

UNIVERSIDADE FEDERAL DE SÃO CARLOS
CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

**MODELO DE ESPECIFICAÇÃO DE INTERFACES
TANGÍVEIS DE MESA – TTUI-SM**

ANTONIO MIGUEL BATISTA DOURADO

ORIENTADORA: PROF.^a DR. REGINA BORGES DE ARAÚJO

São Carlos - SP
Setembro/2012

UNIVERSIDADE FEDERAL DE SÃO CARLOS
CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

**MODELO DE ESPECIFICAÇÃO DE INTERFACES
TANGÍVEIS DE MESA – TTUI-SM**

ANTONIO MIGUEL BATISTA DOURADO

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação, área de concentração: Processamento de Imagens e Sinais.

Orientadora: Dra. Regina Borges de Araújo

São Carlos – SP
Setembro/2012

**Ficha catalográfica elaborada pelo DePT da
Biblioteca Comunitária da UFSCar**

D739me Dourado, Antonio Miguel Batista.
Modelo de especificação de interfaces tangíveis de mesa
– TTUI-SM / Antonio Miguel Batista Dourado. -- São Carlos :
UFSCar, 2012.
172 f.

Dissertação (Mestrado) -- Universidade Federal de São
Carlos, 2012.

1. Interfaces de usuário. 2. Interação homem-máquina. 3.
Especificação formal. 4. Interfaces tangíveis. I. Título.

CDD: 004.6 (20^a)

Universidade Federal de São Carlos
Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

**“Modelo de Especificação de Interfaces
Tangíveis de Mesa – TTUI-SM”**


Antonio Miguel Batista Dourado

Dissertação de Mestrado apresentada ao
Programa de Pós-Graduação em Ciência da
Computação da Universidade Federal de São
Carlos, como parte dos requisitos para a
obtenção do título de Mestre em Ciência da
Computação

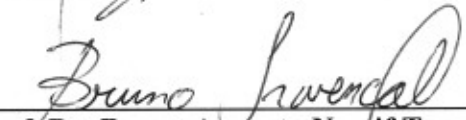
Membros da Banca:



Prof.ª Dra. Regina Borges de Araújo
(Orientadora - DC/UFSCar)



Prof. Dr. Jander Moreira
(DC/UFSCar)



Prof. Dr. Bruno Augusto Nassif Travençolo
(UFU)

São Carlos
Setembro/2012

Dedico este trabalho aos meus pais, familiares e amigos.

AGRADECIMENTO

A Deus, primeiramente, por ter me dado forças para lutar quando o desânimo veio; e sabedoria para vencer essa etapa de minha vida.

Aos meus pais e familiares, por me apoiarem na decisão de ingressar no mestrado, por me incentivarem a batalhar, pelos conselhos e pelo conforto.

À Prof.^a Dr.^a Regina B. Araújo, pela oportunidade de ingressar no mestrado, pelos conselhos acadêmicos e, principalmente, por ser uma segunda mãe para mim quando passei por dificuldades.

Agradeço ao Professor e grande amigo Leonardo Botega, por ter me ajudado e escolher este caminho, pela valiosa parceria de pesquisa e por ter me ajudado a enfrentar muitos desafios durante todo esse tempo.

A todos os meus colegas do Laboratório WINDIS, pelo tempo que passamos juntos, seja discutindo ideias, seja conversando sobre assuntos diversos, seja pelas risadas. Ao Max da IMax Games pelas grandes ideias e contribuições.

Aos meus novos amigos de São Carlos, pelo companheirismo e apoio durante toda essa fase.

Aos meus amigos de Garça, pela compreensão durante minha ausência, pelo apoio e pelos conselhos dados.

A CAPES por ter concedido uma bolsa de estudos para que eu pudesse me sustentar durante o período do mestrado.

Por fim, agradeço de forma geral a todos que contribuíram com o trabalho direta ou indiretamente e que por algum motivo não fazem mais parte de minha vida hoje.

"Pain is Temporary, Glory Lasts Forever"

Mark Allen

RESUMO

No cenário de desenvolvimento de interfaces computacionais, os avanços nas pesquisas buscam oferecer novas formas de interação que se aproximam da forma natural com que o homem interage com o mundo real. Dentre as diversas interfaces avançadas, as interfaces tangíveis de mesa (*tabletop*), promovem a ligação entre objetos físicos e objetos virtuais, possibilitando ao usuário interagir com objetos digitais por meio do ambiente físico, e também por meio de interações multitoques. Entretanto, o processo de desenvolvimento deste tipo de interface carece de um modelo de especificação que contemple, além das interações por meio de objetos, interações multitoques e que organize e classifique a especificação de uma maneira mais ágil e mais fácil de documentar e implementar. Assim, este trabalho apresenta um novo modelo de especificação de elementos de interface tangível de mesa, denominado TTUI-SM, que organiza a especificação de elementos de interface em diversos componentes. Uma ferramenta diagramática, o TTUI-SMT, baseada neste modelo de especificação, também foi desenvolvida visando agilizar, facilitar e automatizar o processo de especificação da interface e do seu desenvolvimento. Para validar o modelo e a ferramenta, dois estudos de caso foram introduzidos e especificados. Um experimento foi conduzido para avaliar o modelo e a ferramenta e, por meio de questionários, os benefícios propostos foram validados.

Palavras-chave: Interfaces Tangíveis, Interfaces Tangíveis *Tabletop*, Interface Homem-Computador, Especificação de Interfaces Tangíveis *Tabletop*, Desenvolvimento de Interfaces Tangíveis.

ABSTRACT

In the scenario of computational interfaces development, researches' efforts aim to offer new ways of interaction that are closer to the natural way which humans interact with the real world. Amongst the diversity of interface modalities, the tabletop tangible interfaces make the link between physical objects and virtual objects, making possible to "grasp" the interface and interact with it physically, also counting on multitouch interactions. However, in the development process of this kind of interface, there is a lack of specification's model that supports, not only the physical objects interaction, but multitouch interactions as well, and that organizes and classifies the specification in a more agile manner, easier to document and implement. Thus, this work presents a new tabletop tangible user interface specification model, TTUI-SM, that classifies and organizes the interface element specification within many components. A diagramatic tool, TTUI-SMT, was developed based on this model, aiming to make the interface specification and development faster, easier and automatized. To validate the model and tool, two study-cases were introduced and specified. An experiment was conducted to evaluate both model and tool, resulting in the comprovation, through questionnaires analysis, of the proposed benefits.

Keywords: Tangible Interfaces, Tabletop User Interfaces, Computer-Human Interface, Tabletop Tangible Interface Specification, Tangible Interface Development.

LISTA DE FIGURAS

Figura 1 - Sketchpad (SUTHERLAND, 1963).....	28
Figura 2 - Interface Gráfica do Xerox Star (JOHNSON et al., 1989)	29
Figura 3 - <i>Binocular Omni-Orientation Monitor (BOOM)</i> (MÁRQUEZ, 2002).....	31
Figura 4 - Protótipo da Rutgers Master II (GOMEZ; BURDEA; LANGRANA, 1995) .	33
Figura 5 - Enciclopédia de Compositores Clássicos (SAUER; ENGELS, 1999)	35
Figura 6 - Interface de Toque (NACENTA; BAUDISCH; BENKO, 2009).....	37
Figura 7 - Microsoft Kinect sendo usado para reconhecer linguagem de sinais (ZAFRULLA; BRASHEAR; STARNER, 2011)	38
Figura 8 - Interface Móvel para auxílio de dietas (KIM et al., 2010)	39
Figura 9 - Realidade Aumentada aplicada a uma mesa (KIRNER; SISCOOTTO, 2007)	40
Figura 10 - Celular com interface de realidade aumentada (MOHRING et. al., 2004)	41
Figura 11 – (a) <i>Optical See-Through</i> (Adaptado de HUCKAUF et. al., 2010) (b) <i>Video See-Through</i> (Adaptado de HAYASHI; KATO; NISHIDA, 2005)	42
Figura 12 – Realidade Aumentada com Display Baseado em Projeção (EHNES et. al., 2004)	42
Figura 13 – Ciclo de uma interface de realidade aumentada (Adaptado de KLINKER et. al., 2000)	43
Figura 14 – Aplicação de Realidade Aumentada utilizando rastreamento por sensor. (a) Visualização do Usuário (b) Arquitetura da Aplicação. (Adaptado de NEWMAN et. al., 2001)	44
Figura 15 – Três gerações de marcadores fiduciais (NAIMARK; FOXLIN, 2002)	44
Figura 16 – Utilização de rastreamento sem marcadores (<i>Markerless</i>) para detecção de bordas de casas (REITMAYR; DRUMMOND, 2006).....	45
Figura 17 - Arquitetura de uma Interface Multimodal (Adaptado de DUMAS; LALANNE; OVIATT, 2009).....	47
Figura 18 - Interface do Tangible Geospace (ISHII; ULLMER, 1997)	49
Figura 19 - <i>Marble Answering Machine</i> (POYNOR, 1995)	50
Figura 20 - Interface do <i>MagicBook</i> . (a) Caderno físico real (b) Caderno físico com RA. (Adaptado de BILLINGHURST et. al., 2001)	51

Figura 21 - Objeto pertencente ao sistema do Nimio reagindo (BREWER; WILLIAMS; DOURISH, 2007).....	52
Figura 22 - "Faça um Enigma" (esquerda) e "Tele História" (direita) (Adaptado de HUNTER e MERRILL, 2010).....	53
Figura 23 - Interface da Reactable	54
Figura 24 - Arquitetura de uma interface tangível utilizando o protocolo TUIO (Adaptado de TUIO, 2011)	56
Figura 25 - Atributos das mensagens SET do TUIO (Adaptado de TUIO, 2011)	57
Figura 26 - <i>Profiles</i> do TUIO (TUIO, 2011).....	58
Figura 27 - Marcadores Fiduciais (REACTIVISION, 2011)	59
Figura 28 - Critérios de Frameworks Multitoques (Adaptado de KAMMER et. al., 2010)	62
Figura 29 - Modelo Cascata (Adaptado de PRESSMAN, 2001).....	64
Figura 30 - Modelo Estrela (Adaptado de HARTSON e HIX, 1989)	65
Figura 31 - Protótipo <i>Tabletop</i> WINDIS.....	66
Figura 32 - Arquitetura <i>tabletop</i> escolhida (Adaptado de TUIO, 2011).....	68
Figura 33 - Visão geral de ferramenta de criação de protótipo visual de interface tabletop em tempo real.....	77
Figura 34 - Modelo Hierárquico de Elemento de Interface TTUI-SM.....	79
Figura 35 – Componentes de um elemento de interface.....	80
Figura 36 – Layout da Ficha de Elemento de Interface.....	87
Figura 37 - Criação de um novo modelo TTUI-SM (Parte 1).....	89
Figura 38 - Criação de um novo modelo TTUI-SM (parte 2)	90
Figura 39 - Criação de um novo modelo TTUI-SM (Parte 3).....	90
Figura 40 - Interface da Ferramenta TTUI-SMT: (a) Arquivos TTUI-SM (b) Área de Diagramação (c) Paleta de Elemento de Interface (d) Propriedades	91
Figura 41 - Paleta de Elemento de Interface.....	92
Figura 42 - Nó de Elemento " <i>Interface Element</i> ".....	93
Figura 43 - Propriedades do nó <i>Interface Element</i>	94
Figura 44 - Exemplo de Especificação Diagramática TTUI-SM.....	95
Figura 45 - Fluxo de geração de código a partir da especificação gerada pelo TTUI-SMT.....	96
Figura 46 - Caminho de geração de arquivo de fichas de elemento de interface.....	96

Figura 47 - Geração de arquivos de fichas de elemento de interface e definições XML.....	97
Figura 48 - Geração de Código por meio do XML.....	98
Figura 49 - (a) Modelo Cascata Original (b) Modelo Cascata com TTUI-SM (Adaptado de PRESSMAN, 2001).....	100
Figura 50 - (a) Modelo Estrela original (b) Modelo Estrela com TTUI-SM (Adaptado de HARTSON e HIX, 1989).....	101
Figura 51 - Organização básica do protocolo padrão (Adaptado de BOSSONARO et al., 2011)	103
Figura 52 – Protocolo Padrão potencializado (Adaptado de BOSSONARO et al., 2011)	104
Figura 53 – Camadas de Visualização da Interface Avançada de Controle e Comando. (a) Controle de Pessoal (b) Inteligência (c) Operações (d) Logística.....	105
Figura 54 - Disposição do Diagrama de Especificação	108
Figura 55 – <i>Part “Media Frame”</i> e suas interações	109
Figura 56 – Elemento de Interface “ <i>Sink</i> ” e seus <i>States</i>	110
Figura 57 – Elemento de Interface “ <i>Sensor Network</i> ” e seus <i>Listeners</i>	111
Figura 58 - Ficha do Elemento de Interface Topografia	114
Figura 59 - Esboço da Interface Voo Interativo	115
Figura 60 - Diagrama de Especificação do Estudo de Caso Voo Interativo	117
Figura 61 - Elemento de Interface UAV com duas <i>Parts</i>	118
Figura 62 - Elemento de Interface Waypoint com suas <i>Parts</i> e <i>Listeners</i>	119
Figura 63 - <i>Part Map</i> do Elemento de Interface Map com suas interações.....	120
Figura 64 - Ficha do Elemento de Interface Waypoint	121
Figura 65 – (a) <i>Pictures Demo</i> do Kivy (KIVY, 2010) e (b) <i>Reactable</i>	123
Figura 66 - Resultado do Questionário TTUI-SM	125
Figura 67 - Seções mais difíceis por nível de conhecimento.....	126
Figura 68 - Seções mais fáceis por nível de conhecimento	126
Figura 69 - Ilustração do Modo de Visualização da Ferramenta TTUI-SM.....	133
Figura 70 - Elemento de Interface da Camada de Controle de Pessoal	146
Figura 71 - Elemento de Interface Sentinela	148
Figura 72 – Elemento de Interface Comandante.....	149
Figura 73 - Elemento de Interface Patrulheiro.....	150

Figura 74 - Elemento de Interface Força de Reação.....	151
Figura 75 - Elemento de Interface Funcionário	152
Figura 76 - Elemento de Interface da Camada de Inteligência	153
Figura 77 - Elemento de Interface Redes de Sensores.....	155
Figura 78 - Elemento de Interface Rota de Robô	157
Figura 79 - Elemento de Interface VANT	160
Figura 80 - Elemento de Interface Mesa de Comando e Controle	161
Figura 81 - Elemento de Interface da Camada de Operações	163
Figura 82 - Ficha do Elemento de Interface Media Frame	169
Figura 83 - Ficha do Elemento de Interface Map	170
Figura 84 - Ficha do Elemento de Interface UAV	172

LISTA DE TABELAS

Tabela 1 - Comparativo das características dos <i>trackers</i>	61
Tabela 2 - Comparativo entre os <i>Application Frameworks</i>	63
Tabela 3 - Resultados do Questionário.....	124
Tabela 4 - Seções difíceis e fáceis.....	125

LISTA DE ABREVIATURAS E SIGLAS

API – *Application Programming Interface*

AUIML – *Abstract User Interface Markup Language*

BOOM – *Binocular Omni-Orientation Monitor*

CAD – *Computer-aided design*

CLG – *Command Language Grammar*

COMPSI – *Computing and Information Systems Research Lab*

DI – *Diffused Illumination*

DSI – *Diffused Surface Illumination*

EMF – *Eclipse Modeling Framework*

FML – *Face Modeling Language*

FPS – *Frame Per Second*

FTIR – *Frustrated Total Internal Reflexion*

GDL – *Gesture Definition Language*

GMF – *Graphical Modeling Framework*

GUI – *Graphical User Interface*

HMD – *Head Mounted Display*

IDE – *Integrated Development Environment*

IHC – *Interação Homem-Computador*

INCT-SEC – *Instituto Nacional de Ciência e Tecnologia em Sistemas Embarcados Críticos*

InTML – *Interaction Technique Markup Language*

JET – *Java Emitter Template*

LCD – *Liquid Crystal Display*

LED – *Light-Emitting Diode*

LLP – *Laser Light Plane*

MCRpd – *Model-Control-Representation physical digital*

MIT – *Massachusetts Institute of Technology*

MVC – *Model View Controller*

OSC – *Open Sound Control*

OST – *Optical See-Through*

PDA – *Personal Digital Assistant*

RA – *Realidade Aumentada*

RFID – *Radio Frequency IDentification*

RSSF – *Redes de Sensores Sem Fio*

RTI – *Robôs Táticos Internos*

SGML – *Standard Generalized Markup Language*

SMS – *Short Message Service*

TAC – *Tokens and Constrains*

TAG – *Task Action Grammar*

TCP – *Transmission Control Protocol*

TTUI-SM – *Tabletop Tangible User Interface Specification Model*

TTUI-SMT – *Tabletop Tangible User Interface Specification Model Tool*

TUIML – *Tangible User Interface Markup Language*

TUIO – *Tangible User Interface Objects*

UAV – *Unmanned Aerial Vehicle*

UDP – *User Datagram Protocol*

UIML – *User Interface Markup Language*

USIXML – *USer Interface eXtensible Markup Language*

VANT – *Veículo Aéreo Não Tripulado*

VST – *Video See-Through*

VUI – *Voice User Interface*

WIMP – *Windows, Icons, Menus and Pointers*

WINDIS – *Wireless Networks and Distributed Interactive Simulation*

XIML – *eXtensible Interface Markup Language*

XML – *eXtensible Markup Language*

Sumário

CAPÍTULO 1 - INTRODUÇÃO.....	20
1.1 Motivação.....	22
1.2 Objetivos e Metas.....	23
1.3 Metodologia.....	24
1.4 Organização.....	25
CAPÍTULO 2 - INTERFACES COMPUTACIONAIS PÓS WIMP.....	27
2.1 Interfaces Pós WIMP.....	29
2.1.1 Interfaces de Realidade Virtual.....	30
2.1.1.1 Visual.....	30
2.1.1.2 Auditiva.....	32
2.1.1.3 Tátil.....	32
2.1.1.4 Navegação.....	33
2.1.2 Interfaces Multimídia.....	34
2.2 Interfaces de Voz, Fala, Gestos, Escrita e Toques.....	35
2.2.1 Interfaces de Voz e Fala.....	35
2.2.2 Interfaces de Gestos, Escrita e Toque.....	36
2.3 Interfaces Móveis.....	38
2.4 Interfaces de Realidade Aumentada.....	39
2.4.1 Dispositivos de Visualização.....	40
2.4.2 Rastreamento.....	42
2.5 Interfaces Multimodais.....	45
2.5.1 Arquitetura.....	46
2.6 Interfaces Tangíveis.....	48
2.6.1 Realidade Aumentada Tangível.....	50
2.6.2 Monitores Ambientes.....	51
2.6.3 Interfaces de Usuário Incorporadas.....	52
2.6.4 Interface Tangível <i>Tabletop</i>	53
2.7 Considerações Finais.....	54
CAPÍTULO 3 - DESENVOLVIMENTO DE INTERFACES TANGÍVEIS <i>TABLETOP</i>.....	55

3.1 Ferramentas de Suporte ao Desenvolvimento para Interfaces Tangíveis <i>Tabletop</i>	55
3.1.1 TUIO	56
3.1.2 Especificação do TUIO	56
3.1.2.1 Mensagens	57
3.1.2.2 <i>Profiles</i>	58
3.1.2.3 Eficiência e Confiabilidade	58
3.1.3 <i>Trackers</i>	59
3.1.4 <i>Application Frameworks</i>	61
3.2 Desenvolvimento <i>Tabletop</i> e Metodologias de Desenvolvimento	63
3.3 Protótipo <i>Tabletop</i> do laboratório WINDIS	65
3.3.1 Ferramental de Desenvolvimento das Aplicações do Protótipo	67
3.4 Considerações Finais	68
CAPÍTULO 4 - TRABALHOS RELACIONADOS	69
4.1 Trabalhos Relacionados	69
4.2 Discussão	73
4.3 Considerações Finais	75
CAPÍTULO 5 - MODELO DE ESPECIFICAÇÃO DE ELEMENTOS DE INTERFACE TTUI-SM	76
5.1 Modelo de Especificação de Elementos de Interface TTUI-SM	77
5.1.1 Diagrama de Elemento de Interface	78
5.1.1.1 Dados Gerais do Elemento de Interface	80
5.1.1.2 <i>Parts</i> e <i>Interactions</i> do Elemento de Interface	81
5.1.1.3 <i>States</i> do Elemento de Interface	83
5.1.1.4 API do Elemento de Interface	84
5.1.1.5 <i>Listeners</i> do Elemento de Interface	84
5.1.1.6 <i>Connections</i> do Elemento de Interface	85
5.1.2 Ficha de Elemento de Interface	86
5.2 Ferramenta de Especificação TTUI-SMT	87
5.2.1 Utilização da Ferramenta TTUI-SMT	88
5.2.1.1 Criação de Novo Modelo TTUI-SM	88
5.2.1.2 Interface de Especificação de Elementos de Interface	91
5.2.2 Geração de Ficha de Elemento de Interface e Definições XML	95

5.3 Modelo TTUI-SM e Metodologia de Desenvolvimento	99
5.4 Considerações Finais	99
CAPÍTULO 6 - AVALIAÇÃO DO MODELO TTUI-SM	102
6.1 Estudo de Caso: Vigilância e Segurança de Infraestruturas Críticas	102
6.1.1 Identificação dos Elementos de Interface	105
6.1.1.1 Camada de Topografia	106
6.1.1.2 Camada de Controle de Pessoal	106
6.1.1.3 Camada de Inteligência	106
6.1.1.4 Camada de Operações	107
6.1.2 Especificação dos Elementos de Interface	107
6.2 Estudo de Caso: Voo Interativo	114
6.2.1 Identificação dos Elementos de Interface	115
6.2.1.1 Elemento de Interface Mapa (<i>Map</i>)	116
6.2.1.2 Elemento de Interface VANT (<i>UAV</i>)	116
6.2.1.3 Elemento de Interface Waypoint	116
6.2.1.4 Elemento de Interface Frame de Mídia (<i>MEDIA FRAME</i>)	116
6.2.2 Especificação dos Elementos de Interface	116
6.3 Experimento TTUI-SM	122
6.3.1 Etapas do Experimento de Avaliação	122
6.3.2 Resultados e Discussão	123
6.4 Considerações Finais	128
CAPÍTULO 7 - CONCLUSÕES	130
7.1 Limitações	131
7.2 Trabalhos Futuros	131
7.2.1 Tratamento de Exceções	132
7.2.2 Modo de Visualização	132
7.2.3 Validação automatizada da ferramenta TTUI-SM	133
7.2.4 Criação de Parsers e Implementação de Interfaces	134
7.3 Contribuições	134
REFERÊNCIAS BIBLIOGRÁFICAS	136
ANEXO A	143
ANEXO B	145

ANEXO C	164
ANEXO D	168

Capítulo 1

INTRODUÇÃO

O constante desafio de aprimorar a forma de interação do homem com o computador tem levado à busca de interfaces naturais devido ao surgimento de diferentes tecnologias de interação, além de modelos, técnicas e ferramentas de especificação de interfaces.

No início, com a chegada das máquinas e a simplificação das tarefas que os usuários realizam por meio delas, era comum que o usuário se adaptasse às maneiras de interação oferecidas por elas, a fim de realizar suas tarefas. Com o tempo, surgiu a preocupação de criar novos meios de interação com as máquinas, de forma a se aproximar cada vez mais das formas de interação naturais do homem com o mundo. Diante desta preocupação, surgiram diversos tipos de interfaces.

Dentre estes diferentes tipos, há as Interfaces WIMP (*Windows, Icons, Menus e Pointers* – Janelas, Ícones, Menus e Ponteiros) incluindo recursos que facilitam a interação do usuário com as máquinas, tais como janelas e menus (VAN DAM, 2000), Interfaces Gráficas Avançadas (Realidade Virtual, Multimídia e Visualização de Informação), Interface de Voz, Interfaces de Caneta, Gestos e Toque, Interfaces Móveis, Interface de Realidade Aumentada, Interface Multimodal e as Interfaces Tangíveis; cada uma objetiva, à sua maneira, que os usuários possam visualizar e interagir com as máquinas de formas mais próximas da forma que interagem com o mundo real (ROGERS; SHARP; PREECE, 2006).

