs, described by satellite subsystem design engineers, this work proposes three dimensional visualization, natural interaction techniques, virtual and augmented reality as well as interaction with touch-screen and gesture recognitions. This research uses an open source C/C++ toolkit, designed to provide interactivity, networking and scripting capabilities for simulator developers, this facilitates in providing specific input or output driver engines. Finally, some actual release pictures and information which includes single-touch interaction in the Smart Board Screen and some further developments are described for future work.

Index Terms — Interaction, virtual reality, simulation, interface, computer graphics, distributed systems.

I. Introduction

The space exploration generally drives research and applications to areas such as rocket propulsion, life support, new materials, reliable computer algorithms, autonomous operations, etc. The applications of space exploration include Space Observation, Earth navigation, Communication, Meteorology and Remote Sensing. All space data gathered provides resourceful information to our day-to-day life. In Brazil, space missions are mainly performed by the National Institute for Space Research (INPE) and they play a key role for the vast Brazilian territory in water, fishery, agricultural and deforestation monitoring as well as weather/climate data gathering from ground sensing data platforms or obtained from images taken by artificial satellites [1].

A space mission is divided into four segments [2]:

- Ground Segment - provides communication with Space Segment to answer user's requests with the information acquired;
- Space Segment - provides data to the Ground Segment, usually could be a satellite, probe, capsules, space telescopes and space shuttles;
- Launch Segment - places the Space Segment into the space, characterized by a rocket propelled artifact and;
- User Segment - receives and uses the acquired data, e.g. scientists, media, agricultural companies and government.

After a satellite launch, an interconnection of the segments operates like is shown in Fig.1. The Ground Segment interfaces to the User Segment and the Space Segment; it is responsible to provide reliable information to the user communities. After launch, all tasks such as control and operation of the satellites and management of the mission are handled by the Space Segment elements. The Ground Segment is subdivided into four elements [2]:

- Mission Center - handles mission concept, evaluation, analysis and Mission Exploration and Payload Data;
- Control Center -handles operation control, simulation, flight dynamics, data handling and distribution;
- Network - handles the interconnection of centers, stations and the spacecraft, using ground and space links and;
- Ground Stations - handles the communication link between the Ground Segment and the Space Segment.

Fig. 1. Ground Segment and its relations to other space mission segments

Ground Segment provides facilities and resources to control and operate the Space Segment artifacts, as satellites. Activities of the Satellite Control Center, for instance, briefly depicted in Fig. 2, are divided into flight dynamics, mission planning from user requests, acquired data distribution to user and simulations

ISSN: 2236-3297

Figura C.2 - *Development of an Interface to a Spacecraft Simulator Empowered by Virtual Reality*

Tendências e Técnicas em Realidade Virtual e Aumentada, v. 3, p. 250-278, maio/2013.
CERQUEIRA, Christopher Shneider; KIRNER, Claudio. *Introdução à utilização de openFrameworks para o desenvolvimento de aplicações de RVA*

Introdução à utilização de openFrameworks para o desenvolvimento de aplicações de RVA

Christopher Shneider Cerqueira
Claudio Kirner

Abstract

Virtual and augmented reality interactive applications development requires tools that easy and accelerates the coding time through the reuse of components. The openFrameworks is an example of tool oriented to reuse. This chapter shows the context of openFramework use and this type of tool-framework, with the exploration of three examples: one basic to explain the structure, one using additional components and the third to develop an augmented reality environment.

Resumo

Desenvolvimento de aplicações interativas de realidade virtual e aumentada requer ferramentais que facilitem e acelerem o tempo de codificação através do reuso de componentes. O openFrameworks é um exemplo deste tipo de ferramental. Neste capítulo é mostrado o contexto de utilização e outros exemplos de frameworks, a exploração de três exemplos: um básico para explicar a estrutura, um utilizando componentes adicionais e o terceiro desenvolvendo um ambiente de realidade aumentada.

1. Introdução

O desenvolvimento de sistemas interativos nem sempre é apenas uma atividade de codificação para cientistas da computação ou engenheiros. Artistas, professores, designer e estudantes dos mais variados segmentos podem se interessar em criar interações que satisfaçam as suas necessidades de apresentação ou coleta de dados, focando em aplicações que possam parecer extremamente high-tech ou utilizar metáforas de interação mais parecidas com o ambiente temático onde será implantado.

Este capítulo descreve em breves linhas, um roteiro de desenvolvimento desde a explicação de conceitos básicos de desenvolvimento interativo e criativo, os cenários atuais de alguns frameworks e a motivação de reutilizar componentes de código livre. Passa por uma explicação estrutural e dinâmica

Figura C.3 - Introdução à utilização de openFrameworks para o desenvolvimento de aplicações de RVA

Serious Game Interaction Techniques Applied to an Operational Satellite Simulator

Christopher Shneider Cerqueira[1], Walter Abrahão dos Santos[2] and Ana Maria Ambrosio[1]
Engineering and Space Technology[1], Applied Computing Laboratory[2]
National Institute for Space Research - INPE
São José dos Campos - Brazil
Email: christophercerqueira@gmail.com, walter.abrahao@lac.inpe.br, ana.ambrosio@inpe.br

***Abstract*—This work describes an initiative to a serious game-like user interface used for an operational satellite simulator. The simulator is part of a simulation architecture intended to provide a richer behavior analysis tool set to the real artefact, as a source of possible behaviors scenarios. In order to interact with the underlying behaviour, the user interface can benefit from: game interaction techniques, increasing cognitive response and, fast data recovery. This is done by extending traditional manipulation techniques and allowing the transition from menus, tables and charts components to data in three-dimensional models, cone-trees, multiple views, mini-maps, touching interface, as well as augmented reality. Finally, a short survey collects final users' satisfaction to this new concept.**

Keywords*—*User interface, simulation, games

I. Introduction

Simulation is a key element for supporting a wide range of engineering and operational activities during the life cycle and it is recognized as a good practice to generate products [1].

Simulation has many uses, such as training, testing, scientific research, manufacturing, engineering, military and entertainment. In Space programs, the European Space Agency (ESA) [1] indicates that simulation has potential uses across the life cycle, in activities such as:

- Analysis, definition and validation of systems and technical requirements.
- Design validation of high-level performance requirements from various points of view as : electrical, thermal, mechanical, operational.
- Software verification and validation.
- Development of EGSE (Electronic Ground Support Equipment) and test procedures.
- Support of units and subsystem tests activities.
- Prediction of systems performance.
- Development and validation of Operations procedure.
- Troubleshooting for Systems failures and anomalies.
- Control center and crew operator training.

In Brazil, space missions are mainly performed by the National Institute for Space Research (INPE) and they play a key role for its vast territory in water, fishery, agricultural and deforestation monitoring as well as weather/climate data gathering from ground sensing data platforms or obtained from images taken by artificial satellites [2].

Several activities in satellite simulation are performed at INPE, such as those in the Satellite Control Center, which is a key part of the Ground Segment, as shown in Figure 1.

Fig. 1. Space and the Ground Segment facilities, adapted from [3].

A satellite simulation has several stakeholders with different backgrounds that use the simulated data to acquire information, as shown in Figure 2. An example is a faulty satellite condition, which requires an inquiry to the problem by testing conditions and understanding its real behaviour. However, often during operation time, systems engineers and managers who built the satellite are in different projects, and they need to quickly re-learn the possible operational conditions in order to fix the problem. Hence, they can think on possible correction procedures with a final validation by the simulator. The satellite control team receives the engineers'

Figura C.4 - *Serious Game Interaction Techniques Applied to an Operational Satellite Simulator*