As interfaces tangíveis caracterizam-se por utilizar os meios computacionais para realizar a integração entre objetos reais e objetos virtuais para que seja possível a utilização de objetos físicos para controlar ou representar uma informação virtual (ULLMER; ISHII, 2000). Uma categoria importante destas interfaces são as interfaces tangíveis de mesa (*tabletop*). Elas constituem-se no foco deste trabalho e

forneem interações tangíveis por meio de gestos multitoques (SHAER e HORNECKER, 2009). Pelo fato de apresentar-se na forma de uma mesa, mais de um usuário pode interagir com a aplicação simultaneamente oferecendo potencial de colaboração.

A área de Interação Homem-Computador (IHC), surgida nos anos 1980, estuda as questões de projeto de interface, visando um uso fácil e eficiente dos computadores por meio do design. O processo de design de interfaces é um dos tópicos estudados pela IHC que é composto por fases que vão da análise de tarefas, passando pela especificação e chegando à avaliação formal (MARCHIONINI; SIBERT, 1991). Segundo CARR (1994), a especificação de uma interface tem o objetivo de auxiliar no design dos softwares de forma a aprimorar e enriquecer a comunicação entre o usuário e o computador. Para guiar o processo de design de interfaces, a área de IHC desenvolveu modelos que auxiliam na especificação das diversas modalidades de interação, inclusive a de interfaces tangíveis (SHAER e JACOB, 2009).

Diferentes modelos foram criados para melhor apoiar o desenvolvimento de interfaces, tais como as gramáticas TAG (PAYNE e GREEN, 1986), os diagramas de estados (HAREL, 1987) e o InTML (FIGUEROA, GREEN e HOOVER, 2002). Para as interfaces tangíveis há modelos específicos para facilitar a especificação e implementação das interfaces. Exemplos desses modelos incluem o paradigma TAC (CALVILLO-GÁMEZ et al., 2003) e o TUIML (SHAER e JACOB, 2009).

Embora esses modelos existentes sirvam ao propósito de especificar as interfaces tangíveis, eles possuem limitações. A especificação de interfaces tangíveis carece de um modelo de especificação que atenda as tarefas e ações do usuário no contexto específico das interfaces *tabletop* e as exiba de modo compreensível depois de especificadas (BOTEGA et al., 2011). Além disso, há também a necessidade de simplificação da organização dos dados especificados, já que são importantes componentes da documentação de projetos e podem ser utilizadas como base para a codificação das interfaces. Com os modelos atuais, as interfaces tangíveis são tipicamente especificadas com base apenas em suas interações tangíveis e de maneira que possam oferecer dificuldades para que a especificação seja compreendida (como diversos diagramas fragmentados e arquivos XML). Sendo assim, há limitações tanto de organização da especificação quanto do suporte às características das interfaces tangíveis *tabletop*. Por fim, as

interações multitoques não são especificadas de modo compreensível nos modelos atuais, devido à preocupação depositada na especificação das interações tangíveis. Portanto, um modelo que supra essas limitações se faz necessário.

Com a demanda de desenvolvimento de um número crescente de aplicações em diferentes domínios, foi necessário pesquisar métodos de especificação de interfaces para *tabletops* que permitissem uma visão mais holística da interface. Ou seja, métodos que oferecessem facilidades não só para a especificação das interações tangíveis, suportados por métodos atuais, como o TUIML (SHAER e JACOB, 2009), mas também para especificação ágil de interfaces, além de simplificar a forma que a especificação é organizada e fornecer suporte para a especificação de gestos e documentação

Mais ainda, aplicações que integram veículos autônomos (como VANTs e robôs) em diferentes domínios de aplicação estão sendo construídas com visualização e interação por meio de interfaces tangíveis em dispositivos *tabletop* como parte do projeto do Instituto Nacional de Ciência e Tecnologia em Sistemas Embarcados Críticos (INCT-SEC) Grupo GT4. As características deste tipo de interface favorecem a colaboração, um requisito importante nas classes de aplicações sendo desenvolvidas. Exemplos de aplicações em desenvolvimento no INCT-SEC incluem monitoramento de segurança de infraestruturas críticas (hidrelétricas), controle de voo interativo, suporte à gestão da segurança pública, gerenciamento de emergência e monitoramento ambiental, dentre outras.

A falta de métodos que fornecessem esta visão ampla para a especificação ágil e facilitada de diferentes e complexas classes de interfaces foi o agente motivador deste trabalho. Assim, o objetivo geral deste projeto é criar um modelo de especificação de interfaces tangíveis para *tabletop* juntamente com ferramenta de apoio.

1.1 Motivação

Atualmente, no cenário do desenvolvimento de aplicações para interfaces tangíveis *tabletop*, quando uma aplicação é desenvolvida, é possível realizar, utilizando os métodos existentes, a especificação das interações em relação aos

objetos tangíveis da interface. Entretanto, os documentos de especificação gerados (diagramas complexos em diferentes camadas) são de difícil compreensão fazendo com que a implementação da interface se torne mais trabalhosa caso o desenvolvedor utilize os documentos gerados como guia. Além disso, os aspectos de interação por meio de gestos, que são característicos das interfaces tangíveis *tabletop*, não são suportados pelos modelos de especificação existentes.

Assim, foi desenvolvido um novo modelo de especificação de elementos de interfaces tangíveis *tabletop*, chamado *Tabletop Tangible User Interface Specification Model* (TTUI-SM) que tem como objetivo simplificar a forma de organização dos dados especificados além de prover suporte a gestos avançados e customizados de forma a explorar todo o potencial multitoques oferecido pelas interfaces tangíveis *tabletop*.

Visando agilizar o processo de especificação com o modelo TTUI-SM, uma ferramenta diagramática de especificação foi desenvolvida oferecendo facilidade e agilidade de especificação de elementos de interface e automatização para geração de documentação e definições dos dados especificados.

1.2 Objetivos e Metas

Facilitar o processo de desenvolvimento das interfaces tangíveis de mesa para diferentes domínios de aplicações, simplificando a organização dos dados e suportando o potencial multitoque deste tipo de interface. Para isso, este trabalho foi guiado pelas seguintes metas:

- Desenvolvimento de um modelo de especificação para as interfaces tangíveis *tabletop* que viabilize:
 - Melhor compreensão da especificação dos elementos de interface e de suas interações;
 - Especificação de interações avançadas com gestos e possibilidade de personalização desses gestos.
- Desenvolvimento de uma ferramenta de especificação de interfaces tangíveis *tabletop* com as seguintes características:

- Maior agilidade e automatização do processo de especificação e desenvolvimento de interfaces tangíveis *tabletop*;
- Prover uma visão holística da interface com a visualização dos elementos de interface, formas de interação, relação entre as entidades, geração de documentação e definições XML para agilizar o processo de implementação de interfaces tangíveis *tabletop*.
- Avaliação do modelo TTUI-SM e da ferramenta de especificação TTUI-SMT por meio de dois estudos de caso (voo interativo e proteção de infraestrutura crítica).

1.3 Metodologia

Para a elaboração deste trabalho foi feito um estudo e levantamento de requisitos de duas aplicações sendo desenvolvidas pelo laboratório WINDIS (laboratório no qual foi desenvolvido este trabalho): voo interativo e segurança de infraestruturas críticas.

Deste modo, realizou-se uma pesquisa sobre o estado da arte de interfaces tangíveis e de suas modalidades, seguida de avaliação crítica dos métodos atuais de especificação de interfaces tangíveis *tabletop*, em que foram identificadas as limitações dos modelos atuais para o suporte à especificação das interfaces das aplicações sendo desenvolvidas.

Com base nessa pesquisa, o modelo TTUI-SM foi elaborado para superar as limitações encontradas nos modelos atuais. Para facilitar a estruturação da interface, um modelo abstrato de um elemento de interface foi construído por meio de um mapa mental que define a arquitetura de um elemento de interface e sua hierarquia.

A partir do modelo mental, um modelo concreto que fosse de melhor compreensão e que organiza a especificação de forma simples foi construído na forma de uma ficha de elemento de interface. Esta ficha é organizada na forma de tabelas, com separação de componentes de elementos de interface que facilitam a visualização pelo projetista da interface e também pelo desenvolvedor da aplicação.

Com o modelo abstrato de interface e a ficha de elemento de interface definidos, identificou-se a necessidade de uma forma ágil para a realização da especificação utilizando o modelo TTUI-SM. Assim, foi desenvolvida ferramenta de especificação TTUI-SMT que se baseia o modelo TTUI-SM. Além da especificação, as funcionalidades de gerar automaticamente as fichas de elemento de interface e também um arquivo XML com as definições dos elementos de interfaces foram adicionadas à ferramenta.

Com o modelo TTUI-SM e a ferramenta TTUI-SMT prontos, dois estudos de caso foram parcialmente especificados a fim de validar tanto o modelo TTUI-SM quanto a ferramenta TTUI-SMT. Tais estudos de caso são: **Voo Interativo** e **Vigilância e Segurança de Infraestruturas Críticas**.

Por fim, para avaliar o modelo TTUI-SM e a ferramenta TTUI-SMT, um experimento foi aplicado para diversos estudantes de diferentes níveis acadêmicos entre Iniciação Científica e Doutorandos, consistindo em: apresentação do modelo e especificação de duas aplicações utilizando o; um questionário que foi preenchido pelos participantes após o experimento; por meio da escala de Likert, presente no questionário; e da observação. Os resultados foram analisados e discutidos de forma a concluir que os objetivos do trabalho foram atingidos.

1.4 Organização

No Capítulo 2, Interfaces Computacionais, conceitos gerais sobre interfaces convencionais serão apresentados, citando as primeiras interfaces criadas. Em seguida, serão apresentadas e conceitualizadas algumas das principais interfaces Pós-WIMP: Interfaces Gráficas Avançadas (Realidade Virtual, Multimídia e Visualização de Informação), Interfaces de Voz, Fala, Gestos, Escrita e Toques, Interfaces Móveis, Interface de Realidade Aumentada, Interface Multimodal, e as Interfaces Tangíveis.

No Capítulo 3, Desenvolvimento de Interfaces Tangíveis *Tabletop*, os aspectos arquitetônicos e as ferramentas que fazem parte do desenvolvimento de aplicações para interfaces tangíveis *tabletop* (Protocolo, *Trackers* e *Application Frameworks*) serão conceitualizados e explicados.

No Capítulo 4, Trabalhos Relacionados, serão apresentados às diversas abordagens existentes para a especificação de diversas modalidades de interface.

No Capítulo 5, Modelo de Especificação de Interfaces TTUI-SM, o modelo de especificação será apresentado e explicado. Em seguida, será apresentada a ferramenta desenvolvida para a especificação diagramática do modelo.

No Capítulo 6, Avaliação do Modelo TTUI-SM, dois estudos de caso serão apresentados a fim de demonstrar a utilização do modelo de especificação utilizando a ferramenta de especificação TTUI-SMT. Em seguida, será apresentado o experimento realizado com diversos estudantes, bem como o questionário e as atividades aplicadas. É também apresentado o resultado da experimentação por meio da análise do questionário e das observações durante o experimento.

No Capítulo 7, Conclusões, as conclusões sobre o trabalho são apresentadas juntamente com as limitações identificadas e os trabalhos futuros.

Capítulo 2

INTERFACES COMPUTACIONAIS PÓS WIMP

Segundo KIRNER e SISCOOTTO (2007), desde antes do surgimento das primeiras máquinas conhecidas por utilizarem microchips e microprocessadores (computadores, notebooks, micro-ondas, televisões, entre outros), as pessoas interagem em seu cotidiano por meio de interfaces naturais, ou seja, utilizam seus próprios sentidos para que haja a interação, como por exemplo, uma conversa. Com o surgimento das máquinas, muitas tarefas se tornaram relativamente simples, porém, fez-se necessário agregar um novo conhecimento das novas e diferentes interfaces fornecidas pelas máquinas com o passar do tempo.

As primeiras interfaces computacionais surgiram entre as décadas de 40 e 50, e toda a interface na qual o usuário poderia resgatar os dados computacionais era baseada apenas em chaves e lâmpadas. Somente a partir da década de 60, os terminais de vídeo começaram a ser utilizados fazendo com que o termo “Interface Gráfica” tivesse seu início de uso, embora fossem ainda rudimentares (KIRNER;SISCOOTTO,2007). A primeira interface gráfica manipulável que se tem registro foi o Sketchpad (Figura 1), criado por Ivan Sutherland em 1963, no qual, utilizando uma caneta e o desenho de linhas, um computador e uma pessoa podiam se comunicar (SUTHERLAND, 1963).



Figura 1 - Sketchpad (SUTHERLAND, 1963)

Na década de 70, os microcomputadores se popularizaram graças à utilização dos microprocessadores. Esses microcomputadores tinham, inicialmente, suas interfaces baseadas em linha de comando como, por exemplo, o COMMAND do MS-DOS ou o SHELL do UNIX (RADDICHI, 2010).

Em 1977, o conceito de manipulação direta, ou seja, em que é possível selecionar objetos e textos e manipulá-los, foi introduzido por Alan Kay (KAY; GOLDBERG, 1977) e, assim, o primeiro sistema comercial utilizando o conceito de manipulação direta foi o Xerox Star, em 1981 (MYERS, 1998). Tal conceito é empregado no desenvolvimento das interfaces gráficas mais utilizadas neste começo do século XXI: as interfaces WIMP.

As interfaces WIMP surgiram na década de 80 e estão amplamente presentes nas interfaces de aplicações utilizadas no dia-a-dia. Um fator positivo sobre esta interface é a facilidade de aprendizagem, uma vez que memória e reconhecimento de imagens são explorados (VAN DAM, 2000). Além disso, há também a característica de multitarefa no nível de interface, ou seja, é possível, por exemplo, que mais de uma janela esteja processando uma operação simultaneamente. Em contraparte, a naturalidade deste tipo de interface é limitada ao mouse e ao teclado de um computador, o que pode causar certa frustração em usuários que buscam uma interação natural com a interface (VAN DAM, 2000). A Figura 2 exibe a interface gráfica do Xerox Star, uma interface WIMP.

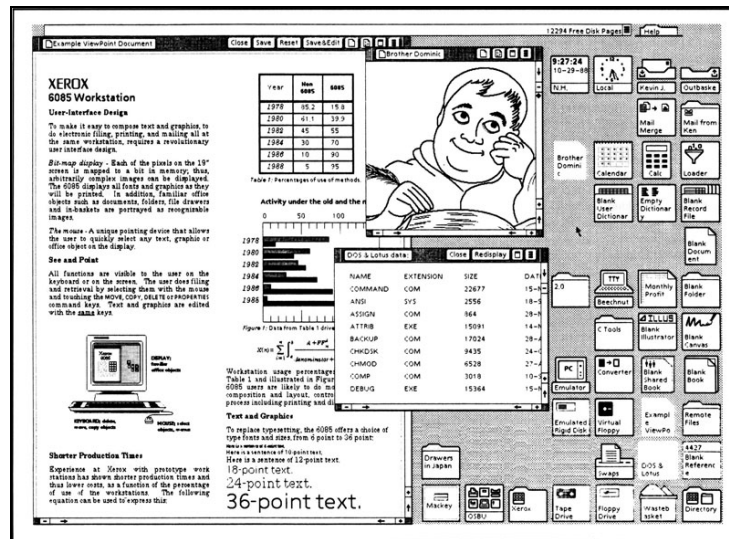


Figura 2 - Interface Gráfica do Xerox Star (JOHNSON et al., 1989)

Embora as interfaces WIMP funcionem de forma satisfatória em *desktops*, não é possível utilizar de forma nativa recursos como fala e gestos, por exemplo. Assim, as interfaces pós WIMP começaram a ser desenvolvidas possibilitando o uso de técnicas de interação cada vez mais naturais, além dos clássicos mouse e teclado.

As interfaces Pós WIMP surgiram a partir da década de 1990 e destacam-se as Interfaces Gráficas Avançadas (Realidade Virtual e Multimídia), Interface de Voz, Fala, Gestos, Escrita e Toques, Interfaces Móveis, Interface de Realidade Aumentada, Interface Multimodal, Interface Robótica, Interface Vestível e a Interface Tangível.

2.1 Interfaces Pós WIMP

Buscando oferecer cada vez mais naturalidade nas interações dos usuários para com os computadores, novas modalidades de interface foram criadas. Nesta seção serão descritas algumas delas.

2.1.1 Interfaces de Realidade Virtual

Embora seja considerada uma interface Pós-WIMP, o conceito de realidade virtual não é algo novo. Ivan Sutherland, em 1965, com o artigo “The Ultimate Display” (SUTHERLAND,1965), introduziu conceitos que são muito utilizados nas pesquisas da área. Esses conceitos envolvem imersão em um mundo simulado, e sensorialmente completo, de entrada e saída (GOBBETTI; SCATENI, 1998). Sutherland estabeleceu o seguinte desafio: “A tela é uma janela pela qual se vê um mundo virtual. O desafio é fazer com que o mundo pareça real, aja real, soe real, sinta real”.

KIRNER e SISCOOTTO (2007) definiram realidade virtual da seguinte maneira:

Realidade virtual é uma interface avançada para aplicações computacionais, que permite ao usuário navegar e interagir, em tempo real, com um ambiente tridimensional gerado por computador, usando dispositivos multissensoriais.

PERRY et al. (1997) dividiu as interfaces de realidade virtual em quatro áreas específicas: Visual, Auditiva, Tátil e Navegação.

2.1.1.1 Visual

As principais preocupações sobre os aspectos virtuais da realidade virtual são sobre detalhe da imagem *versus* velocidade de renderização, sendo que, quanto maiores os detalhes contidos na imagem, maior será o tempo de gasto na renderização da mesma.

Para a visualização da realidade virtual, existem alguns tipos de monitores, que são LCD Flicker Lens, Head Mounted Displays e Binocular Omni-Orientation Monitor (PERRY; SMITH; YANG, 1997).

2.1.1.1.1 LCD Flicker Lens

LCD Flicker Lens (Lentes de tremulação LCD) são como um par de óculos. Um fotosensor é montado nos óculos para ler o sinal do computador. O seu funcionamento é alternado, ou seja, por meio do sinal é decidido se a luz passará pela lente da direita ou pela lente da esquerda. Assim, o usuário terá a visão do olho esquerdo sendo recebida apenas pelo lado esquerdo e a visão do olho direito sendo

recebida apenas pelo olho direito. Este tipo de display é fácil de vestir e remover, uma vez que os óculos são leves e sem cabos. O usuário precisa olhar diretamente para a tela do computador para ver a cena e, logo, não há um efeito imersivo (PERRY; SMITH; YANG, 1997).

2.1.1.1.2 Head Mounted Displays (HMD)

Head Mounted Displays (HMD) (Displays Montados na Cabeça) fazem com que o usuário sempre esteja visualizando o ambiente virtual colocando uma tela na frente de cada um de seus olhos. O dispositivo é comumente chamado de “capacete” e o que o usuário visualizará do ambiente virtual é decidido de acordo com a orientação captada pelos sensores que são embutidos no capacete. Um problema deste tipo de dispositivo é a necessidade de haver um cabo conectando o HMD a um computador, gerando assim um problema de mobilidade. Existem quatro tipos de HMD: *LCD display HMD*, *Projected HMD*, *Small CRT HMD* e *Single Column LED HMD* (PERRY; SMITH; YANG, 1997).

2.1.1.1.3 Binocular Omni-Orientation Monitor (BOOM)

Binocular Omni-Orientation Monitor é um monitor acoplado na ponta de um braço mecânico munido de sensores em suas juntas a fim de captar as movimentações feitas e gerar a cena correspondente. Ao contrário do HMD, o usuário não precisa vestir o dispositivo, mas sim segurar o monitor e colocar seu rosto nele (PERRY; SMITH; YANG, 1997). A Figura 3 exibe um exemplo de BOOM.

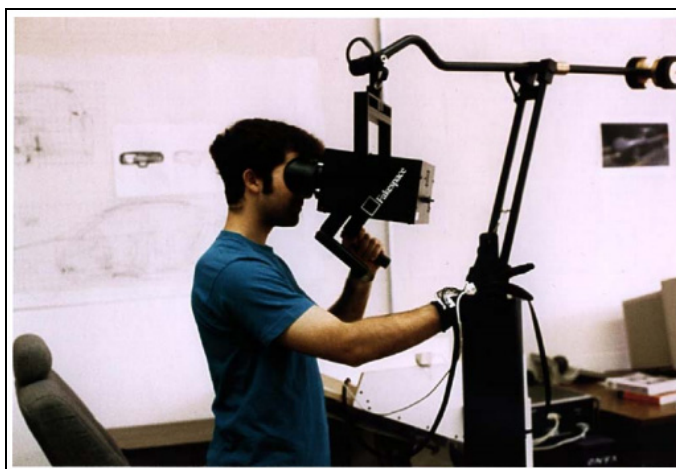


Figura 3 - Binocular Omni-Orientation Monitor (BOOM) (MÁRQUEZ, 2002)

2.1.1.2 Auditiva

Quando se fala em simulação de áudio em ambientes de realidade virtual, o principal foco das pesquisas voltadas para esta área é de trabalhar as melhores maneiras de simular a origem do som e, para isso, utiliza-se o Som 3D. Um grande poder computacional é exigido para simular o áudio de aplicações de Realidade Virtual, uma vez que os sons simulados podem reagir com ruídos, paredes ou outras fontes sonoras. A relatividade entre a posição da origem do som e o ouvinte também precisam ser determinados para, por exemplo, quando a origem do som se encontrar atrás do ouvinte, e o mesmo se virar, a origem do som se mudar para a sua frente e tudo isso ser simulado no áudio (PERRY; SMITH; YANG, 1997).

Todas essas preocupações têm como objetivo proporcionar um som mais realista, fazendo com que o usuário consiga obter uma experiência imersiva também em nível auditivo. BEGAULT (2000) destaca ainda que, ganhando vantagem sobre a visão, em um ambiente de alerta, no qual qualquer manifestação pode ocorrer em um ângulo de 360°, o som 3D utilizado em aplicações de realidade virtual possibilita a experiência de, apenas com o som, o ouvinte definir onde está a origem da manifestação.

2.1.1.3 Tátil

Um ambiente de realidade virtual é repleto de objetos com os quais, possivelmente, um usuário necessitará interagir, manipulando-os de diversas formas. PERRY et. al. (1997) destacam que um grande problema dos ambientes virtuais é justamente a falta de tangibilidade, ou seja, a falta do uso de recursos físicos na manipulação de objetos dos ambientes virtuais. Porém, projetos voltados a resolver esse problema têm mostrado ótimos resultados. Os principais conceitos que possibilitam a tangibilidade em ambientes virtuais são o *Force Feedback* e o *Tactile Feedback*.

Force Feedback controla a reação de objetos virtuais em relação ao usuário como, por exemplo, quando em um ambiente virtual, um usuário é atingido por algum objeto. Neste caso, o *Force Feedback* será o responsável pela simulação da sensação de impacto para o usuário. Dentre os diversos dispositivos de *Force Feedback* algumas categorias podem ser citadas: luvas, exoesqueletos, plataformas de movimentação. Como um exemplo deste tipo de dispositivo, GOMEZ et. al.

(1995) apresentam o dispositivo *Rutgers Master II*, um dispositivo na forma de uma luva que o usuário veste em suas mãos. Este dispositivo é composto de quatro cilindros pneumáticos responsáveis pela movimentação da luva. O protótipo da *Rutgers Master II* é exibido na Figura 4.

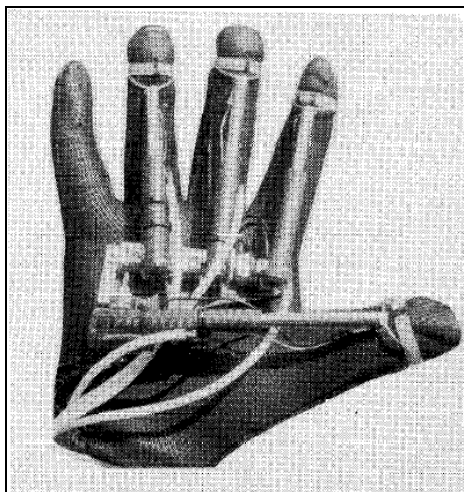


Figura 4 - Protótipo da Rutgers Master II (GOMEZ; BURDEA; LANGRANA, 1995)

Tactile Feedback controla a forma pela qual os objetos causam sensações para o usuário, como por exemplo, a temperatura ou a textura de algum objeto virtual. Este tipo de simulação é um grande desafio, uma vez que os seres humanos possuem vários nervos que são responsáveis por vários setores, por exemplo, sentir o calor, sentir a pressão e sentir forças. Portanto, é necessária grande atenção para todos estes detalhes quando se pretende, utilizando um dispositivo *Tactile Feedback*, simular tais sensações.

2.1.1.4 Navegação

Para que um ambiente virtual possa gerar uma boa experiência de imersão para o usuário, é essencial que a posição da pessoa imersa no ambiente seja sempre verificada, assim como sua orientação no espaço. Assim, posicionamento nos eixos **x**, **y** e **z**, além de passos e movimentações em geral deverão ser monitorados constantemente. Para tal função, existem os seguintes tipos de rastreadores de posicionamento e movimentação: Rastreadores Mecânicos, Eletromagnéticos, Ultrassônicos, Infravermelhos e Inerciais (PERRY et. al., 1997).

Por meio das interfaces de realidade virtual e aplicando os conceitos e dispositivos citados, é possível fazer a imersão de um indivíduo em um mundo virtual. Porém, há também situações nas quais o usuário deseja trazer o mundo

virtual para o mundo real. A realidade aumentada possibilita que tal necessidade se torne real.

2.1.2 Interfaces Multimídia

Interfaces multimídia, como o nome sugere, são interfaces que vão além das interfaces gráficas de usuário fazendo uso de diferentes mídias para enriquecer a experiência e interatividade do usuário. Com a combinação de sons, vídeos, animações e textos essas interfaces permitem que a informação seja apresentada de forma a melhor atender a natureza da informação que está sendo fornecida aos usuários.

Ao apresentar a informação em diversas formas e formatos acredita-se que “mais é melhor” e que a “soma das partes é melhor que o todo” (LOPUCK, 1996). Ao interagirmos com diversas mídias percebemos que é mais fácil de aprender, temos uma melhor compreensão da informação, prestamos mais atenção e aumentamos nosso prazer (SCAIFE; ROGERS, 1996).

Apenas com o poder computacional recente a capacidade de manipular diferentes mídias foi alcançada pelos computadores de mesa. Um exemplo de uso da interface multimídia surgiu com o nascimento do primeiro navegador gráfico para a internet, o Mosaic em 1993 e, após duas décadas, os navegadores ainda são baseados neste navegador, mantendo muitas das características principais.

SAUER e ENGELS (1999) exemplificam uma aplicação de interface multimídia chamada “Enciclopédia de Compositores Clássicos” em que cada objeto visível na aplicação está associado a objetos de mídia. Assim, é possível visualizar imagens, pedaços de músicas e pedaços de vídeos. A aplicação é exibida na Figura 5.

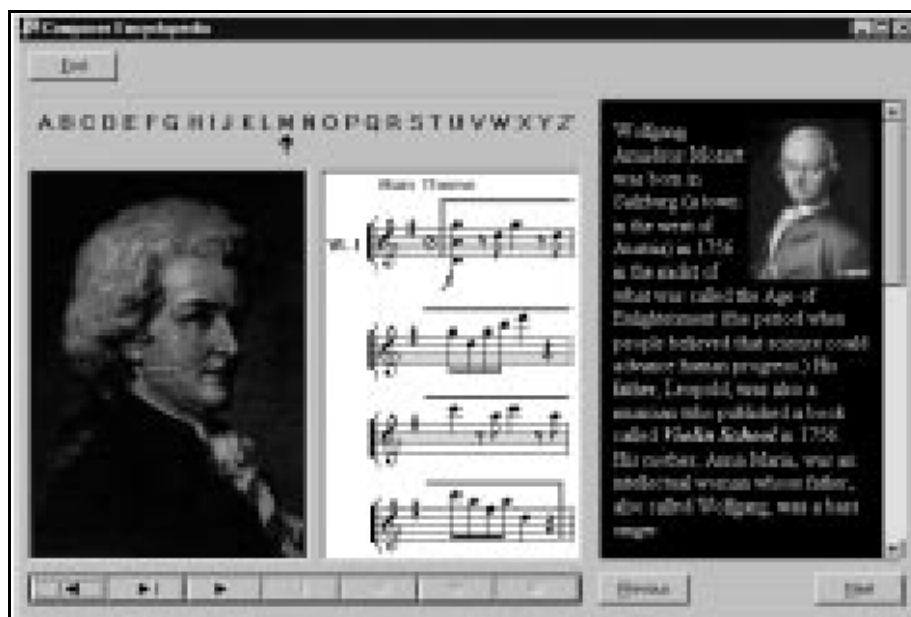


Figura 5 - Enciclopédia de Compositores Clássicos (SAUER; ENGELS, 1999)

2.2 Interfaces de Voz, Fala, Gestos, Escrita e Toques

Acompanhando a evolução do poder de processamento dos computadores e procurando tornar cada vez mais naturais as interações dos usuários com as máquinas, as interfaces de voz e fala, gestos e escrita foram desenvolvidas.

2.2.1 Interfaces de Voz e Fala

Segundo COHEN e GIANGOLA (2004) uma Interface de Voz, também conhecida por *Voice User Interface* (VUI), possibilita que o usuário “converse” com os computadores e diga a eles quais tarefas deverão ser realizadas pelo sistema. De via inversa, o sistema também pode responder sintetizando palavras pré-programadas pelo sistema. Esse tipo de interface tem sendo utilizado em “*Call Centers*” substituindo atendentes no primeiro atendimento de um usuário quando ele realiza uma ligação para algum serviço de informação.

As implementações mais comuns de VUI apenas registram o som de comandos de forma isolada. Quando o usuário pronuncia um comando para o sistema, é realizada uma busca pelos comandos pré-programados e, com o uso de

técnicas de processamento de sinais e reconhecimento de padrões, o comando que obtiver maior probabilidade de acerto é executado.

Em sua versão mais simples, amplamente utilizada em telefones celular, o próprio usuário deve gravar os comandos a serem pronunciados posteriormente. Já em interfaces mais sofisticadas, o sistema tenta reconhecer não apenas os comandos isoladamente, mas também a semântica em determinada sequência de palavras pronunciadas.

HUA e WHENG (2010) propuseram o design de uma interface de voz e fala para o domínio de sistemas automobilísticos. O produto, chamado “*the Wireless Media Gateway Generation*”, se conecta ao celular do usuário via Bluetooth permite que o usuário solicite ao sistema que realize algumas tarefas como realizar chamadas, tocar músicas, realizar busca de contatos ou até mesmo solicitar que suas mensagens de texto sejam lidas.

2.2.2 Interfaces de Gestos, Escrita e Toque

Além da naturalidade das interações oferecidas pelas interfaces de voz e fala, os seres humanos também utilizam escritas, gestos e toques para interagir e, desta forma, interfaces que recebem estes meios de entrada foram criadas.

Segundo ROGERS, SHARP e PREECE (2006), estes tipos de interface possibilitam que os usuários escrevam, desenhem, selecionem e movam objetos por meio do uso de canetas de luz (ou canetas stylus), gestos ou toques. Desta forma, as interações são totalmente baseadas nas ações que geralmente são aprendidas durante a infância.

Para a captação destas interações, câmeras são utilizadas a fim de visualizar as imagens e, utilizando técnicas de visão computacional, as interações são interpretadas e reconhecidas.

As interfaces de escrita, também conhecidas como *Pen-Based* (baseadas em canetas) são comumente usadas em PDAs (*Personal Digital Assistant*) e também em interfaces que possuem grandes telas. Para a interação são utilizadas canetas que permitem ao usuário escrever, desenhar ou realizar comandos de acordo com a forma de utilizar a caneta sobre a interface.

As interfaces de toque recebem como meio de entrada os toques das pontas dos dedos dos usuários. Com a evolução do reconhecimento de toques, as

interfaces multitoques surgiram de forma a suportar toques simultâneos na superfície da interface. Segundo NACENTA, BAUDISCH e BENKO (2009), é possível aplicar diversas transformações espaciais em objetos virtuais (como transladar, rotacionar e ampliar uma foto, por exemplo) utilizando apenas dois dedos na superfície. A Figura 6 exibe um exemplo de interface de toque.



Figura 6 - Interface de Toque (NACENTA; BAUDISCH; BENKO, 2009)

As interfaces de gestos têm diversos sistemas desenvolvidos e em desenvolvimento ainda. As interações são detectadas por meio do uso de visão computacional e aspectos como o posicionamento das mãos e o movimento dos braços são reconhecidos (NACENTA; BAUDISCH; BENKO, 2009). Diversos domínios de aplicação são beneficiados com esta modalidade fazendo com que novos rumos, em termos de usabilidade, sejam tomados.

ZAFRULLA, BRASHEAR e STARNER (2011) fazem o uso do Microsoft Kinect, um dispositivo capaz de captar e reconhecer gestos do usuário (MICROSOFT, 2012), para o reconhecimento da linguagem de sinais norte americana. A Figura 7 exibe o uso do Microsoft Kinect para a detecção da linguagem de sinais.

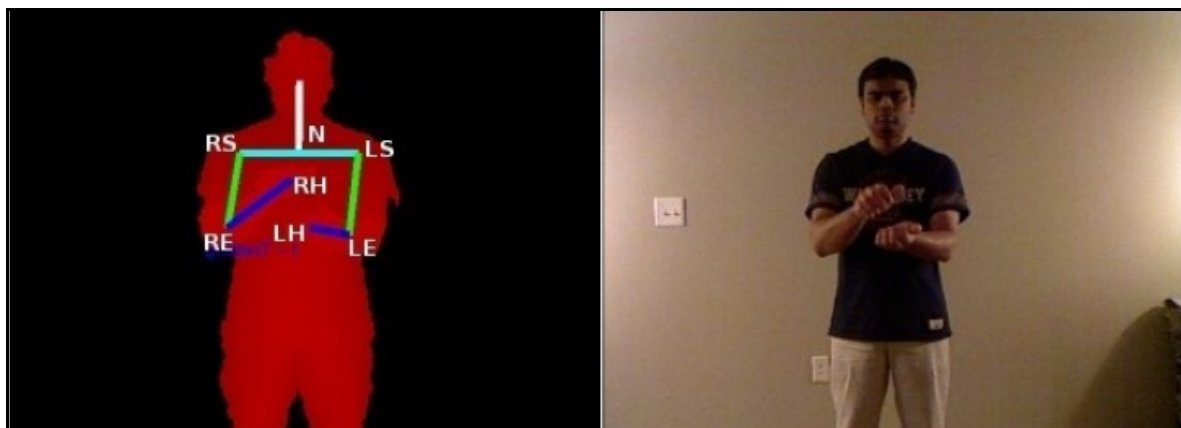


Figura 7 - Microsoft Kinect sendo usado para reconhecer linguagem de sinais (ZAFRULLA; BRASHEAR; STARNER, 2011)

2.3 Interfaces Móveis

As interfaces móveis foram designadas para aparelhos portáteis que possam ser utilizados sem a necessidade do usuário parar de se movimentar. Celulares, PDAs e *tablets* são exemplos destes aparelhos (ROGERS; SHARP; PREECE, 2006). No ano de 2008, cerca de quatro bilhões de pessoas eram usuários de telefones celulares, fazendo com que estes dispositivos se tornassem uma potencial plataforma para o desenvolvimento de diversas aplicações de domínios diferentes como, por exemplo, educação, saúde, finanças e agricultura.

Aparelhos celulares que antes eram feitos apenas para discar números e realizar ou receber chamadas, agora têm uma grande flexibilidade de uso devido aos novos recursos que possibilitam às interfaces móveis fazerem uso de meios de entrada como fala livre ou estruturada, digitação, escrita, e meios de saída como texto, áudio e animações com áudio. Com esses meios de entrada e saída, as interfaces móveis podem oferecer recursos ricos, como diálogo entre o dispositivo e usuário, SMS, interface gráfica, entre outros. (MEDHI et al., 2011).

Com o passar do tempo, os aparelhos celulares passaram a vir com câmeras de vídeo que, inicialmente, eram usadas apenas para tirar foto, mas passaram a também gravar vídeos, inclusive em alta definição, e realizar transmissão ao vivo de filmagens. KIM et al. (2010) desenvolveram uma aplicação de interface móvel que auxilia os usuários com suas dietas utilizando imagens captadas pela câmera do celular. Como é exibido na Figura 8, o usuário tira fotos de suas refeições e as envia

para o servidor que faz a identificação dos alimentos por meio de reconhecimento de imagens e estima as calorias das refeições submetidas pelo usuário.



Figura 8 - Interface Móvel para auxílio de dietas (KIM et al., 2010)

2.4 Interfaces de Realidade Aumentada

Quando se pretende estender os elementos do mundo real com a utilização de objetos de um mundo virtual, as interfaces de realidade aumentada têm seu papel justificado. AZUMA et. al. (2001) explicam que, a partir da coexistência de mundo real e mundo virtual, a realidade aumentada tem como objetivo aprimorar a percepção do usuário de forma a permitir a interação com objetos virtuais sobre o plano do mundo real. Assim, a realidade aumentada pode ser definida da seguinte forma:

O termo Realidade Aumentada refere-se à integração de elementos virtuais em uma visão do ambiente do mundo real. Isso é obtido adicionando-se informações virtuais ou sintéticas geradas em um vídeo ao vivo do mundo real capturado por câmeras. Enquanto o aumento está sendo feito, a informação sintética será mostrada em tempo real em dispositivos de display de RA como monitores, telas

de projeção ou head mounted (CHEN; MACDONALD; WUNSCHE, 2008).

A Figura 9 exibe um exemplo de realidade aumentada em que um carro pequeno e um abajur virtual são posicionados no mundo real.



Figura 9 - Realidade Aumentada aplicada a uma mesa (KIRNER; SISCOOTTO, 2007)

Três características devem compor um sistema de realidade aumentada: misturar real e virtual em um ambiente real, ou seja, possibilitar que elementos virtuais possam ser incorporados à realidade; ser interativo em tempo real, fazendo com que a interação com objetos virtuais e reais seja a mais natural possível; e ser registrado em 3D, realizando um preciso posicionamento dos objetos virtuais no mundo real, dando a ilusão de que o objeto de fato faz parte do ambiente.

Uma interface de realidade aumentada depende de dois importantes componentes: dispositivos de visualização e técnicas de rastreamento.

2.4.1 Dispositivos de Visualização

Para que seja possível a visualização da integração dos elementos virtuais juntamente com elementos reais no mundo real é necessário um meio pelo qual os objetos virtuais sejam visualizados. Para isso, existem dispositivos por meio dos quais é possível visualizar uma interface de realidade aumentada. Dentre eles, três tipos de dispositivos se destacam: *Handhelds*, *See-throughs* e displays baseados em projeção.

Os ***Handhelds*** (considerando também Celulares, ***Smartphones*** e PDAs), por sua mobilidade e por não haver a necessidade de “vesti-lo”, se tornaram uma alternativa totalmente viável na visualização de interfaces de realidade aumentada

(ZHOU et. al., 2008). A primeira interface de realidade aumentada feita para celulares comuns (Figura 10) foi apresentada por MOHRING et. al. (2004), com integração em tempo real de gráficos 2D e 3D nas imagens captadas pela câmera do aparelho celular.



Figura 10 - Celular com interface de realidade aumentada (MOHRING et. al., 2004)

Os dispositivos **See-Through**, considerados displays acoplados à cabeça ou *Head Mounted Display* (HMD), permitem ao usuário olhar para o mundo real e enxergar os objetos virtuais sobre ele. Existem dois tipos de dispositivos *see-through*: óticos e de vídeo.

Um dispositivo *see-through* ótico ou *Optical See-Through* (OST), exibido na Figura 11a, é um display contendo uma lente (normalmente em um tom fraco da cor prata) ou tecnologia parecida pela qual o usuário consegue visualizar o mundo real juntamente com os objetos virtuais que são projetados na lente, sobrepondo a visão dos objetos do mundo real.

Um dispositivo *see-through* de vídeo ou *Video See-Through* (VST) como exibido na Figura 11b é um display onde um vídeo da visão do usuário é exibido em tempo real, ou seja, o usuário não estará de fato visualizando o ambiente, mas sim um vídeo, em tempo real, com a inserção dos objetos virtuais no ambiente (ZHOU et. al., 2008).

Em um comparativo, os dispositivos OST tem em sua vantagem o fato de apenas projetar os objetos virtuais em sua lente e assim o usuário continuará visualizando o mundo real, porém, é possível um melhor tratamento da imagem utilizando dispositivos VST (ROLLAND; FUCHS, 2000).

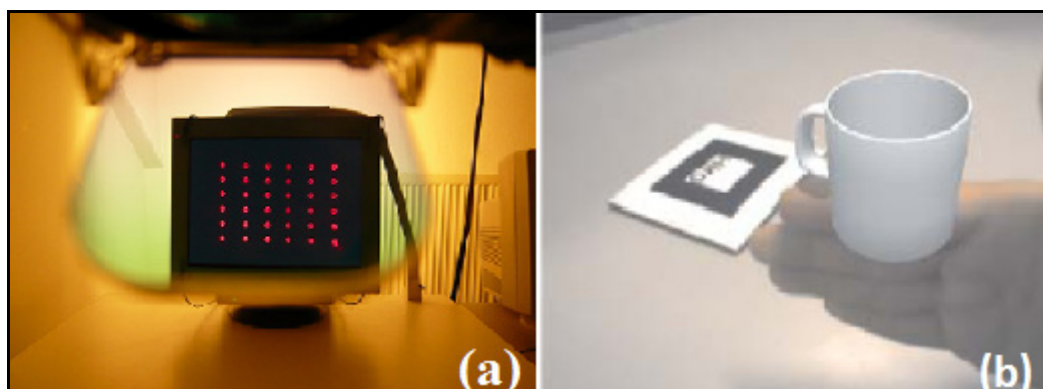


Figura 11 – (a) *Optical See-Through* (Adaptado de HUCKAUF et. al., 2010) (b) *Video See-Through* (Adaptado de HAYASHI; KATO; NISHIDA, 2005)

Os **Displays Baseados em Projeção** são dispositivos que não necessitam que o usuário vista ou tenha em mãos qualquer equipamento para que possa visualizar a realidade aumentada (ZHOU et. al., 2008). Este tipo de visualização utiliza um projetor em que, a partir de sua projeção, todos visualizarão os objetos virtuais em um único ponto comum. A Figura 12 exibe um exemplo de display baseado em projeção onde elementos virtuais são projetados de forma a auxiliar um usuário a fazer um furo na parede, evitando danificar fiações (EHNES et. al., 2004).

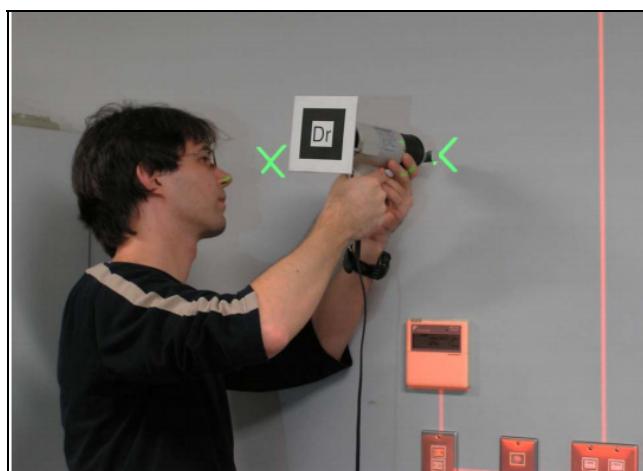


Figura 12 – Realidade Aumentada com Display Baseado em Projeção (EHNES et. al., 2004)

2.4.2 Rastreamento

Para que o elemento virtual seja posicionado em um local definido, é necessário, por parte da aplicação, um conhecimento da posição do usuário, para onde ele está olhando, o que está no ambiente. As técnicas de rastreamento são de grande importância para uma interface de realidade aumentada, pois, a partir de tais

técnicas, é possível definir exatamente onde ocorrerá o aumento da realidade, bem como a forma que interações serão tratadas.

KLINKER et. al. (2000) definiram uma representação gráfica acerca de como é o funcionamento da realidade aumentada entre três pontos: o mundo real, o usuário e o computador. A partir de dados do mundo real, rastreados utilizando técnicas de rastreamento, o computador realiza o processamento de acordo com os comandos fornecidos pelo usuário e executa o aumento da realidade de forma a possibilitar a interação do usuário. A Figura 13 ilustra o ciclo de uma interface de realidade aumentada.

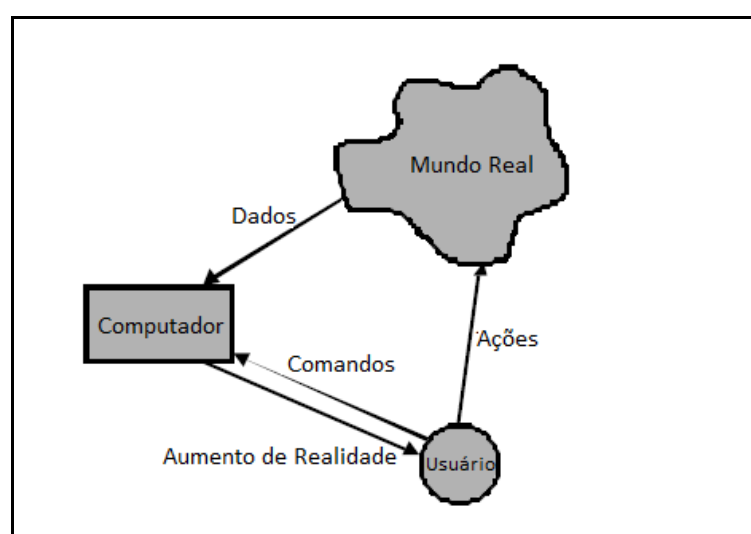


Figura 13 – Ciclo de uma interface de realidade aumentada (Adaptado de KLINKER et. al., 2000)

ZHOU et. al. (2008) definem três tipos de técnicas de rastreamento: Baseada em Sensores, Baseada em Visão e Híbrida.

O rastreamento **Baseado em Sensores** utiliza sensores como, por exemplo, os magnéticos, óticos, inerciais e mecânicos. Estes sensores são físicos, incapazes de reconhecer imagens e foram os primeiros sensores a serem usados. NEWMAN et. al. (2001), utilizaram rastreamento baseado em sensor inercial juntamente com dispositivos sem fios chamados Bats (utilizando-se de pulsos ultrassônicos para calcular a distância entre o usuário e receptores em pontos fixos) em uma aplicação de realidade aumentada. Ambos os dispositivos, sensor e Bats, foram ligados a um HMD e exibiam para os usuários (Figura 14a), por meio da lente do HMD, os dados de pessoas (nome), computadores (*hostname*) e telefone (ramal). A Figura 14b ilustra a arquitetura da aplicação.

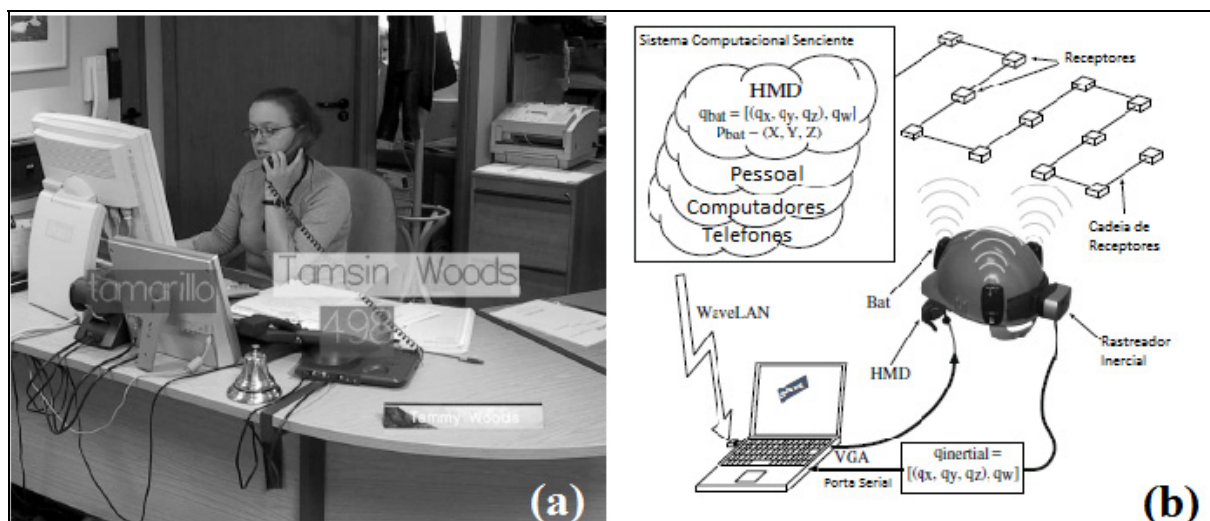


Figura 14 – Aplicação de Realidade Aumentada utilizando rastreamento por sensor. (a) Visualização do Usuário (b) Arquitetura da Aplicação. (Adaptado de NEWMAN et. al., 2001)

O rastreamento **Baseado em Visão** utiliza técnicas de processamento de imagens para fornecer os dados para a aplicação de realidade aumentada. A partir de uma câmera, elementos captados são processados utilizando algoritmos de visão computacional e, de acordo com os objetivos da aplicação, os elementos virtuais são inseridos no ambiente e exibidos para o usuário. Em geral, marcadores são amplamente utilizados para este tipo de rastreamento.

NAIMARK e FOXLIN (2002) apresentam a terceira geração de marcadores fiduciais e apresentam um comparativo entre as duas primeiras gerações desses marcadores (Figura 15). Os marcadores fiduciais são utilizados de forma a serem detectados por meio de visão computacional e, a partir de seu ponto, inserir um elemento virtual.



Figura 15 – Três gerações de marcadores fiduciais (NAIMARK; FOXLIN, 2002)

Além da utilização de marcadores fiduciais para o rastreamento, é possível também a utilização de rastreamento de características naturais de um ambiente, ou

seja, sem marcadores. Para o rastreamento sem marcadores (*Markerless Tracking*), uma técnica baseada em modelo (*Model-Based Tracking*) é aplicada de forma que todo o rastreamento se baseará em modelos pré-definidos. REITMAYR e DRUMMOND (2006) utilizaram modelos para o rastreamento de bordas de casas e pequenos prédios (Figura 16) e destacaram que, para pequenas aplicações, a utilização de modelos CAD para as bordas, tal abordagem funciona bem.

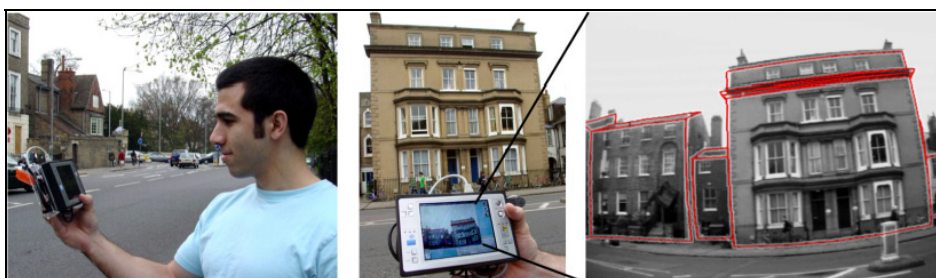


Figura 16 – Utilização de rastreamento sem marcadores (*Markerless*) para detecção de bordas de casas (REITMAYR; DRUMMOND, 2006)

O **Rastreamento Híbrido** visa unir as técnicas baseadas em sensores juntamente com as técnicas baseadas em visão e dessa forma resolver problemas de ambas as técnicas. Assim, é possível, por exemplo, a utilização de visão computacional para posicionar um elemento virtual no ambiente e utilizar um sensor inercial para, de acordo com a movimentação do usuário (como uma aproximação), realizar transformações nos elementos virtuais.

2.5 Interfaces Multimodais

Na busca por aprimoramento na naturalidade da interação do homem com a máquina, as pesquisas caminharam de forma a explorar recursos cognitivos do homem. As interfaces multimodais são construídas não apenas sobre um tipo de interação, mas a partir de vários, como fala, olhar, movimentação da cabeça, movimentação do corpo, entre outros. OVIATT (2003) definiu as interfaces multimodais da seguinte forma:

Interfaces Multimodais processam dois ou mais meios de entrada (como fala, caneta, toque, gestos manuais, olhar e movimentos de cabeça e corpo) de uma maneira coordenada com o sistema de

saída multimídia. Eles são a nova classe de interfaces que tem a intenção de reconhecer ocorrências de formas naturais da linguagem humana e seu comportamento, e que incorporam uma ou mais tecnologias baseadas em reconhecimento (ex: fala, caneta, visão).

Por meio da naturalidade de interação, as interfaces multimodais traçam dois objetivos: dar suporte a usuários que possuem problemas de percepção ou comunicação e integrar habilidades computacionais no mundo real de acordo com as maneiras naturais de interação do homem (DUMAS et. al., 2009). Assim, a linguagem que o usuário pode utilizar para interagir com a interface é simplificada de forma que, para criar um círculo azul na tela, é necessário apenas apontar para tela e dizer “Criar um círculo azul”.

2.5.1 Arquitetura

A arquitetura das interfaces multimodais pode ser dividida em partes. Por receberem diversos meios de entrada que podem ser processados de maneiras totalmente diferentes e independentes, além da possibilidade de necessitarem enviar diversas formas de resposta a partir de cada meio de interação, as interfaces multimodais precisam de uma arquitetura complexa para ambos os processamentos. Assim, a Figura 17 ilustra o fluxo de uma arquitetura de interface multimodal.

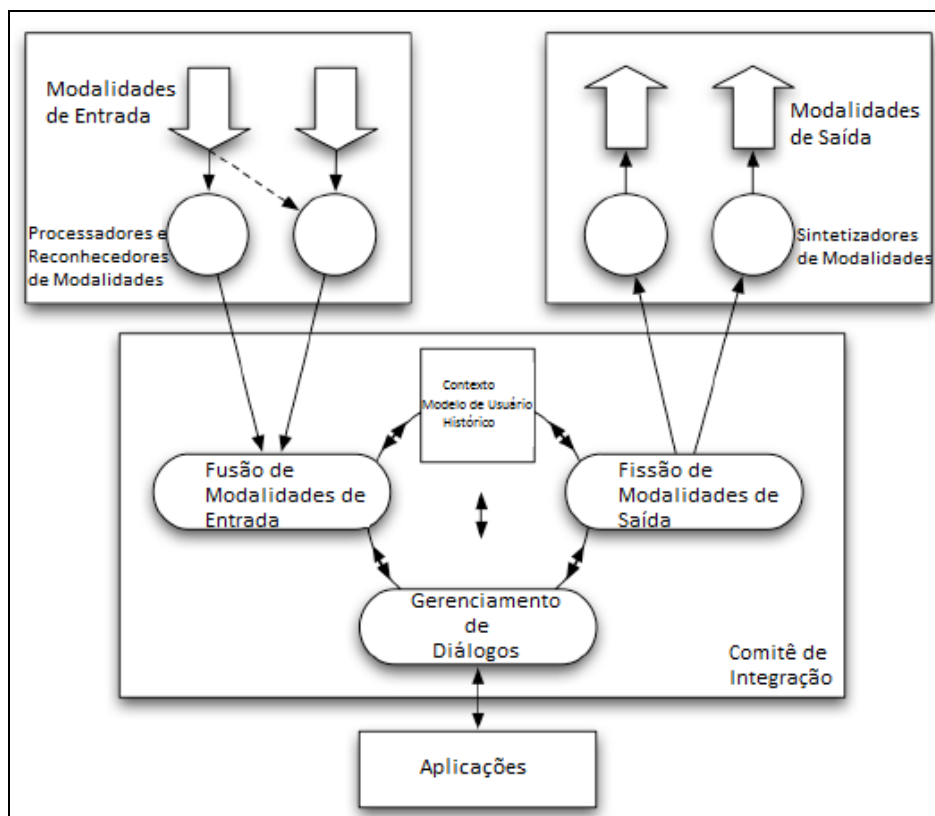


Figura 17 - Arquitetura de uma Interface Multimodal (Adaptado de DUMAS; LALANNE; OVIATT, 2009)

De acordo com a arquitetura ilustrada na Figura 17, o fluxo segue de forma que a aplicação envia os dados para o “Comitê de Integração”, no qual o componente “Gerenciamento de Diálogos” encaminha as entradas para a “Fusão de Modalidades de Entrada” que, utilizando reconhecedores, processa as entradas. Uma vez processadas, haverá o processamento da saída por meio do “Fissão de Modalidades de Saída”, que utiliza sintetizadores de modalidades para definir quais saídas pertencem a quais modalidades e novamente o “Gerenciamento de Diálogos” envia as saídas para a aplicação.

Todo o processo da arquitetura está sempre em contato com três recursos: Contexto, Modelo do Usuário e Histórico. Assim, tanto processamento de entrada quanto processamento de saída são capazes de aproximar ainda mais a precisão de suas decisões.

2.6 Interfaces Tangíveis

As interfaces tangíveis permitem, por meio de objetos físicos (podendo ser de representação ou controle), que formas físicas ganhem formas digitais. Assim, há a união de representações físicas (objetos) e representações digitais (gráficos e áudio) fazendo com que haja a mediação computacional do sistema interativo (ULLMER; ISHII, 2000).

Antes do surgimento da nomenclatura “Interfaces Tangíveis”, este tipo de interface foi introduzido como “*Graspable Interface*” (Interface “Palpável”), em 1995, no qual, a partir de um pequeno ladrilho colocado em cima da superfície, um objeto virtual era criado (FITZMAURICE et. al., 1995). As interfaces tangíveis passaram a ser consideradas como um novo estilo de interação quando ISHII e ULLMER (1997) propuseram (e introduziram o termo “Interface Tangível”) três conceitos básicos para as interfaces tangíveis: Superfícies Interativas; Acoplamento entre Bits e Átomos (ou seja, permitir a ligação entre os objetos virtuais e reais); Mídia Ambiente (onde o ambiente real pode ser usado como interface de fundo da percepção humana).

A justificativa para este novo modelo de interação das interfaces tangíveis foi dada da seguinte maneira:

GUIs exploram pouco a riqueza das habilidades e sentidos humanos que as pessoas desenvolveram através do tempo de vida com o mundo físico. Nossa tentativa é de mudar os “Bits Desenhados” para “Bits Tangíveis” tirando vantagem dos múltiplos sentidos e multimodalidade da interação humana com o mundo real. Nós acreditamos que o uso de objetos palpáveis e mídia ambiente vai nos levar a uma experiência muito mais rica da informação digital. (SHAER; HORNECKER apud ISHII; ULLMER, 2009).

Um dos primeiros protótipos de interfaces tangíveis que se tem conhecimento é o *Tangible Geospace*, desenvolvido pelo MIT, que era baseado em um mapa interativo do campus do MIT projetado em uma superfície de acrílico. Ao colocar objetos físicos sobre a superfície como, por exemplo, modelos em acrílico do domo do MIT, faz com que o mapa se redimensione de forma a posicionar o prédio relativo ao modelo logo abaixo do mesmo. Por meio de uma pequena tela móvel, é possível

também resgatar uma representação 3D da área abaixo da tela (ISHII; ULLMER, 1997). A Figura 18 exibe a interface do *Tangible Geospace*.



Figura 18 - Interface do Tangible Geospace (ISHII; ULLMER, 1997)

Embora o primeiro protótipo, a *Tangible Geospace*, tenha sido apresentado em 1997, o primeiro dispositivo que pode ser considerado uma interface tangível é a *Slot Machine* (PERLMAN, 1976), desenvolvido por Radia Perlman em 1976. Por meio da *Slot Machine*, cartas de plástico representavam partes da construção de uma linguagem de programação para o ensino. A máquina era composta por três racks coloridos onde em cada rack, as cartas eram colocadas sequencialmente a fim de especificar as ações do programa. Em cada rack havia um botão “*Do it*” (Faça Isto) e então as cartas eram lidas da esquerda para a direita e interpretadas.

Durrell Bishop projetou a *Marble Answering Machine* (POYNOR, 1995) em 1992 e esta pode também ser chamada interface tangível de acordo com a conceitualização apresentada. Trata-se de um telefone com uma secretária eletrônica na qual cada mensagem de voz recebida era representada por meio de bolas de gude coloridas. Elas eram roladas por um pequeno corredor na própria máquina e eram colocadas enfileiradas de acordo com as mensagens recebidas. Quando colocadas em outra taça no próprio telefone, as mensagens eram reproduzidas. Havia também uma segunda taça que, ao receber um mármore, fazia com que o telefone chamasse o número relacionado ao mármore. A Figura 19 ilustra a *Marble Answering Machine*.

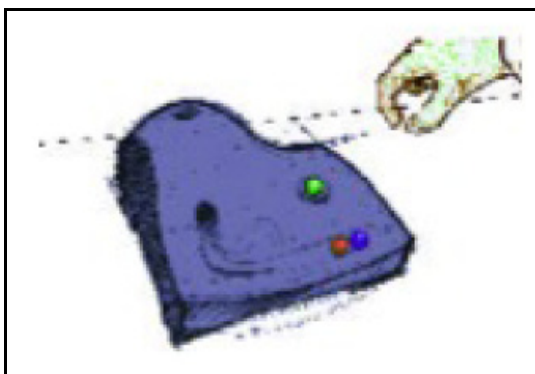


Figura 19 - *Marble Answering Machine* (POYNOR, 1995)

A partir do conceito de interfaces tangíveis apresentado anteriormente, SHAER e HORNECKER (2009) apresentaram quatro áreas de pesquisa nas quais a interface tangível está ativamente presente: **Realidade Aumentada Tangível**, **Monitores Ambientais**, **Interfaces de Usuário Incorporadas** e **Interação Tangível Tabletop**.

2.6.1 Realidade Aumentada Tangível

A realidade aumentada tangível realiza a combinação de elementos de realidade aumentada com interações tangíveis. LEE et. al. (2004) definem realidade aumentada tangível como sendo uma interface na qual há a combinação de meios de entrada característicos da interface tangível juntamente com os meios de visualização e saída de realidade aumentada (RA). Assim, utilizando objetos físicos reais, as informações exibidas por meio de RA podem ser manipuladas.

O *MagicBook* (BILLINGHURST et. al., 2001) pode ser considerado como uma interface de realidade aumentada tangível. Trata-se de um livro real contendo páginas marcadas com padrões utilizados pela realidade aumentada e, utilizando um monitor portátil de realidade aumentada, as páginas do livro são virtualizadas e seus padrões processados de forma a aplicar a realidade aumentada na visualização. A tangibilidade desta interface é dada por haver a manipulação de um livro físico, sendo possível paginá-lo e alterar a visualização fornecida pela RA. A Figura 20 exibe a interface do *MagicBook*.

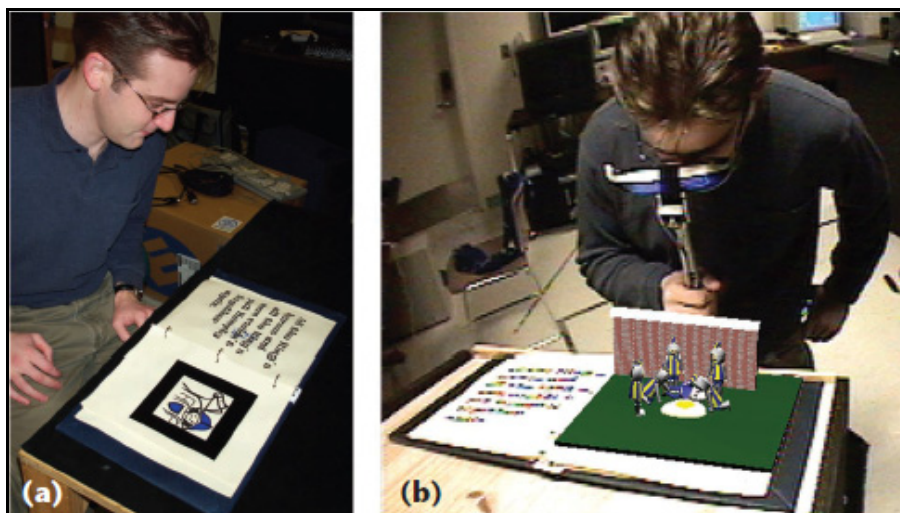


Figura 20 - Interface do *MagicBook*. (a) Caderno físico real (b) Caderno físico com RA. (Adaptado de BILLINGHURST et. al., 2001)

2.6.2 Monitores Ambientés

ISHII e ULLMER (1997) propuseram três objetivos em seu artigo *Tangible Bits*, entre eles, a utilização do ambiente como plano de fundo para auxiliar a percepção humana. Assim, os monitores ambientes utilizam objetos físicos para complementar o ciclo de interação tangível do usuário com a interface.

BREWER et al. (2007) apresentaram o Nimio, um sistema que utiliza vários objetos físicos interligados por meio de rede sem fio, porém utilizados de forma individual. Os objetos são pequenos brinquedos de silicone portando sensores como meios de entrada e LEDs como meios de saída. Desta forma, utilizando os sensores, os brinquedos são capazes de captar sons e toques e reagir (acendendo os LEDs), fazendo com que, por meio da interligação pela rede sem fio, os demais brinquedos também alterem seu estado com diferentes cores e modos. A Figura 21 exibe um objeto pertencente ao sistema Nimio.



Figura 21 - Objeto pertencente ao sistema do Nimio reagindo (BREWER; WILLIAMS; DOURISH, 2007)

2.6.3 Interfaces de Usuário Incorporadas

As interfaces de usuário incorporadas utilizam tecnologias embarcadas e incorporadas com o fim de possibilitar uma manipulação direta de interfaces incorporadas em objetos físicos.

Então, porque os usuários não podem manipular dispositivos de várias maneiras - espremer, chacoalhar, pincelar, inclinar - como uma parte integral de sua utilização? (...) Nós queremos levar o design de interface de usuário um passo à frente para uma integração mais próxima do corpo físico do dispositivo com os conteúdos virtuais e o monitor gráfico do conteúdo (SHAER; HORNECKER, 2009 apud FISHKIN et al., 2000).

O *Siftables* (MERRILL; KALANITHI; MAES, 2007) é uma interface de usuário incorporada em que pequenos blocos são compostos de pequenos monitores LCD, baterias, placas-mãe, micro controladores, infravermelhos, rádios sem fio e acelerômetros. Cada bloco é responsável por exibir dados em sua tela LCD e, por meio de comunicação sem fio, os dados são inseridos nos blocos. Os sensores infravermelhos e acelerômetros fazem com que gestos feitos com os blocos e proximidade de outros blocos sejam capturados e assim as interações ocorram.

HUNTER e MERRILL (2010) utilizaram o *Siftables* para a criação de aplicações do tipo “Faça um Enigma” e “Tele História” para auxiliar no processo de aprendizagem da linguagem de crianças entre quatro e sete anos. Estas aplicações

utilizam o Siftables combinado com quatro gestos principais: Adjacência, Chacoalhar, Seleção Manual e Inclinação.

Com a aplicação “Faça um Enigma”, as crianças podiam organizar três blocos de palavras a fim de formar uma sentença e um quarto bloco exibiria, a partir das palavras organizadas nos três blocos, uma imagem correspondente.

Já com a aplicação “Tele História”, um cenário contendo um cachorro e um gato era exibido em uma tela grande para as crianças e estas podiam manipular a história utilizando o Siftables por meio do posicionamento de blocos adjacentes (ações) e também levantando blocos correspondentes ao dia (Sol) e à noite (Lua). O cenário era atualizado na tela grande. A Figura 22 exhibe as aplicações “Faça um Enigma” (esquerda) e “Tele História” (direita)

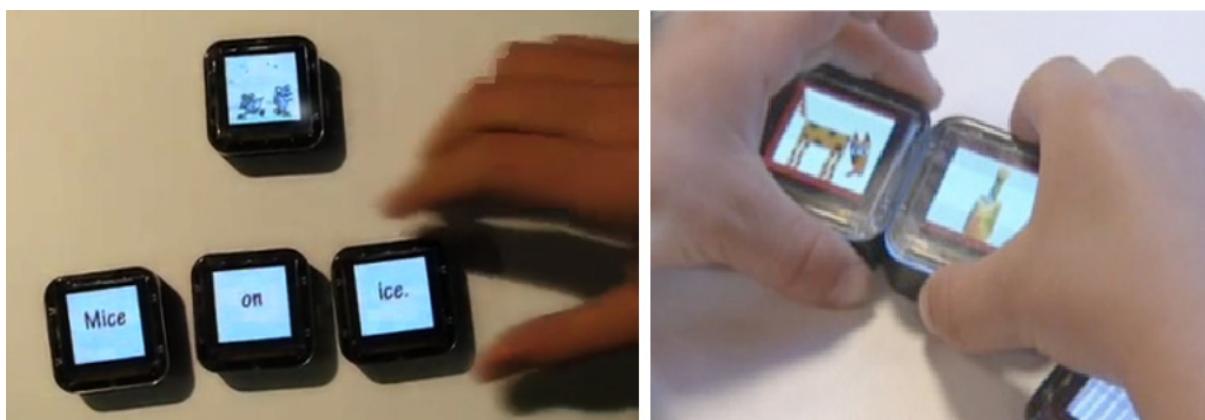


Figura 22 - "Faça um Enigma" (esquerda) e "Tele História" (direita) (Adaptado de HUNTER e MERRILL, 2010)

2.6.4 Interface Tangível *Tabletop*

Os sistemas interativos denominados *tabletop* fazem uso dos recursos e técnicas oferecidos pelas superfícies multitoques. Assim, a interação pode ocorrer a partir de gestos utilizando os dedos, as mãos ou até mesmo canetas e, por meio destas interações, objetos virtuais podem, então, ser manipulados (Shaer; Hornecker, 2009).

Por suportarem também a adição de objetos físicos como forma de manipulação e também de controle, fazendo com que estes objetos físicos sejam também capazes de controlar e manipular informações virtuais (JORDÀ, 2010), os sistemas interativos *tabletop* podem ser dotados de tangibilidade, nomeado Interação Tangível *Tabletop* ou Interfaces Tangíveis *Tabletop*.

A *Reactable* é uma interface tangível *tabletop* voltada para a área musical. Trata-se de uma superfície na qual, a partir de objetos colocados sobre ela, músicas e efeitos são criados e misturados. Ao colocar o objeto sobre a mesa, a interface reagirá de forma a exibir as ligações sonoras entre os objetos e também possibilitar a movimentação do objeto (e/ou rotação do mesmo) a fim de alterar os sons/efeitos que estão sendo executados (JORDÀ, 2010). A Figura 23 ilustra a interface da *Reactable*.



Figura 23 - Interface da *Reactable*

2.7 Considerações Finais

Neste capítulo, foram apresentadas as interfaces computacionais desde as interfaces mais rústicas, nas quais a interação era feita utilizando chaveamento de circuitos, até as interfaces Pós-WIMP, com todas as suas diferentes nuances, e o objetivo de oferecer aos usuários interações cada vez mais próximas da forma natural com a qual interagem com o mundo real em seu cotidiano.

Como foco deste trabalho, no capítulo a seguir, as interfaces tangíveis *tabletop* terão seus modos de desenvolvimento explorados por meio da apresentação da arquitetura e dos frameworks que compõem a construção deste tipo de interface.

Capítulo 3

DESENVOLVIMENTO DE INTERFACES TANGÍVEIS *TABLETOP*

Neste capítulo, serão apresentados os principais aspectos envolvidos na arquitetura e no desenvolvimento das interfaces tangíveis *tabletop* e suas aplicações, como protocolos, *frameworks* e rastreadores. As escolhas das ferramentas para o desenvolvimento dos protótipos do laboratório WINDIS também serão apresentadas.

3.1 Ferramentas de Suporte ao Desenvolvimento para Interfaces Tangíveis *Tabletop*

Construir com sucesso uma interface tangível *tabletop* depende de várias questões, entre elas, um bom projeto de software (SHAER, 2009). Um projeto de software voltado para a construção de interfaces tangíveis pode ser dividido em três partes:

- Um protocolo definido para a codificação de dados a serem transportados de um ponto até outro.
- Um *tracker* (rastreador) para a captura de interações com a interface.
- Uma aplicação cliente para processar as interações capturadas pelo *tracker*.

3.1.1 TUIO

O *Tangible User Interface Objects* (TUIO) é um framework aberto que define um protocolo e uma API para interfaces tangíveis multitoques. Ele foi criado com o intuito de suprir as necessidades das superfícies interativas nas quais o usuário pode manipular um conjunto de objetos (KALTENBRUNNER et. al., 2005). Inicialmente, foi projetado como parte do sintetizador Reactable e, então, passou a ser usado em outros projetos.

O protocolo permite a transmissão de descrições abstratas de superfícies interativas, incluindo eventos e estados de objetos tangíveis. Dessa forma, os *trackers* utilizam o protocolo para codificar os dados e enviam para uma aplicação cliente que os decodificam e utilizam seus eventos e estados para alterar a aplicação. A combinação de *trackers*, protocolo e aplicações clientes possibilita o desenvolvimento de interfaces tangíveis. A Figura 24 ilustra uma arquitetura de interface tangível *tabletop* utilizando o protocolo TUIO.

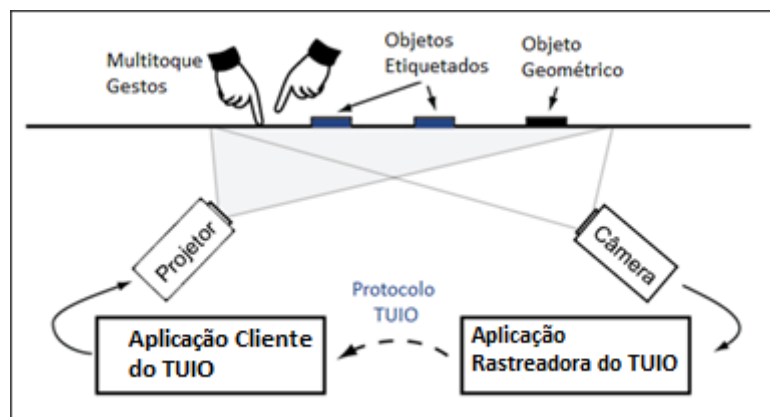


Figura 24 - Arquitetura de uma interface tangível utilizando o protocolo TUIO (Adaptado de TUIO, 2011)

3.1.2 Especificação do TUIO

O protocolo TUIO é codificado utilizando o formato *Open Sound Control* (OSC) que provê eficiente codificação para transmissão de dados. Assim, as mensagens do TUIO podem ser enviadas por qualquer canal que suporte OSC. Por padrão, as mensagens TUIO são enviadas em pacotes UDP pela porta 3333 e este

método de transporte é comumente denominado TUIO/UDP (KALTENBRUNNER et. al., 2005).

3.1.2.1 Mensagens

O TUIO define duas classes de mensagens principais:

- SET: Envia informação sobre o estado do objeto como posição, orientação e outros estados. O envio é feito a cada mudança de estado.
- ALIVE: Conjunto de identificadores de sessão de objetos que estão presentes na interface. O envio pode ser feito periodicamente.

O protocolo não define explicitamente mensagens de ADD e REMOVE. Para controlar a adição e remoção de objetos, o receptor das mensagens TUIO faz a comparação entre as mensagens sequenciais ALIVE. Além das mensagens SET e ALIVE, mensagens FSEQ são definidas de forma a identificar, por meio de um ID único, um bloco contendo várias mensagens de SET e ALIVE. Opcionalmente, mensagens SOURCE podem ser utilizadas para identificar para onde as mensagens serão enviadas. Caso não seja definido o SOURCE, as mensagens serão enviadas para a própria máquina (*localhost*).

As mensagens do protocolo TUIO tem o seguinte formato (TUIO, 2011):

```
/tuo/[profile] source [aplicação@endereço]
/tuo/[profile] alive s_id0 ... s_idN
/tuo/[profile] set Session_ID [atributos]
/tuo/[profile] fseq f_id
```

Mensagens SET definem os atributos de um objeto de acordo com os atributos da Figura 25.

s	Session ID (identificação do objeto temporário)	int32
i	Class ID (ID do marcador, por exemplo)	int32
x, y, z	Posição	float32, uma faixa de 0 ... 1
a, b, c	Ângulo	float32, faixa de 0 .. 2PI
w, h, d	Dimensão	float32, intervalo 0 .. 1
f, v	Area, Volume	float32, intervalo 0 .. 1
X, Y, Z	Vetor velocidade (velocidade de movimento e direção)	float32
A, B, C	Rotação vetor velocidade (velocidade de rotação e direção)	float32
m	aceleração do movimento	float32
r	aceleração da rotação	float32
P	parâmetro livre	tipo definido pelo OSC cabeçalho da mensagem

Figura 25 - Atributos das mensagens SET do TUIO (Adaptado de TUIO, 2011)

3.1.2.2 Profiles

Os *profiles* (perfis) são definidos de forma a viabilizar a transmissão de cursores, objetos e *blobs*¹ dentro de planos bidimensionais e tridimensionais. Além dos *profiles* já pré-definidos, é possível a definição de *profiles* customizados de acordo com a necessidade de cada aplicação. A Figura 26 exhibe os *profiles* pré-definidos pelo TUIO e também o formato possível de *profiles* customizados (TUIO, 2011).

```
2D Interactive Surface
/tuio/2Dobj set s i x y a X Y A m r
/tuio/2Dcur set s x y X Y m
/tuio/2Dblb set s x y a w h f X Y A m r

2.5D Interactive Surface
/tuio/25Dobj set s i x y z a X Y Z A m r
/tuio/25Dcur set s x y z X Y Z m
/tuio/25Dblb set s x y z a w h f X Y Z A m r

3D Interactive Surfaces
/tuio/3Dobj set s i x y z a b c X Y Z A B C m r
/tuio/3Dcur set s x y z X Y Z m
/tuio/3Dblb set s x y z a b c w h d v X Y Z A B C m r

custom profile
/tuio/_ [formatString]
/tuio/_sixyP set s i x y 0.57
```

Figura 26 - Profiles do TUIO (TUIO, 2011)

3.1.2.3 Eficiência e Confiabilidade

Para maior eficiência, as mensagens são todas colocadas em um pacote a fim de aproveitar melhor o espaço de um pacote UDP. Porém, com a utilização do protocolo de transmissão UDP para a diminuição da latência, há o risco de perda de pacotes durante a transmissão. Para garantir a confiabilidade, o TUIO inclui informações redundantes para corrigir possíveis perdas de pacotes. É possível usar o protocolo TCP, porém a latência deve aumentar (TUIO, 2011).

¹ Blobs são objetos genéricos não identificados

3.1.3 Trackers

Trackers (rastreadores) são softwares que, a partir de uma imagem, captam interações com a interface tangível. Essas interações podem ser toques, gestos, objetos tangíveis, marcadores fiduciais e infravermelhos, entre outros. Um exemplo contendo seis marcadores fiduciais é exibido na Figura 27.

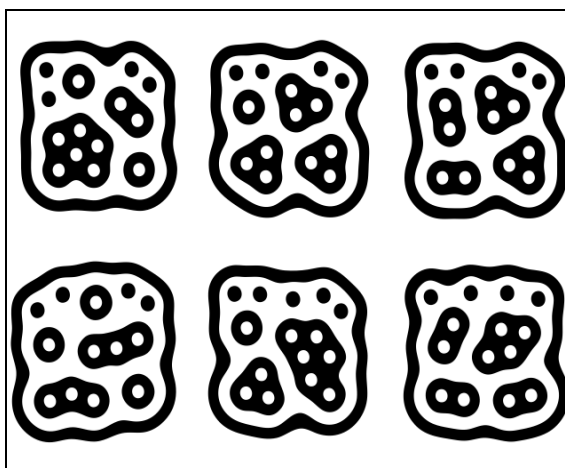


Figura 27 - Marcadores Fiduciais (REACTIVISION, 2011)

Existem diversos *trackers* que são compatíveis com o protocolo TUIO. Os *trackers* oficialmente reconhecidos pelo TUIO.org são (TUIO, 2011):

- reactIVision: Possui rápido e robusto rastreamento de marcadores fiduciais, além de algum suporte a multitoques. Originalmente criado para o uso com a Reactable, foi adaptado para uso geral.
- Touchlib: Primeira biblioteca livre voltada para superfícies multitoques. Suporta apenas o sistema operacional Windows e possui suporte limitado a marcadores fiduciais.
- Community Core Vision: *Framework* voltado a multitoques. Posteriormente foi adicionado um *add-on* para ter algum suporte a marcadores fiduciais. Possui interface para várias câmeras.
- Tuiokinect: *Tracker* desenvolvido para o uso do Kinect da Microsoft, realizando o rastreamento de movimentos corporais.
- Touché: *Tracker* exclusivo do Mac OSX. Possui apenas suporte a multitoques.

- Surface Tracker: Voltado para o rastreamento de baixa latência de multitoques. Suporta apenas câmeras de baixa latência com taxas de até 100 fps (quadros por segundo).
- Ortholumen: *Tracker* para o rastreamento de canetas de luz.
- Wiimote Whiteboard: Suporta o rastreamento de movimentos a partir do Wiimote.
- MSA Remote: A superfície multitoques é projetada na tela de um iPhone e por meio deste *tracker*, as interações feitas na tela do iPhone são enviadas para a superfície real.

Dados os *trackers* acima listados, a Tabela 1 exhibe um comparativo entre as características dos *trackers* voltados a superfícies tangíveis baseadas em multitoques e marcadores fiduciais (reacTIVision, Community Core Vision, Touché, Touchlib e Surface Tracker). Os critérios de comparação são:

- Suporte a multitoques
- Suporte a marcadores fiduciais
- Plataformas suportadas
- Técnica de iluminação suportada
 - *Frustrated Total Internal Reflexion* (FTIR) – Utiliza luz infravermelha paralela a um acrílico qualquer para detectar toques e objetos, permitindo o uso de uma camada de silicone para deixar o toque agradável.
 - *Diffused Surface Illumination* (DSI) – Utiliza a mesma técnica de infravermelho do FTIR, porém com um acrílico específico chamado “*Endlighten*” e não permitindo o uso de uma camada de silicone, mas de vidro.
 - *Diffused Illumination* (DI) – Os infravermelhos são posicionados sob o acrílico e iluminam a parte de baixo do mesmo.
 - *Laser Light Plane* (LLP) – Lasers são posicionados alguns poucos milímetros acima da superfície formando uma camada de infravermelho.

Tabela 1 - Comparativo das características dos *trackers*

	Multitoques	Fiducial	Plataforma	Técnica
reactIVision	Básico	Avançado	Multiplataforma	FTIR, DI
Community Core Vision	Avançado	Fraco	Multiplataforma	FTIR, DI, DSI, LLP
Touchlib	Básico	Não Possui	Windows	FTIR, DI
Touché	Básico	Não Possui	Mac OSX	FTIR, DI
Surface Tracker	Avançado	Não Possui	Windows	LLP

Portanto, utilizando *trackers* acima descritos, os dados referentes às interações por meio de toque, marcadores, lasers e toda interação suportada por cada *trackers*, são codificados com base no protocolo TUIO e enviados para as aplicações que farão uso destes dados. Essas aplicações são desenvolvidas com base em frameworks que o TUIO.org denomina *Application Frameworks*.

3.1.4 *Application Frameworks*

Application Frameworks são *frameworks* (ou *scripts*, bibliotecas e similares) compatíveis com o protocolo TUIO e, a partir destes *frameworks*, é possível construir aplicações destinadas a interfaces tangíveis que recebam dados codificados por meio do protocolo TUIO.

É possível desenvolver aplicações em diversas linguagens de programação, uma vez que existem diversas implementações de *Application Frameworks* para muitas das linguagens mais populares e também de linguagens menos conhecidas. O TUIO.org define uma lista de *Application Frameworks* para várias linguagens de programação. Alguns dos principais *Application Frameworks* são (TUIO, 2011):

PyMT: *Framework open source* baseado na linguagem Python. Possibilita o ágil desenvolvimento de aplicações multitoques com *widgets* pré-construídos e customizáveis.

MT4j: *Framework open source* baseado na linguagem Java. Possui todo o *background* robusto da plataforma Java além de possuir formas pré-definidas.

MIRIA: *Framework open source* baseado no *framework* Silverlight, da Microsoft.

Grafiti: *Framework open source* para reconhecimento de gestos baseado na linguagem C#. Foco principal em objetos tangíveis.

Além destes, existem vários outros *Application Frameworks* baseados em várias linguagens de programação.

KAMMER et. al. (2010) definiu critérios nos quais os *Application Frameworks* podem ser avaliados. A Figura 28 exhibe os critérios.

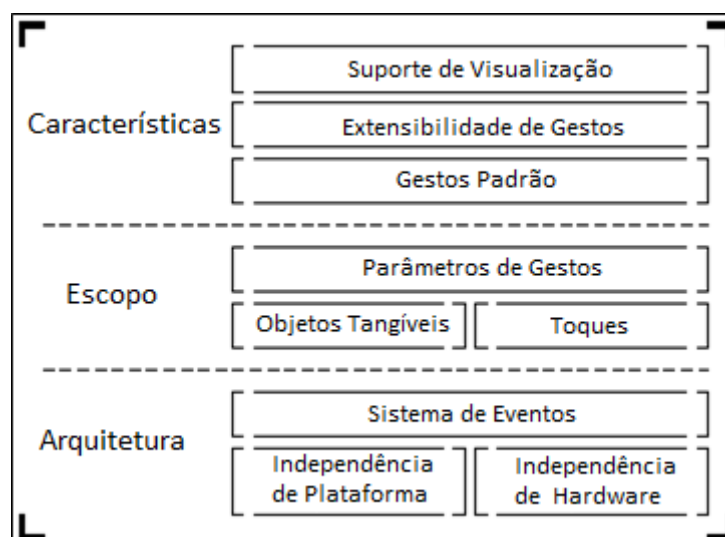


Figura 28 - Critérios de Frameworks Multitoques (Adaptado de KAMMER et. al., 2010)

Os critérios apresentados na Figura 28 são divididos em três níveis: arquitetura, escopo e características.

A arquitetura de um *framework* pode ser avaliada com base em três critérios: independência de plataforma, independência de hardware e sistema de eventos. A independência de plataforma provê ao *framework* a liberdade de ser utilizado em qualquer sistema operacional. Seguindo a mesma linha teórica, a independência de hardware permite que, não importa qual seja o hardware que executará a aplicação desenvolvida no *framework*, a aplicação funcionará sem problemas. Por fim, o sistema de eventos diz respeito à forma na qual os eventos são tratados internamente pelo *framework*. Alguns *frameworks* tem um servidor centralizado para o tratamento de eventos e estes são feitos por meio da rede (KAMMER et. al., 2010).

Já o escopo de um *framework* tem como base os seguintes critérios: objetos tangíveis, toques e parâmetros de gestos. O critério de objetos tangíveis define o suporte que o *framework* oferece para a exploração dos objetos tangíveis no desenvolvimento de aplicações. O critério Toques se baseia nas informações (parâmetros) sobre toques nos quais o *framework* tem a capacidade de trabalhar. Da

mesma forma do critério Toques, o critério Parâmetro de Gestos possibilita também se baseia no suporte a parâmetros de gestos e seu tratamento a partir do *framework* (KAMMER et. al., 2010).

Por fim, as características do *framework* podem ser avaliadas seguindo os critérios: suporte de visualização, extensibilidade de gestos e gestos padrão. O suporte de visualização diz respeito ao suporte gráfico dado pelo *framework* para o desenvolvimento da aplicação, como por exemplo, componentes prontos como botões e menus. Os gestos padrão são os tipos de gestos suportados nativamente. Podem ser *online*, tendo os gestos processados imediatamente de acordo com a interação, ou *offline* que é feito o processamento apenas após o final da interação do gesto (por exemplo, um gesto circular para abrir um menu). A extensibilidade de gestos é a forma pela qual é possível criar novos gestos (KAMMER et. al., 2010).

Com base na maioria dos critérios citados, a Tabela 2 realiza um comparativo entre alguns dos *Application Frameworks* exemplificados anteriormente.

Tabela 2 - Comparativo entre os *Application Frameworks*

	Indep. De Plataforma	Sist. de Eventos	Objetos Tangíveis	Param. Toques	Param. Gestos	Gestos Padrão	Suporte Visualização
PyMT	Sim	Decentralizado	Sim	Sim	Não	Online Offline	Widgets Próprios
MT4j	Sim	Decentralizado	Sim	Sim	Sim	Online	Widget Próprios
MIRIA	Sim	Decentralizado	Não	Sim	Sim	Online	Controles WPF
Grafiti	Sim	Centralizado	Sim	Sim	Sim	Online	Não Possui

3.2 Desenvolvimento *Tabletop* e Metodologias de Desenvolvimento

Assim como em projetos de software para interfaces WIMP, o desenvolvimento de aplicações no contexto das interfaces tangíveis *tabletop* também pode ser apoiado por metodologias de desenvolvimento, que são soluções estratégicas e metódicas de desenvolvimento utilizadas pelos engenheiros de software para organizar em fases o processo de desenvolvimento.

Embora as *tabletops* sejam diferentes tanto esteticamente quanto nas formas de interação oferecidas ao usuário, não é necessário criar uma nova metodologia de

desenvolvimento para elas. A mudança ocasionada pelo paradigma acontece internamente em cada fase das metodologias, impactando na forma de analisar requisitos e o design.

PRESSMAN (2001) apresenta uma abordagem clássica de metodologia de desenvolvimento conhecida como “Modelo Cascata”, um modelo linear e sequencial contendo quatro fases: Análise, Design, Código e Teste. A estrutura do modelo cascata, quando utilizada em projetos de aplicações *tabletop*, não é alterada em nenhum ponto. Entretanto, as atividades em cada fase são adaptadas para este tipo de interface. A Figura 29 ilustra o modelo cascata.

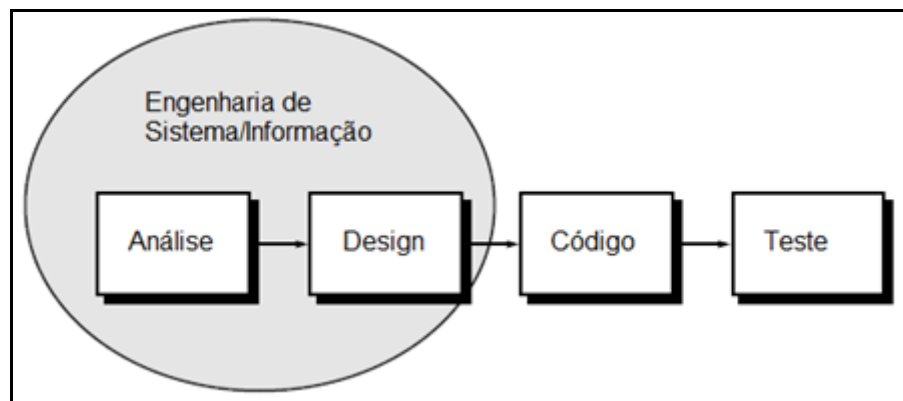


Figura 29 - Modelo Cascata (Adaptado de PRESSMAN, 2001)

HARTSON e HIX (1989) exibem o “Modelo Estrela”, um ciclo de vida voltado para o desenvolvimento de interfaces humano-computador. Este modelo propõe que as fases podem ser realizadas em qualquer ordem desde o início do ciclo de vida com avaliações entre uma fase e outra. Assim, a ordem do ciclo de vida sempre será da seguinte forma: “FASE, AVALIAÇÃO, FASE, AVALIAÇÃO, FASE, AVALIAÇÃO” e assim sucessivamente fazendo com que a fase “Avaliação” seja o centro do modelo, formando assim uma estrela composta das seguintes etapas: Implementação, Prototipagem, Especificação de Requisitos, Análise de Tarefas / Análise Funcional, Design Conceitual / Design Formal e Avaliação. Assim como no modelo cascata, o modelo estrela pode ser aplicado ao desenvolvimento *tabletop*. A Figura 30 exibe a estrutura do modelo estrela.

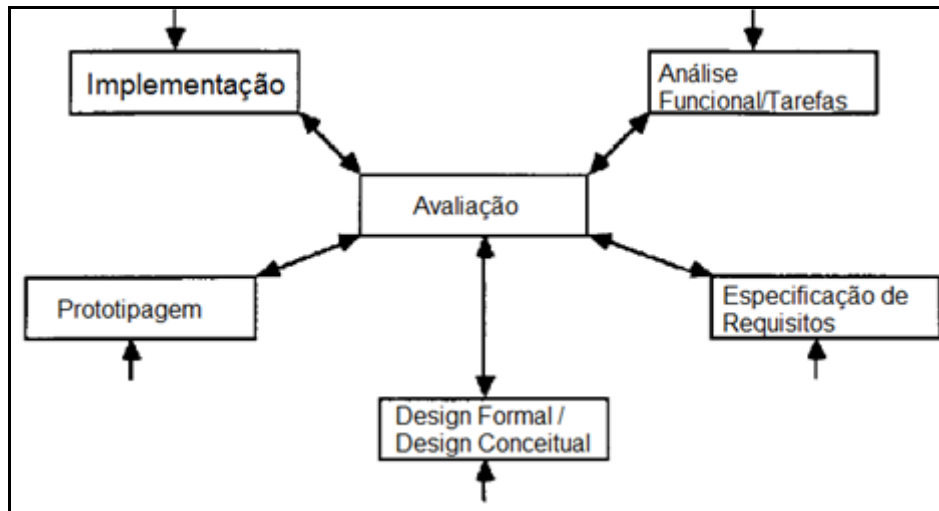


Figura 30 - Modelo Estrela (Adaptado de HARTSON e HIX, 1989)

Com a aplicação das metodologias de desenvolvimento em projetos de aplicações *tabletop* há a necessidade de adaptar as etapas individualmente. Como o foco deste trabalho é a especificação de interfaces tangíveis *tabletop* serão apresentados no Capítulo 5 dois exemplos em que os modelos cascata e estrela são adaptados para a utilização do modelo desenvolvido por este trabalho. Estes modelos foram escolhidos como exemplos com a finalidade de demonstrar o modelo TTUI-SM inserido em um modelo clássico (cascata) e também em um modelo voltado para a IHC.

3.3 Protótipo *Tabletop* do laboratório WINDIS

Com base nas técnicas existentes para a construção de interfaces tangíveis *tabletop*, os pesquisadores integrantes do laboratório *Wireless Networks and Distributed Interactive Simulation* (WINDIS) desenvolveram seu próprio protótipo de interface tangível *tabletop*. O protótipo em funcionamento é exibido na Figura 31.



Figura 31 - Protótipo *Tabletop* WINDIS

O protótipo foi construído em forma de mesa com aproximadamente 100 cm de altura e a superfície *tabletop* tem cerca de 37 polegadas. Sua arquitetura é constituída por:

- 01 Macbook Mini
- 01 Câmera PS-EYE
- 01 Fonte de Computador
- 01 Projetor de Curto Alcance
- 112 Leds Infravermelhos
- 03 Leds de Potência
- 01 Placa de Acrílico
- 01 Camada de silicone líquido
- 01 Película de Projeção

Dentre as diversas possibilidades de aplicações que podem ser desenvolvidas para o protótipo *tabletop* de nosso laboratório, as aplicações voltadas ao domínio de comando e controle têm tido grande foco devido à crescente

demanda de implantação de novas tecnologias para auxiliar o cenário deste tipo de operação (RINTAKOSKI; ALHO, 2008).

JOINT PUBLICATION (2012) define comando e controle como sendo:

“O exercício da autoridade e direção por um comandante designado sobre forças atribuídas e anexadas no cumprimento da missão. Funções de comando e controle são realizadas por meio da organização do pessoal, equipamento, comunicações, instalações, e procedimentos empregados pelo comandante em planejamento, direcionamento, coordenação, e controle de forças e operações no cumprimento da missão. Também chamado C2.”

Desta forma, dois tipos de operações de comando e controle serão introduzidos no Capítulo 6 e em seguida suas interfaces de demonstração serão especificadas.

3.3.1 Ferramental de Desenvolvimento das Aplicações do Protótipo

O leque de opções entre *trackers* e *application frameworks* é grande. É possível fazer diversas combinações viabilizando o uso de um *tracker* com vários *application frameworks* da mesma forma que é possível fazer o uso de um *application frameworks* com vários *trackers*.

Para desenvolvimentos, serão utilizados os *trackers* reactIVision e/ou Community Core Vision em conjunto com o *application framework* MT4j. A Figura 32 ilustra a arquitetura com esta escolha. Tal escolha foi feita baseada nas seguintes características:

- reactIVision possui o melhor suporte para marcadores fiduciais que serão amplamente utilizados em projetos futuros. Para aplicações que necessitem primariamente de um rastreamento robusto de marcadores fiduciais, o reactIVision será utilizado.
- Community Core Vision realiza um excelente rastreamento de toques fazendo com que a eficiência dos mesmos seja alta. Para aplicações que não necessitem de um robusto sistema de rastreamento de marcadores fiduciais, o Community Core Vision será utilizado.

- MT4j possibilita um rápido desenvolvimento de aplicações e customizações além de ser robusto e possuir um forte suporte a diversos tipos de interação e um sólido suporte à interações tangíveis.

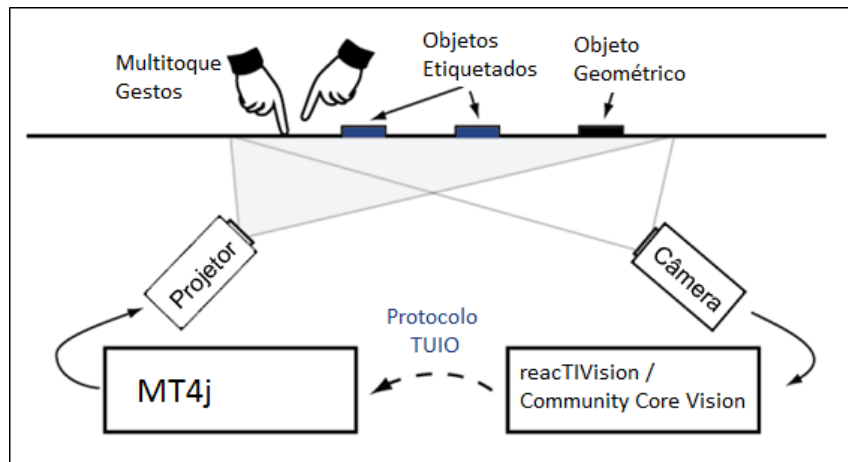


Figura 32 - Arquitetura *tabletop* escolhida (Adaptado de TUIO, 2011)

3.4 Considerações Finais

Neste capítulo, foram apresentadas as principais ferramentas que suportam o desenvolvimento de aplicações para as interfaces tangíveis *tabletop* como o protocolo TUIO, os *trackers* de reconhecimento de gestos e objetos e os principais *frameworks* de codificação. Foi também apresentado o protótipo *tabletop* do laboratório WINDIS juntamente com a escolha do ferramental a ser utilizado para o desenvolvimento das aplicações. Por fim, foi exemplificado por meio de dois modelos de desenvolvimento (cascata e estrela), no qual se insere a especificação das interfaces no ciclo de desenvolvimento. Assim, no próximo capítulo serão apresentados os diversos tipos de modelo de especificação de interface atuais.

Capítulo 4

TRABALHOS RELACIONADOS

Neste capítulo, serão apresentadas as principais técnicas de especificação utilizadas para realizar a descrição de diversas modalidades de interface. Com base nestas técnicas, será realizada uma discussão a fim de apontar as limitações encontradas e que justifiquem a concepção de um novo modelo de especificação.

4.1 Trabalhos Relacionados

BROWNE et al. (1986) fizeram o uso de *Command Language Grammar* (CLG) para realizar a especificação formal de interfaces de usuário adaptativas. Tal abordagem foi usada por possibilitar uma descrição concreta das interações do usuário além traçar as tarefas do usuário em nível sintático e semântico.

PAYNE e GREEN (1986) apresentaram a *Task-Action Grammar* (TAG) buscando um formalismo para linguagens de interface por meio da representação mental de *Task Languages* (Linguagens de Tarefa). Assim, todas as tarefas da interface são mapeadas a partir da TAG.

PARNAS (1969) fez o uso de diagramas de transição de estados para realizar o design de interfaces de usuário de sistemas interativos apontando como uma alternativa ao uso de CLGs.

HAREL (1987) utiliza *Statecharts* (Diagramas de estado) para descrever formalmente o comportamento dinâmico de sistemas complexos.

O *eXtensible Interface Markup Language* (XIML) (PUERTA; EISENSTEIN, 2001) apresenta uma abordagem na qual três requisitos, considerados importantes

para linguagens com tal finalidade, são trabalhados: (1) Suportar funções de design, operação, organização e avaliação; (2) Possibilitar o relacionamento de dados abstratos e concretos da interface e (3) Permitir que sistemas baseados em conhecimento explorem dados capturados.

PHANOURIOU (2000) propôs em sua tese o *User Interface Markup Language* (UIML) com o objetivo de possibilitar a construção de interfaces de usuário independentes de dispositivos. O UIML implementa um meta-modelo criado e chamado *Meta-Interface Model* (MIM).

O *USer Interface eXtensible Markup Language* (USIXML) (LIMBOURG et al., 2004) leva em consideração o contexto no qual a interface está inserida ao especificar tal interface e seus diversos níveis de detalhe por meio de vários modelos. Tais modelos são aplicados de acordo com o ciclo de vida apresentado por CALVARY et al. (2003).

O *Abstract User Interface Markup Language* (AUIML) (AZEVEDO; MERRICK; ROBERTS, 2000) foi criado com o objetivo de definir a intenção das interações realizadas pelos usuários, fazendo com que os designers se concentrem principalmente na semântica da interação do usuário.

FIGUEROA, GREEN, M. e HOOVER (2002) criaram o *Interaction Technique Markup Language* (InTml), visando a especificação da interações em 3D. Além disso, o InTml também se integra com aplicações de realidade virtual.

RANEBURGER et al. (2011) propuseram modelos de estrutura e comportamento para especificação de interfaces WIMP para complementar os modelos de tarefa e discurso que não realizavam a modelagem propriamente dita. Com a finalidade de oferecer geração automática de interfaces WIMP, os modelos são constituídos de dados oriundos de uma etapa anterior na qual se encontra o “Modelo de Comunicação”. A especificação dos modelos de estrutura e comportamento é feita por meio de componentes hierárquicos e em seguida são geradas pré-visualizações de tela, seguidas do código fonte final.

O Chasm é um modelo de especificação de interfaces de usuário. Possui uma linguagem de domínio chamada ChasmXML e é dividido em quatro camadas: *Envisioned Behaviour Tier*, onde o designer coloca, em modo puramente textual, todo a sua compreensão sobre a interface; *Casuality Tier*, onde os relacionamentos são declarados na forma “Quando X e Y acontecer, então Z”; *Automata Tier*, onde é criada uma representação em forma de autômato para as interações da interface; e

Code Tier, onde é armazenado o código gerado a partir das camadas anteriores (WINGRAVE; JR, 2009).

ARYA e HAMIDZADEH (2004) apresentam a linguagem *Face Modeling Language* (FML), uma linguagem baseada em XML para interfaces multimídia que tem a finalidade de representar hierarquicamente as animações faciais considerando a linha de tempo da relação entre ações faciais e eventos, definir *templates* comportamentais e de capacidade, além de ser compatível com as tecnologias web e parâmetros de animação facial usadas pelo formato MPEG-4.

JACOB, DELIGIANNIDIS e MORRISON (1999) propuseram um modelo para a especificação de interfaces não WIMP. O modelo combina grafos de fluxo de dados e diagramas de transição de estados para modelar o relacionamento contínuo de componentes baseados em eventos que podem, a partir de interações discretas, ter a ativação ou desativação da continuidade de seus relacionamentos. Baseada no modelo proposto, uma linguagem foi desenvolvida para ser implementada de duas formas, uma visual e outra textual. A linguagem visual é implementada por meio da ferramenta VRED enquanto a linguagem textual se baseia na metalinguagem *Standard Generalized Markup Language* (SGML) possibilitando a inserção de trechos de código em C++.

EDWARDS e MITSOPOULOS (2005) introduziram um modelo de especificação de *widgets* não visuais. O objetivo é suportar a especificação de interfaces que serão utilizadas por deficientes visuais. Desta forma, o modelo possui três níveis de especificação. O primeiro nível é o conceitual, no qual informações relevantes sobre as tarefas e requisitos necessários para a realização da interação deverão estar devidamente abstraídos. O segundo nível é o estrutural, e nele serão determinadas as estruturas das cenas auditivas, determinando quando os sons são emitidos e se são contínuos. O terceiro e último nível é o de implementação, que define as dimensões físicas dos sons com base nas limitações dos dois níveis anteriores.

HENG et al. (2008) apresentaram um novo modelo de especificação de interações baseadas em toques para *tabletops* e PDAs. O modelo é dividido em três níveis: Nível de Ação, que representa a ação de toque realizada pelo usuário (um toque, dois toques, arrastar etc.); Nível de Motivação, que especifica o que o usuário quer fazer a partir da ação realizada; e Nível de Computação, que trata das

questões do processamento (em nível de hardware e software) da detecção e resposta das ações realizadas.

A *Gesture Definition Language* (GDL) foi criada buscando oferecer um modelo de especificação que seja utilizável para a especificação de qualquer tipo de interação baseada em gestos (Gestos de mãos livres, interações tangíveis, multitoques etc.). A GDL é composta de três conceitos abstratos: *Features* (Características), que são as propriedades básicas da interação; *Regions* (Regiões), que são as áreas onde as interações ocorrerão, bem como suas coordenadas; e *Gestures* (Gestos), que são os tipos de gestos que podem ser realizados (ECHTLER; KLINKER; BUTZ, 2010).

HOLMQUIST et al. (1999) classificaram os tipos de objetos que podem ser usados em interações tangíveis em três tipos: *Containers*, *Tokens* e *Tools*. *Containers* são os objetos que são utilizados para carregar uma informação de um dispositivo para outro. *Tokens* são objetos que são diretamente associados com uma informação presente na interface. *Tools* são objetos que realizam funções computacionais na interface, em outras palavras, que a manipulam.

Baseado no modelo MVC, ULLMER e ISHII (2000) desenvolveram um modelo de interação chamado *Model-Control-Representation physical digital* (MCRpd). Este modelo normalmente define os elementos *Model* e *Controller* do MVC, porém a camada *View* é dividida em duas partes, sendo elas representação física (utilizando objetos físicos) e representação virtual (objetos virtuais, vídeos, imagens, entre outros).

ULLMER (2002) propôs em sua tese de doutorado o conceito de *Tokens* e *Constraints* (Restrições) no qual os *tokens* são objetos físicos utilizados para representar ou controlar informações digitais e *constraints* são restrições impostas para a interação dos *tokens*. Assim, um *token* pode apenas realizar interações de acordo com suas restrições descritas e nada mais.

Com base nesta proposta de Ullmer, CALVILLO-GAMEZ et al. (2003) apresentaram um paradigma para descrição de interfaces tangíveis nomeado de *Tokens And Constraints* (TAC). O paradigma TAC estende o conceito de *Tokens* e *Constraints* proposto por Ullmer de forma que qualquer objeto deve ter restrições independentemente de suas características. Considerando TAC representando um *Token* e suas *Constraints*, o paradigma TAC tem cinco propriedades:

- Um objeto físico deve ser associado a uma *constraint* e a uma variável (informação digital) para ser considerado um *token*.
- Cada objeto físico pode ser definido como um *token*, como uma *constraint* ou como ambos.
- Um *token* necessita manter uma lista de suas *constraints*, porém uma *constraint* não precisa manter uma lista de seus *tokens*.
- Há sempre uma variável associada com cada TAC.
- Cada TAC tem um comportamento físico discreto ou contínuo.

O *Tangible User Interface Markup Language* (TUIML) (SHAER; JACOB, 2009) foi proposto de forma a permitir a especificação das interfaces tangíveis tanto em nível estrutural quanto em nível comportamental, podendo ser convertido de forma automática ou semiautomática em implementações concretas. Esta linguagem possui dois elementos principais: uma linguagem visual de especificação que utiliza formas diagramáticas e icônicas para especificar a interface, além de utilizar o paradigma TAC como base; e uma linguagem XML com *tags* pré-definidas de forma a especificar o comportamento em uma interface tangível.

4.2 Discussão

Embora todas as abordagens funcionem bem para seus propósitos e sejam grandes contribuições para o cenário de especificação de interfaces, há a ausência de um modelo formal de especificação que leve em consideração a ampla diversidade de interações por meio de gestos multitoques oferecidos pelas interfaces tangíveis *tabletop* e também que apresente o resultado da especificação de uma forma simples e organizada, a fim de facilitar a compreensão dos elementos de interface e suas interações, o que é consideravelmente importante quando a especificação é diretamente utilizada como documento base para a implementação da interface.

As abordagens baseadas em gramáticas e XML são voltadas, em geral, para as interfaces WIMP e suas interações, além de se preocuparem com diferentes aspectos de interação como contexto, interfaces adaptativas, semânticas de interação, interações 3D e bases de conhecimento, porém não abordam

características das interfaces tangíveis, além de o resultado da especificação ser de difícil compreensão. As abordagens baseadas em autômatos podem ser adaptadas para a especificação de gestos, porém, assim como alguns modelos (UIML, Chasm e o modelo para interfaces não WIMP (JACOB; DELIGIANNIDIS; MORRISON, 1999)), a especificação ficaria demasiada abstrata, além dos dados especificados não serem organizados de uma forma simples de se compreender. Algumas das abordagens realizam valiosas contribuições para a especificação de gestos como a GDL e o modelo criado por HENG et al. (2008), entretanto, ainda há o problema da organização dos dados especificados e estes modelos não suportam a especificação de interações características das interfaces tangíveis utilizando objetos físicos.

No cenário de especificação das interfaces tangíveis, alguns excelentes modelos de especificação foram desenvolvidos, como a classificação de tipos de objetos tangíveis, feita por HOLMQUIST, REDSTRÖM e LJUNGSTRAND (1999), o modelo MCRpd e o paradigma TAC. Entretanto, estes modelos voltam toda sua preocupação para a interação por meio de objetos físicos, deixando de lado todo o potencial de gestos multitoques avançados das interfaces tangíveis *tabletop*, além de também haver a carência de organização simplificada dos dados especificados.

Por fim, o modelo TUIML, no qual se baseou este trabalho, realiza a especificação de interfaces tangíveis preocupando-se com diagrama de estados, diagrama de diálogo, diagrama de tarefas e paleta baseada no paradigma TAC. Sem dúvidas, é um modelo de especificação complexo e abrangente de interfaces tangíveis, sendo uma contribuição muito notável para este contexto. Porém, tal complexidade gera diversos diagramas, fazendo com que, embora estejam organizados, os dados especificados não sejam de simples compreensão e seus diagramas estejam separados uns dos outros. Além disso, segundo o autor, o modelo não suporta uma especificação compreensível de interações baseadas em gestos (SHAER; JACOB, 2009).

Portanto, fez-se necessário o desenvolvimento de um modelo que contemple as necessidades das interfaces tangíveis *tabletop* em relação às abordagens baseadas em XML, Gramáticas, Autômatos e outras, que são genéricas; que ofereça uma maior simplicidade na organização dos dados especificados em vez de diversos diagramas, gramáticas complexas e longos arquivos XML; e que sirva de suporte à

especificação de gestos multitoques avançados, juntamente com interações tangíveis.

4.3 Considerações Finais

Neste capítulo, os diversos modelos de especificação de interfaces atuais foram apresentados juntamente com suas limitações e benefícios. Com base nas limitações, realizou-se uma discussão sobre as mesmas levando em consideração o contexto das interfaces tangíveis *tabletop* identificando as necessidades deste tipo de interface que os modelos atuais não suprem. Assim, no próximo capítulo será apresentado o modelo TTUI-SM que visa suportar as necessidades que os modelos atuais não fazem.

Capítulo 5

MODELO DE ESPECIFICAÇÃO DE ELEMENTOS DE INTERFACE TTUI-SM

Como forma de facilitar o processo de especificação de elementos de interfaces tangíveis *tabletop*, considerando também sua modalidade de interação multitoques, foi criado um modelo que visa organizar a especificação dos elementos de interface desta modalidade, desde o design dos elementos até seus meios de interação. Além disso, o laboratório WINDIS, do Departamento de Computação da Universidade Federal de São Carlos (onde este projeto de mestrado foi desenvolvido), como membro do Instituto Nacional de Ciência e Tecnologia em Sistemas Embarcados Críticos (INCT-SEC), desenvolve pesquisa que utiliza interfaces tangíveis *tabletop* em diferentes domínios de aplicação (segurança pública e gerenciamento de emergência), gerando assim uma demanda por um modelo de especificação de interfaces para *tabletops* que facilite e agilize a especificação e também a codificação das interfaces para estas aplicações.

Sendo assim, foi criado o modelo *Tabletop Tangible User Interface Specification Model* (TTUI-SM) e também uma ferramenta de especificação *Tabletop Tangible User Interface Specification Model Tool* (TTUI-SMT) apresentadas ao longo deste capítulo.

TTUI-SMT é parte de uma ferramenta mais completa de projeto de interfaces *tabletop*, sendo desenvolvida no laboratório WINDIS, em colaboração com o laboratório COMPSI do Centro Universitário Eurípides de Marília, que integra o diagrama de elementos de interface com um recurso de criação de protótipo visual de interface *tabletop* em tempo real. Na Figura 33 é mostrada a visão geral desta

ferramenta. O quadrado vermelho representa a ferramenta TTUI-SMT desenvolvida como parte deste trabalho de mestrado.

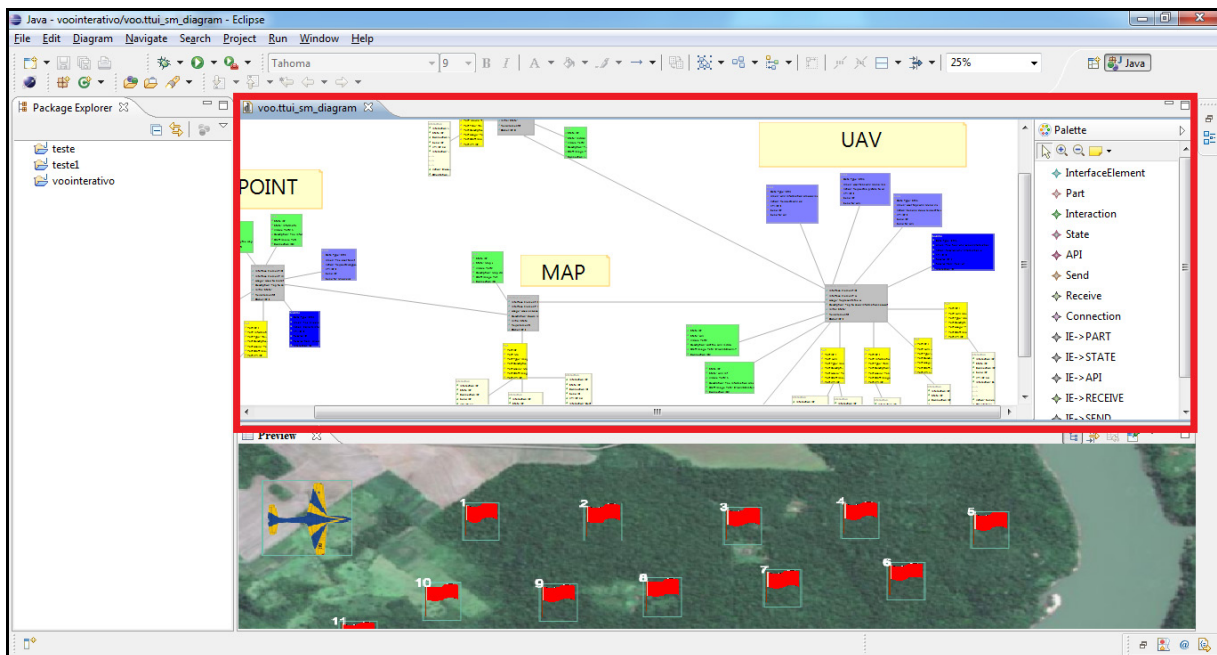


Figura 33 - Visão geral de ferramenta de criação de protótipo visual de interface tabletop em tempo real

5.1 Modelo de Especificação de Elementos de Interface TTUI-SM

O modelo de especificação de elementos de interface TTUI-SM é dirigido, nesta primeira versão, às interfaces tangíveis *tabletop* e seus modelos de interação. Este modelo é baseado nas linguagens descritivas como, por exemplo, e principalmente, o TUIML (SHAER; JACOB, 2009). Porém, diferentemente do TUIML, o modelo de especificação em desenvolvimento tem o foco voltado para os elementos da interface e tem o intuito de simplificar a maneira pela qual as propriedades destes elementos são organizadas no contexto das interfaces *tabletop*, além de contemplar a especificação das interações previstas neste contexto. Assim, o modelo TTUI-SM provê suporte para gestos multitoques avançados e customizados.

Este modelo é composto pelo Diagrama de Elemento de Interface e pela Ficha de Elemento de Interface, representados nas figuras 34 e 36, respectivamente, e descritos a seguir.

5.1.1 Diagrama de Elemento de Interface

O Diagrama de Elemento de Interface é uma representação hierárquica de qualquer possível elemento de interface de *tabletops* e de suas propriedades, que incluem tipos de interação. A partir deste diagrama é possível construir um modelo concreto de interação para cada elemento de interface de uma aplicação. Neste trabalho considera-se elemento de interface qualquer elemento gráfico que faça parte da interface da aplicação.

É importante destacar que o modelo permite flexibilidade na maneira de especificar os elementos de interface. Assim, um elemento gráfico pode ser considerado tanto um elemento de interface como *Part* de um elemento de interface (o conceito de *Part* será explicado a seguir). O projetista da interface pode decidir sobre os elementos de interface e suas propriedades no momento da especificação. Todos os elementos gráficos podem ser considerados como elementos de interface, o que torna a especificação mais simples, porém bastante extensa. Para tornar a especificação mais compacta alguns elementos gráficos podem ser descritos como *Part* de outros elementos de interface.

Tendo o diagrama da Figura 34 como representação de todos os possíveis tipos de elementos de interface e interações, o designer deverá iniciar o processo de especificação dos elementos de interface. Para isso, o designer utiliza a ferramenta de especificação TTUI-SMT, criada como parte deste trabalho de mestrado, e descrita ao longo do texto, para descrever cada elemento de interface, suas características e interações. A descrição dos elementos que compõe o modelo representado na Figura 34 é dada a seguir.

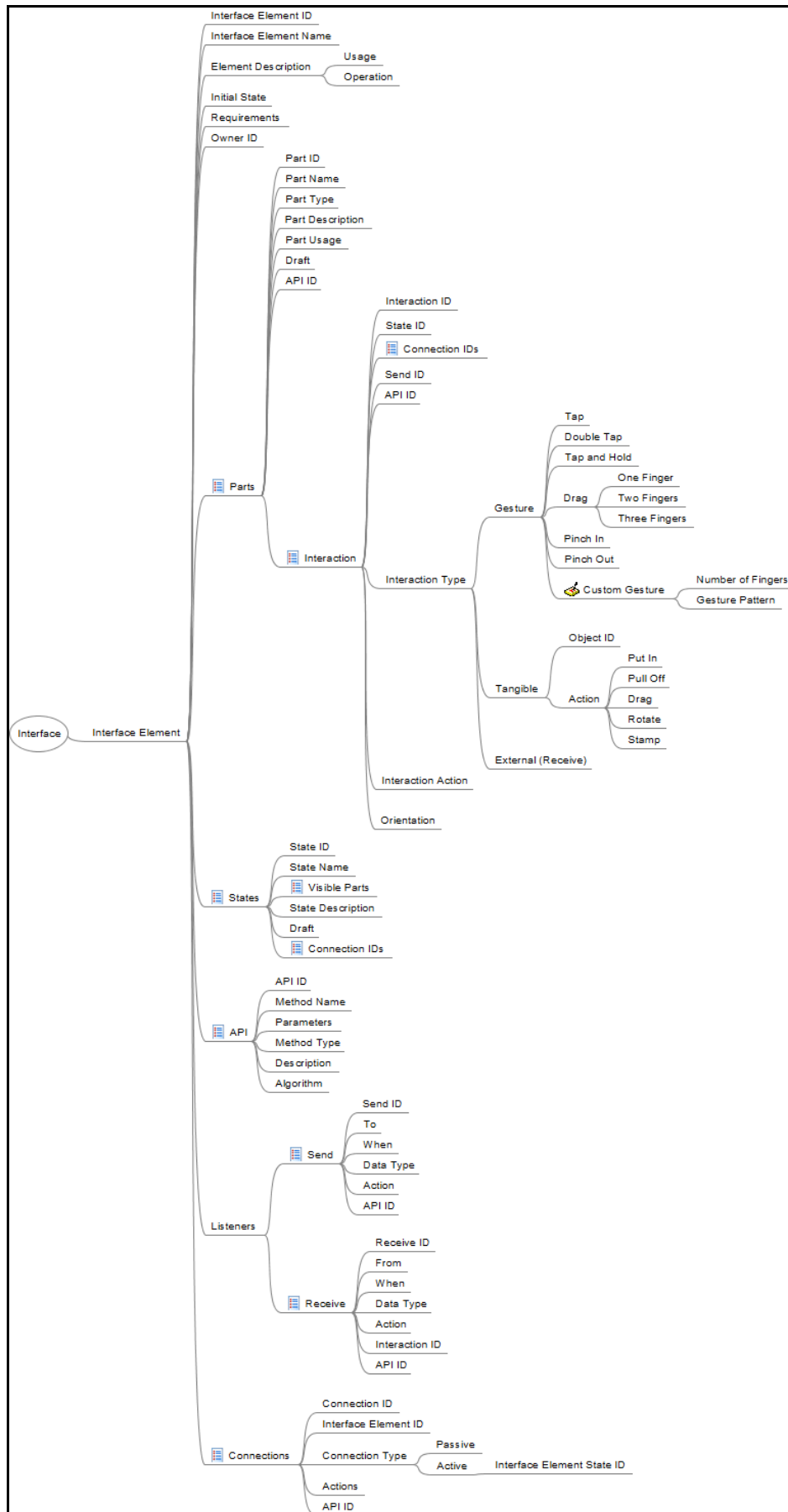


Figura 34 - Modelo Hierárquico de Elemento de Interface TTUI-SM

O modelo TTUI-SM é dividido em seis componentes distintos de especificação, conforme ilustrado na Figura 35. Cada componente é responsável por determinado aspecto do elemento de interface. Os componentes de um elemento de interface são: Dados Gerais, *Parts e Interactions*, *States*, *API*, *Listeners* e *Connections*.

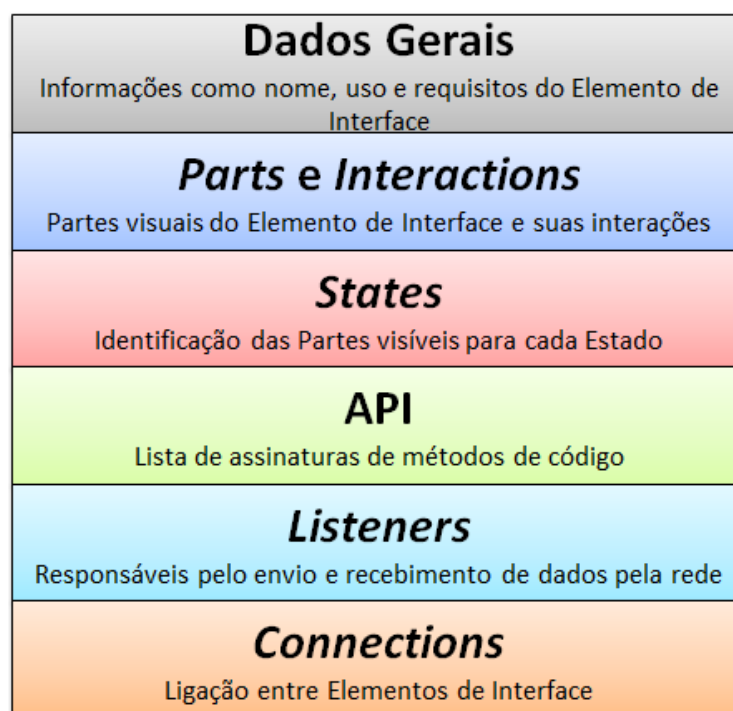


Figura 35 – Componentes de um elemento de interface

5.1.1.1 Dados Gerais do Elemento de Interface

No componente de dados gerais do elemento de interface, suas informações de identificação e uso são especificadas, bem como possíveis restrições. Os seguintes elementos compõem o componente de dados gerais:

- *Interface Element ID* (Identificação do Elemento de Interface) trata-se do identificador do elemento de interface em si.
- *Interface Element Name* (Nome do Elemento de Interface) trata-se do nome do elemento de interface em si.
- *Element Description* (Descrição do Elemento) descreve em linhas gerais a finalidade do elemento de interface. Este item é composto de dois elementos:
 - *Usage* (Uso) explica a forma na qual o elemento utilizado pode ser utilizado (onde, como e quando).

- *Operation* (Operação) diz respeito às capacidades do elemento de interface, ou seja, o que ele pode fazer.
- *Initial State* (Estado Inicial) indica qual é o estado inicial do elemento de interface.
- *Requirements* (Requerimentos) são condições que devem ser cumpridas para que seja possível a utilização do elemento de interface.
- *Owner ID* (Identificação do Proprietário) é um identificador único do proprietário (ou criador/instanciador) do elemento de interface.

5.1.1.2 *Parts* e *Interactions* do Elemento de Interface

Cada elemento de interface pode ser composto de diversas *Parts* distintas de visualização e cada *Part* pode ter diferentes tipos de interações (*Interactions*). Em função do estado (*state*) do elemento de interface, que muda de acordo com as *Interactions* a serem realizadas, *Parts* deste elemento podem ou não ser exibidas na interface em um determinado momento (detalhes sobre estados dos elementos de interface são apresentados na subseção 5.1.1.3).

Assim, define-se “*Parts*” do elemento de interface como sendo uma lista de informações que compõem o elemento de interface. Cada *Part* é composta de seis informações:

- *Part ID* (Identificação da Parte) e *Part Name* (Nome da Parte) são respectivamente o identificador único e o nome da *Part* propriamente dito.
- *Part Type* (Tipo da Parte) contém o tipo da *Part*, por exemplo, um botão.
- *Part Description* (Descrição da Parte) descreve as funções gerais da *Part*, bem como as condições para que a *Part* se torne visível.
- *Part Usage* (Uso da Parte) informa como é feito o uso da *Part* e qual sua utilidade.
- *Draft* (Rascunho) exibe um esboço gráfico da *Part*.
- *API ID* (Identificação de API) indica a API utilizada na codificação da *Part* (o conceito de API é apresentado na seção 5.1.1.4).
- *Interaction* (Interação) trata-se de uma lista de possíveis interações com a *Part*. A lista de interações é composta de oito elementos:
 - *Interaction ID* (Identificação da Interação) é um identificador único da interação em si

- *State ID* (Identificação de Estado) no caso da interação acarretar uma mudança de estado (Estados de elementos de interface são abordados na seção 5.1.1.3), ele deverá ser indicado neste campo.
- *Connection IDs* (Identificações de Conexão) no caso da interação realizar conexões (*Connections*) com outros elementos de interface, elas deverão ser indicadas neste campo (Conexão (*Connections*) entre elementos de interface são abordados na seção 5.1.1.6).
- *Send ID* (Identificação de Envio) no caso da interação resultar em envio de informações para meios externos (envio e recebimento de dados utilizando meios externos são abordados na seção 5.1.1.5), o *Send* deverá ser indicado neste campo.
- *Interaction Action* (Ação da Interação) é uma descrição interação
- *Orientation* (Orientação) relata a orientação na qual foi realizada a interação, por exemplo, um giro.
- *Interaction Type* (Tipo da Interação) pode ser dividida em três partes:
 - *Gesture* (Gesto) trata-se da interação com a interface por meio de toques. São previstas sete categorias de gestos:
 - *Tap* (Toque): Um simples toque na superfície.
 - *Double Tap* (Dois Toques): Dois toques rápidos na superfície.
 - *Tap and Hold* (Tocar e Segurar): Um toque sobre a superfície e manter o dedo sobre a mesma durante um tempo.
 - *Drag* (Arrastar): Três tipos de arrastos são previstos. Com um, dois e três dedos na superfície.
 - *Pinch In* (Pinça Aproximando): Movimento no qual um dedo é arrastado na direção a outro.
 - *Pinch Out* (Pinça Distanciando): Movimento no qual um dedo é arrastado na direção contrária ao outro.
 - *Custom Gesture* (Gesto Customizado): É possível especificar um gesto customizado da seguinte forma:
 - *Number of Fingers* (Número de Dedos): Quantidade de dedos envolvidos na interação.
 - *Pattern* (Padrão): Descrição do padrão do gesto.

- *Tangible* (Tangível) é o tipo de interação onde um objeto tangível é utilizado. O objeto é identificado a partir do marcador fiducial localizado na parte de baixo do objeto. Para a interação tangível, é necessário especificar o identificador do objeto e sua ação:
 - *Object ID* (Identificador de Objeto) informa o número de identificação do objeto.
 - *Action* (Ação) define qual interação será feita com o objeto. São previstas cinco categorias de interação tangível:
 - *Put in* (Colocar): O objeto tangível é colocado sobre a superfície.
 - *Put Out* (Retirar): O objeto tangível é retirado da superfície.
 - *Drag* (Arrastar): O objeto é arrastado sobre a superfície.
 - *Rotate* (Rotacionar): O objeto é virado (girado) sobre a superfície.
 - *Stamp* (Carimbar): O objeto é rapidamente colocado e retirado da superfície.
- *Remote* (Remoto) é uma interação realizada de forma automática quando o elemento de interface recebe dados externos.

5.1.1.3 States do Elemento de Interface

Um elemento de interface possui estados (*States*) distintos de visualização. Para cada *State* é descrito quais *Parts* do elemento de interface estarão visíveis quando tal *State* estiver ativo e quais *Connections* deverão ser realizadas quando tal *State* se tornar ativo. É composto de seis elementos de *State*:

- *State ID* (Identificação de Estado) e *State Name* (Nome do Estado) são respectivamente um identificador único e o nome do *State*.
- *Visible Parts* (Partes Visíveis) é uma lista de *Part ID* contendo quais *Parts* estarão visíveis para o *State*.
- *State Description* (Descrição do Estado) contém a descrição de quando (ou como) o *State* estará ativo.
- *Draft* (Rascunho) exibe um rascunho de como deverá se parecer o *State*.

- *Connection IDs* (Identificações de Conexão) lista quais *Connections* devem ocorrer quando o *State* estiver ativo.

5.1.1.4 API do Elemento de Interface

É possível especificar métodos que serão parte da implementação do elemento de interface utilizando o componente API. A API é uma lista de métodos que o elemento de interface pode usar. É onde as referências de código do elemento de interface são inseridas. É composta de seis elementos:

- *API ID* (Identificador da API) define o identificador do método.
- *Method Name* (Nome do método) trata-se do nome do método
- *Parameters* (Parâmetros) são os parâmetros do método, ex: "int a, boolean b, double c".
- *Method Type* (Tipo do Método) contém o tipo de retorno do método (int, string, boolean, etc.).
- *Description* (Descrição) é a descrição geral do que faz o método.
- *Algorithm* (Algoritmo) é um bloco onde é possível descrever (ou até mesmo escrever em código ou pseudocódigo) o algoritmo do método.

5.1.1.5 Listeners do Elemento de Interface

Para que seja possível a comunicação com meios externos, cada elemento de interface é passível de ter *Listeners* especificados de forma a descrever as ações de entrada e saída para o elemento de interface em relação a dispositivos externos. É composto de dois elementos, *Send* e *Receive*:

- *Receive* é responsável pelo recebimento de dados de meios externos. É composto de sete elementos:
 - *Receive ID* (Identificador de Recebimento): Define um identificador único para o recebimento especificado.
 - *From* (De): Informações sobre a origem dos dados recebidos.
 - *When* (Quando): Informações sobre quando os dados são recebidos.
 - *Data Type* (Tipo de Dado): Tipo de dado que está sendo recebido. Ex: Vídeo, Áudio, etc.
 - *Action* (Ação): A ação a ser tomada.

- *Interaction ID* (Identificador de Interação): Caso seja necessário realizar uma interação após um recebimento, ela será indicada neste campo.
- *API ID*: Definições de métodos para o *Receive*.
- *Send* é responsável pelo envio de dados para meios externos. É composto de seis elementos:
 - *Send ID* (Identificador de Envio): Define um identificador único para o envio especificado.
 - *When* (Quando): Informações temporais sobre o momento do envio dos dados.
 - *To* (Para): Informações sobre o destino dos dados enviados.
 - *Data Type* (Tipo): Tipo de dado que está sendo enviado.
 - *Action* (Ação): A ação a ser tomada.
 - *API ID*: Definições de métodos para o *Send*.

5.1.1.6 *Connections* do Elemento de Interface

Interações com elementos de interface podem fazer com estes se relacionem com outros elementos de interface podendo até alterar o seu *State*. Assim, Conexões (*Connections*) são as possíveis ligações dos elementos de interface uns com os outros e como o elemento alvo é afetado. É composto de cinco elementos:

- *Connection ID* (Identificador de Conexão) é o identificador único da *Connection*.
- *Interface Element ID* (Identificador de Elemento de Interface) é o identificador do elemento de interface no qual o atual elemento irá se conectar.
- *Actions* (Ações) fornece uma descrição das ações que a *Connection* fará.
- *Connection Type* (Tipo de Conexão) descreve o tipo de *Connection* a ser realizada com o outro elemento de interface. É composto de dois tipos:
 - *Passive* (Passiva): A *Connection* não terá nenhuma influência no comportamento do elemento de interface alvo.

- *Active* (Ativa): A *Connection* alterará o comportamento do elemento de interface alvo. O elemento *Interface Element State ID* indica qual *State* deverá se tornar ativo no elemento de interface alvo.
- *API ID*: Definições de métodos para a *Connection*.

5.1.2 Ficha de Elemento de Interface

Os dados mapeados de acordo com o diagrama de interface apresentado na seção 5.1.1 são organizados em uma Ficha de Elemento de Interface, que pode ser usada não apenas como guia de implementação, pelos projetistas de interface, mas também servir de documentação detalhada de toda a interface de uma aplicação. A ficha consiste em uma tabela única para cada elemento de interface na qual cada seção é descrita de forma detalhada. A Figura 36 exibe o *layout* da ficha de elemento de interface.

Interface Element: xxxxx									
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description					
1	xxxxx	1	1	Usage			Operation		
Requirements				xxxxx			xxxxx		
Parts									
Part ID	Part Name	Part Type	Part Description			Part Usage	API ID		
1	xxxxx	xxxxx	xxxxx			xxxxx	1		
Draft									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs		Interaction Action	Interaction Type	Object ID	Orientation
1	1		1			xxxxx	Gesture_Tap	-	xxxxx
States									
State ID	State Name	Visible Parts		Connection IDs		State Description			
1	xxxxx	1		1		xxxxx			
Draft									
API									
API ID	Method Name	Method Parameters			Method Type	Method Description			
1	xxxxx	xxxxx			xxxxx	xxxxx			
Algorithm									
xxxxx									
Listeners									
SEND	Send ID	To		When		Data Type	Action	API ID	
	1	xxxxx		xxxxx		xxxxx	xxxxx	1	
RECEIVE	Receive ID	From		When		Data Type	Action	API ID	
	1	xxxxx		xxxxx		xxxxx	xxxxx	1	
Connections									
Connection ID	Interface Element ID	Connection Type		Interface Element State ID	API ID	Action			
1	2	Active		1	1	xxxxx			

Figura 36 – Layout da Ficha de Elemento de Interface

De modo a facilitar o processo de especificação utilizando o modelo TTUI-SM, uma ferramenta foi desenvolvida para automatizar a especificação diagramática dos elementos de interface. Na próxima seção é descrita esta ferramenta.

5.2 Ferramenta de Especificação TTUI-SMT

A ferramenta de especificação *Tabletop Tangible User Interface Specification Model Tool* (TTUI-SMT) foi implementada com base no Diagrama de Elemento de Interface. Seu objetivo é oferecer ao projetista uma maior facilidade e rapidez na

tarefa de especificar os elementos de interface de uma aplicação. Assim, essa tarefa é feita arrastando-se os componentes para o diagrama e interligando-os de acordo com sua hierarquia.

Com a intenção de automatizar a especificação de elementos de interface, utilizando o modelo TTUI-SM e a ferramenta TTUI-SMT, dois recursos muito úteis foram implementados na ferramenta: a geração de fichas de elemento de interface e a geração das definições dos elementos de interface em um arquivo XML.

Para o desenvolvimento propriamente dito da ferramenta, foi utilizada a *Integrated Development Environment* (IDE) Eclipse 3.4.2 Ganymede (ECLIPSE, 2011) em conjunto com o pacote de ferramentas de modelagem. Assim, por meio do Eclipse *Graphical Modeling Framework* (GMF), diagramas baseados no *Eclipse Modeling Framework* (EMF) foram construídos e, a partir destes, a ferramenta foi implementada.

Para a geração de arquivos XML, foi utilizado um plugin *Model to Text* (Modelo para Texto) do Eclipse chamado *Java Emitter Template* (JET) que, utilizando-se de modelos pré-definidos, gera as fichas de elemento de interface juntamente com o arquivo de definição XML.

A Figura 40 mostra a interface da ferramenta. Detalhes sobre os recursos da ferramenta bem como a sua utilização serão explicados no decorrer desta seção.

5.2.1 Utilização da Ferramenta TTUI-SMT

O primeiro passo é a instalação da ferramenta utilizando a interface de desenvolvimento Eclipse. De posse dos plug-ins TTUI-SMT para Eclipse, eles devem ser colocados no diretório “**plugins**” que pode ser encontrado dentro da pasta raiz do eclipse. Após a instalação da ferramenta TTUI-SMT no Eclipse, é necessário criar o modelo TTUI-SM e em seguida realizar a especificação propriamente dita.

5.2.1.1 Criação de Novo Modelo TTUI-SM

Antes de criar um novo modelo de especificação TTUI-SM, é necessário criar um simples projeto Java no Eclipse. Uma vez criado o projeto, o modelo TTUI-SM deverá ser criado clicando em **File > New... > Other**. Quando a janela se abrir, deve-

se clicar em **Tabletop TUI Specification**, em seguida **TTUI-SM Model** e por fim clicar em **Next**. A Figura 37 exibe a escolha do projeto TTUI-SM.

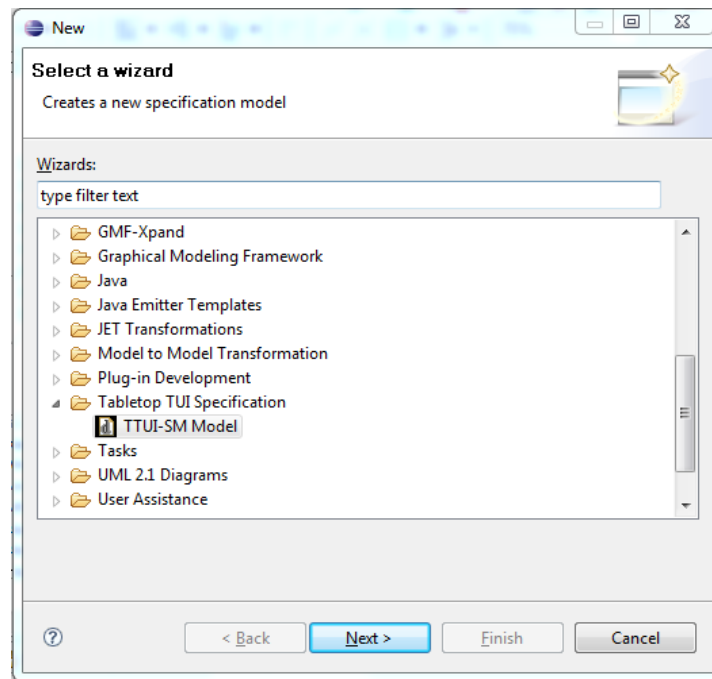


Figura 37 - Criação de um novo modelo TTUI-SM (Parte 1)

Após clicar em **Next**, o projeto Java que foi criado para receber o modelo TTUI-SM deverá ser selecionado e, em seguida, o diagrama deve ser nomeado de forma a conter a extensão “.ttui_sm_diagram”. Por fim, deve-se clicar novamente em **Next**. A Figura 38 exibe este passo.

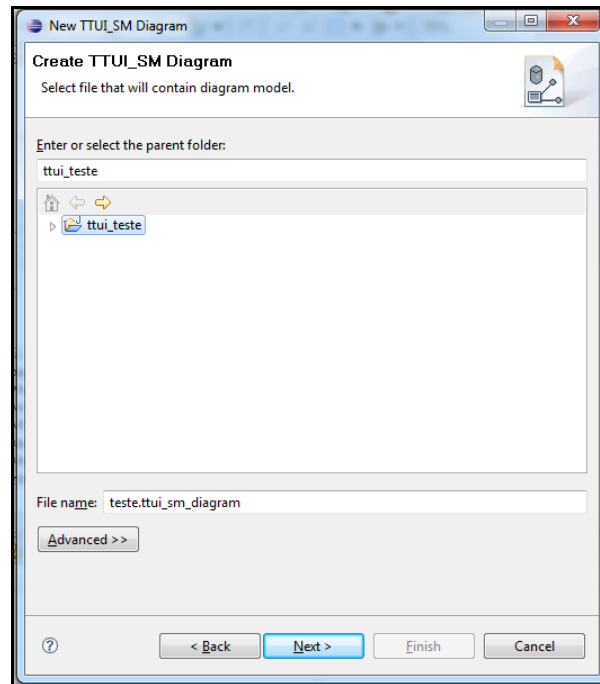


Figura 38 - Criação de um novo modelo TTUI-SM (parte 2)

Após clicar em *Next*, como exibido na Figura 38, deve-se realizar os mesmos passos anteriores, apenas respeitando a extensão do arquivo como “.ttui_sm”. Em seguida, o botão *Finish* deverá ser pressionado como exibe a Figura 39. Assim, o modelo TTUI-SM estará pronto para a especificação dos elementos de interface.

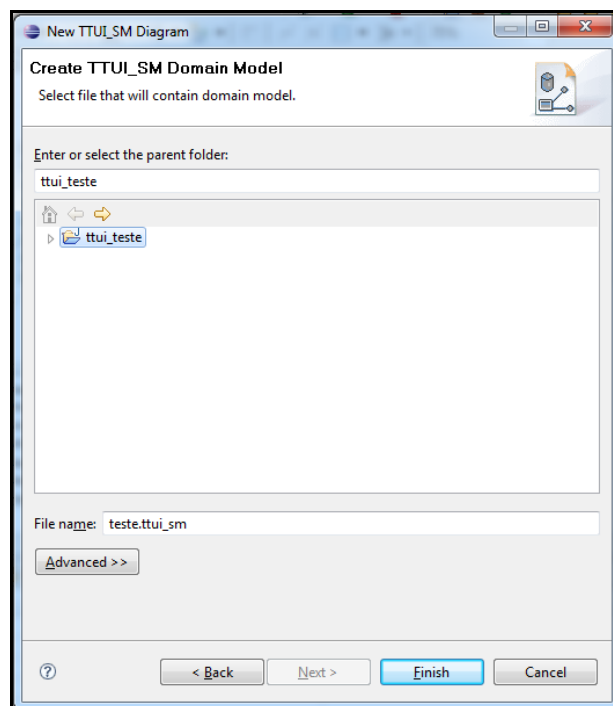


Figura 39 - Criação de um novo modelo TTUI-SM (Parte 3)

Uma vez que o modelo TTUI-SM esteja criado, a especificação deverá ser realizada dentro do arquivo <nome>.ttui_sm_diagram. Ao finalizar a criação do modelo TTUI-SM, ele será aberto automaticamente exibindo a interface de especificação de elementos de interface.

5.2.1.2 Interface de Especificação de Elementos de Interface

A interface de especificação de elementos de interface da ferramenta TTUI-SMT pode ser dividida em quatro principais áreas de acordo com a Figura 40.

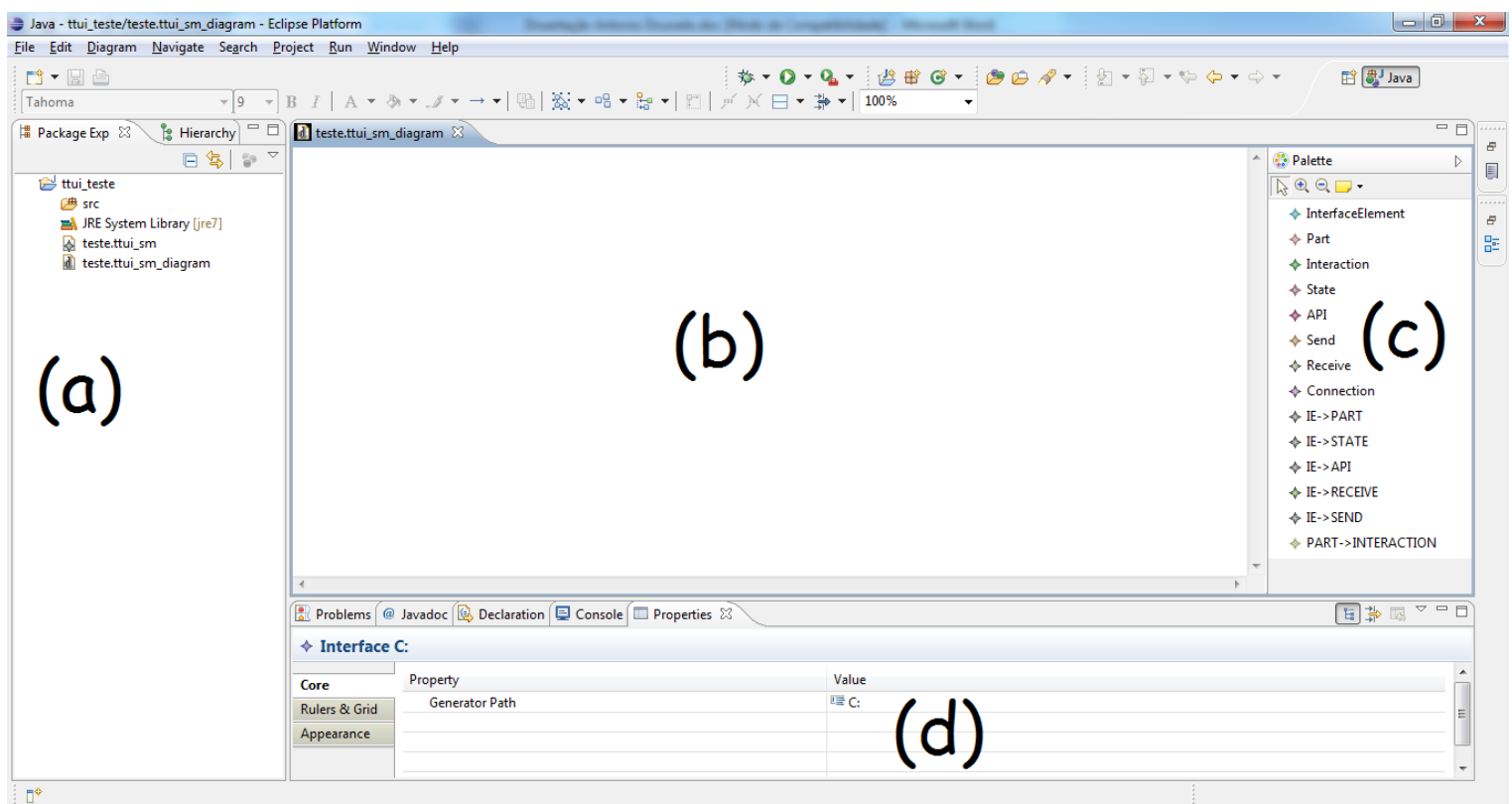


Figura 40 - Interface da Ferramenta TTUI-SMT: (a) Arquivos TTUI-SM (b) Área de Diagramação (c) Paleta de Elemento de Interface (d) Propriedades

A área de Arquivos do TTUI-SMT (Figura 40a) contém os dois principais arquivos do modelo TTUI-SM, sendo eles:

- <nome>.ttui_sm: Arquivo com os elementos especificados e suas definições de acordo com o modelo TTUI-SM.
- <nome>.ttui_sm_diagram: Arquivo onde a especificação diagramática propriamente dita dos elementos de interface ocorre.

A Área de Diagramação de Elementos de Interface (Figura 40b) está contida dentro do arquivo <nome>.ttui_sm_diagram e é onde a especificação diagramática é feita arrastando-se os componentes da paleta (Figura 40c) e interligando-os.

A Paleta de Elemento de Interface (Figura 40c) é composta de seções do Diagrama de Elemento de Interface apresentado na seção 5.1.1 deste capítulo e também de *Connections* entre cada seção do Elemento de Interface. A Figura 41 exibe a Paleta de Elemento de Interface.

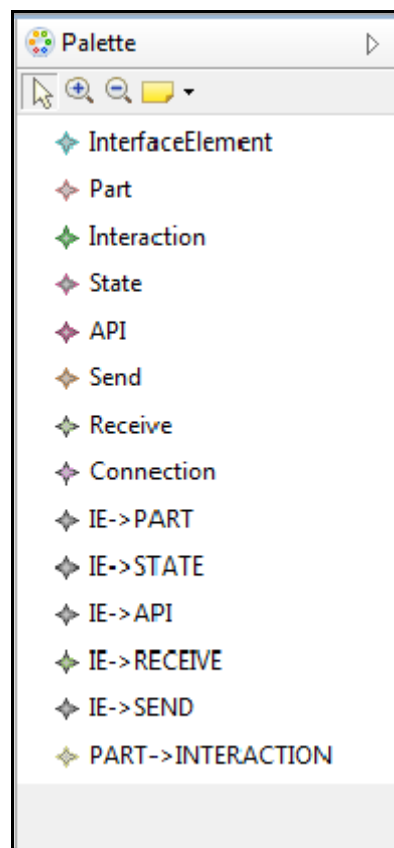


Figura 41 - Paleta de Elemento de Interface

Estes elementos se dividem em dois grupos: Nós de Elemento e Ligação de Elementos.

Os Nós de Elemento são os elementos que são diretamente arrastados para a área de diagramação e então uma caixa referente ao elemento é criada como mostra a Figura 42 ao clicar no elemento "*Interface Element*" e em seguida clicar sobre área de diagramação. Os nós de elemento da paleta são *Interface Element*, *Part*, *Interaction*, *State*, *API*, *Send* e *Receive*. As propriedades destes nós de elemento são as mesmas apresentadas na seção 5.1.1 deste capítulo.

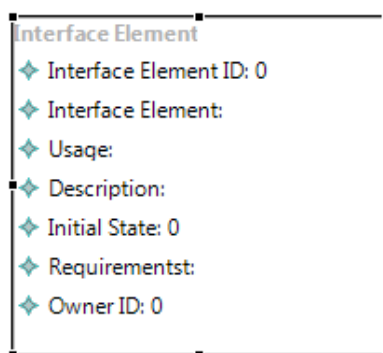


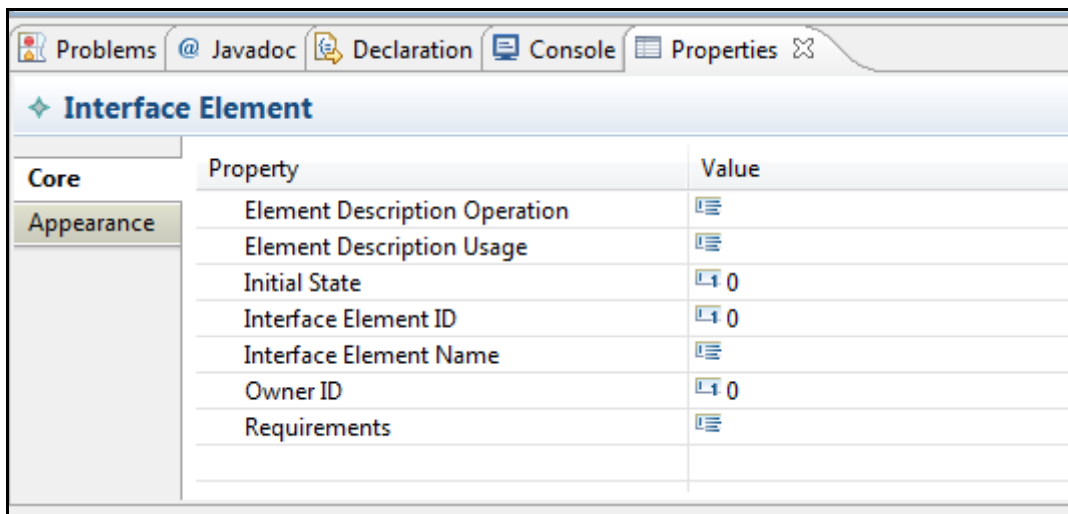
Figura 42 - Nó de Elemento "*Interface Element*"

Os Nós de Ligação de Elementos são responsáveis por interligar os Nós de Elemento, respeitando sua hierarquia, a fim de compor um Elemento de Interface completo com suas seções bem definidas. Para realizar a ligação entre os nós de elemento, é necessário selecionar o nó de ligação de elemento desejado, clicar sobre o nó de elemento origem e arrastar o mouse até o nó de elemento destino. Caso a ligação não respeite a hierarquia, o nó de ligação não será criado. Os nós de ligação de elementos são *Connection*, *IE->PART*, *IE->STATE*, *IE->API*, *IE->SEND*, *IE->RECEIVE* e *PART->INTERACTION*.

- CONNECTION: Realiza a ligação entre *Interface Element* e *Interface Element*. Embora este seja um nó de ligação de elementos, ele também possui propriedades (de acordo com a seção 5.1.1.6 deste capítulo) que podem ser editadas clicando-se sobre a linha formada por este nó.
- IE->PART: Realiza a ligação entre *Interface Element* e *Part*.
- IE->STATE: Realiza a ligação entre *Interface Element* e *State*.
- IE->API: Realiza a ligação entre *Interface Element* e *API*.
- IE->SEND: Realiza a ligação entre *Interface Element* e *Send*.
- IE->RECEIVE: Realiza a ligação entre *Interface Element* e *Receive*.
- PART->INTERACTION: Realiza a ligação entre *Part* e *Interaction*.

A área Propriedades (Figura 40d) da ferramenta é onde os dados de especificação deverão ser inseridos. Para tanto, deve-se clicar sobre o nó desejado na área de diagramação e então suas propriedades serão exibidas e disponibilizadas para edição. Os nós que podem ter suas propriedades editadas de acordo com a seção 5.1.1 deste capítulo são *Interface Element*, *Part*, *Interaction*,

State, *API*, *Send*, *Receive* e *Connection*. A Figura 43 exibe as propriedades do nó *Interface Element*.



Property	Value
Element Description Operation	
Element Description Usage	
Initial State	0
Interface Element ID	0
Interface Element Name	
Owner ID	0
Requirements	

Figura 43 - Propriedades do nó *Interface Element*

Apresentadas as quatro principais áreas da ferramenta TTUI-SMT, já é possível realizar a especificação diagramática dos elementos de interface de acordo com as propriedades propostas pelo Diagrama de Elemento de Interface da seção 5.1.1. A Figura 44 exibe um exemplo de especificação diagramática TTUI-SM de dois Elementos de Interface ligados por meio de uma *Connection* e cada elemento de interface tem uma *Part*.

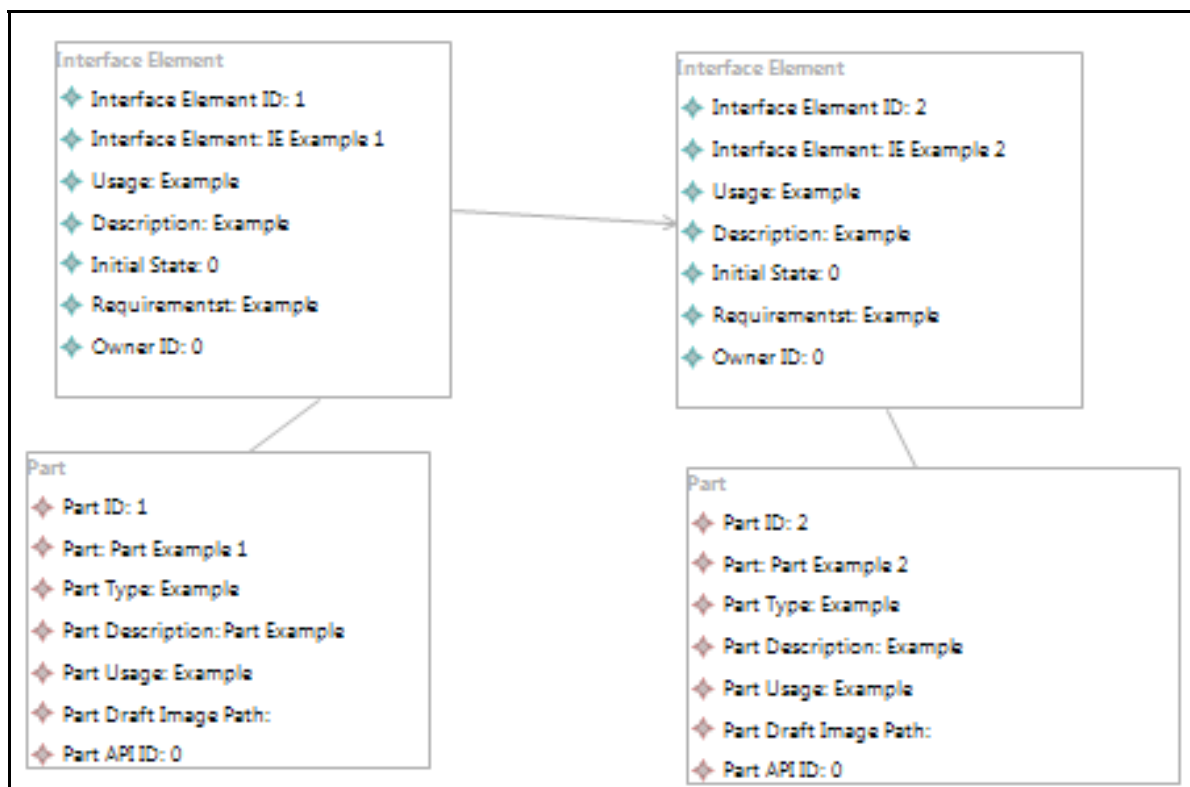


Figura 44 - Exemplo de Especificação Diagramática TTUI-SM

5.2.2 Geração de Ficha de Elemento de Interface e Definições XML

Com a finalidade de diminuir o tempo gasto com a especificação dos elementos de interface, a ferramenta foi implementada de forma a, além de oferecer maior agilidade na especificação, também gerar, de forma automática, as fichas de elemento de interface especificados, que servem como documentação da interface, além de gerar também arquivo XML contendo as definições de todos os elementos. Como parte dos trabalhos futuros (a serem desenvolvidos em parceria entre laboratório WINDIS e Laboratório COMPSI) será criado um *parser* que, a partir do arquivo XML, deverá gerar automaticamente os métodos e classes correspondentes para facilitar a implementação da interface. Na Figura 45 é mostrado o fluxo de geração de código a partir da especificação gerada pelo TTUI-SMT.

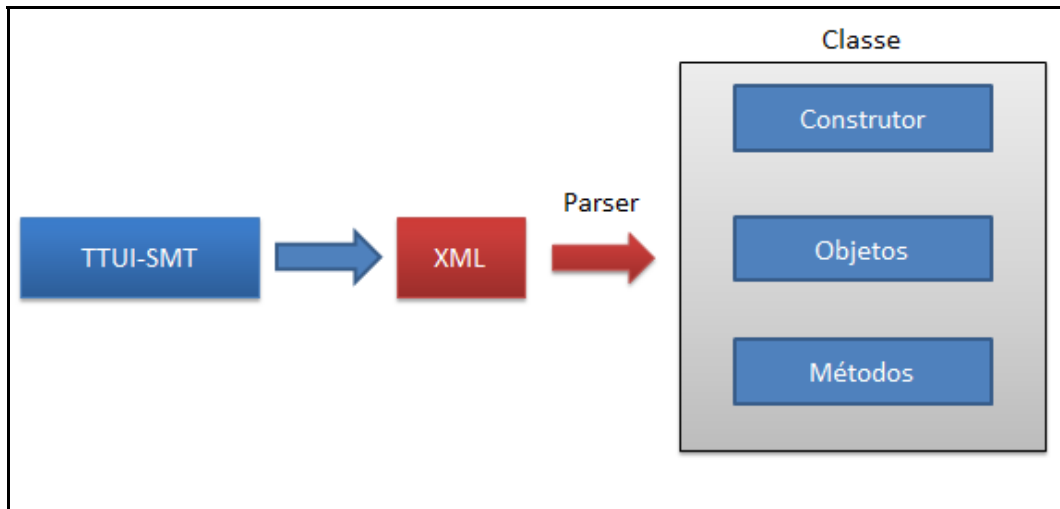


Figura 45 - Fluxo de geração de código a partir da especificação gerada pelo TTUI-SMT

Por padrão, o arquivo de definições XML será gerado no mesmo diretório da modelagem TTUI-SM e o arquivo de fichas de elemento de interface será gerado no diretório “C:\”. No caso de usuários de outros sistemas operacionais ou no caso de o arquivo de fichas de elemento de interface precisar ser gerado em outro local, será necessário alterar a propriedade “*Generator Path*” presente na área de Propriedades ao se clicar em qualquer espaço em branco na Área de Diagramação da Ferramenta TTUI-SMT. A Figura 46 exibe a propriedade. Caso o local especificado não exista, a geração do arquivo não funcionará.

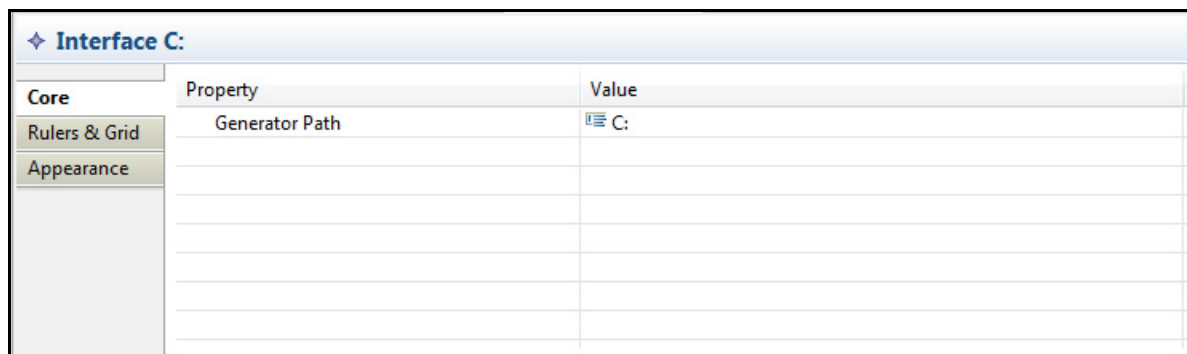


Figura 46 - Caminho de geração de arquivo de fichas de elemento de interface

Uma vez que o caminho esteja devidamente configurado e seja válido, depois de realizada a especificação diagramática dos elementos de interface de uma aplicação *tabletop*, pode-se então solicitar a geração do arquivo de fichas de elemento de interface juntamente com o arquivo de definições XML. Para isso, é necessário clicar com o botão direito do mouse sobre o arquivo <nome>.ttui_sm,

situado na área de Arquivos TTUI-SM e, uma vez que o menu apareça, a opção **“Generate Interface Element Record file and XML Definitions”** deverá ser selecionada como exhibe a Figura 47. O processo de geração pode demorar alguns instantes e, ao fim da geração dos arquivos, uma mensagem de confirmação será exibida.

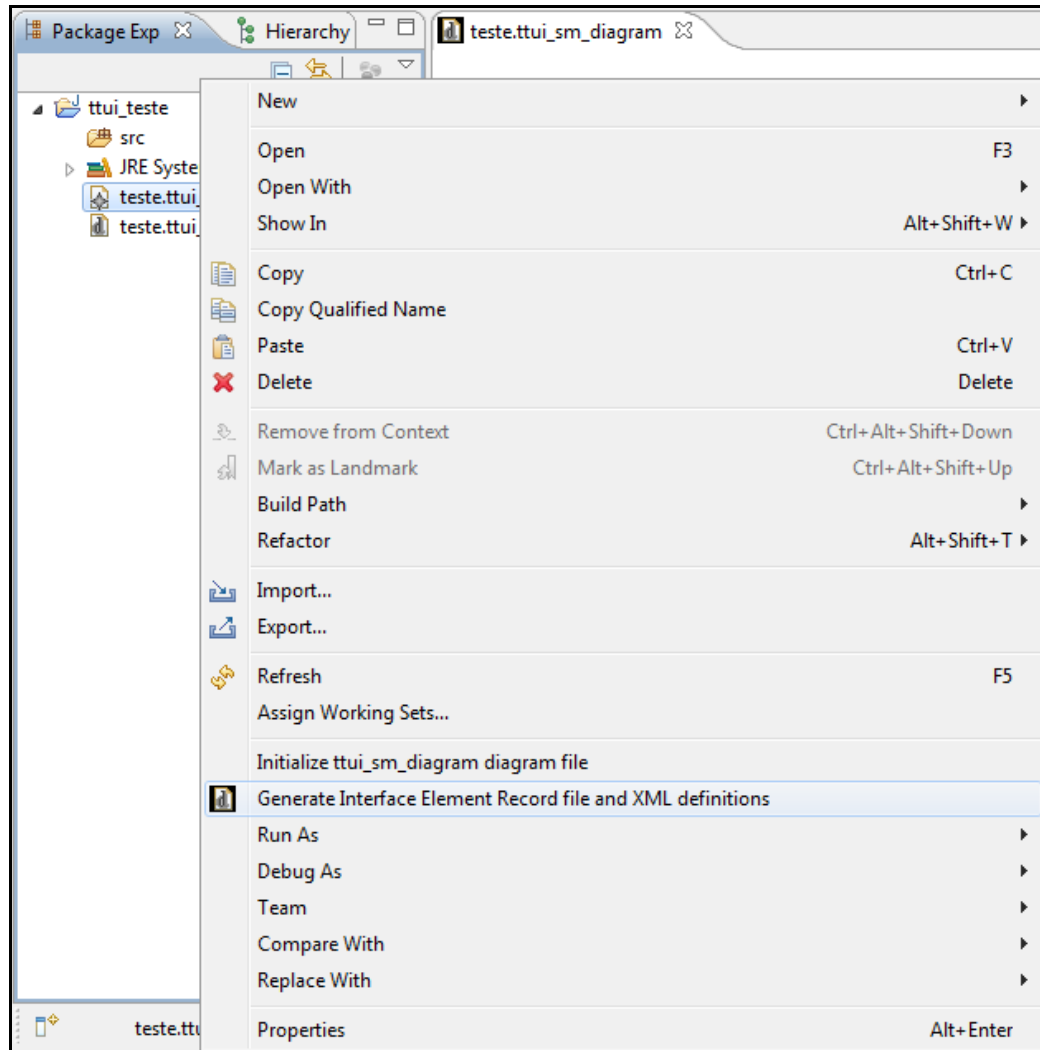


Figura 47 - Geração de arquivos de fichas de elemento de interface e definições XML

O arquivo de fichas de elemento de interface gerado será como o exibido na seção 5.1.2 deste capítulo e os elementos de interface estarão separados por páginas, ou seja, ao fim de um elemento de interface, o próximo se iniciará no começo da próxima página e assim sucessivamente.

Por ser um formato interoperável, o XML foi escolhido como formato padrão para conter as definições dos elementos de interface, uma vez que atualmente existe uma grande diversidade de frameworks implementados em várias linguagens

distintas com a finalidade de codificar aplicações *tabletop*, e todas essas linguagens têm a capacidade de interpretar arquivos XML, de forma a explorá-los para a obtenção dos dados especificados. Assim, optando por gerar um arquivo XML portando as definições dos elementos de interface, é possível o uso de qualquer framework de desenvolvimento *tabletop*. O *template* do arquivo XML com suas *tags* e propriedades pode ser encontrado no Anexo A.

A partir do componente API do elemento de interface, no momento da especificação pode-se ligar assinatura de métodos e links de arquivos com algoritmos a *Parts*, *Interactions*, *States*, *Listeners* e *Connections*. Assim, o desenvolvedor pode desenvolver um *parser* que interprete o arquivo XML e crie classes de código com métodos prontos para serem executados. A Figura 48 ilustra a ligação entre as seções de um elemento de interface com as seções de código após a aplicação de um *parser*.

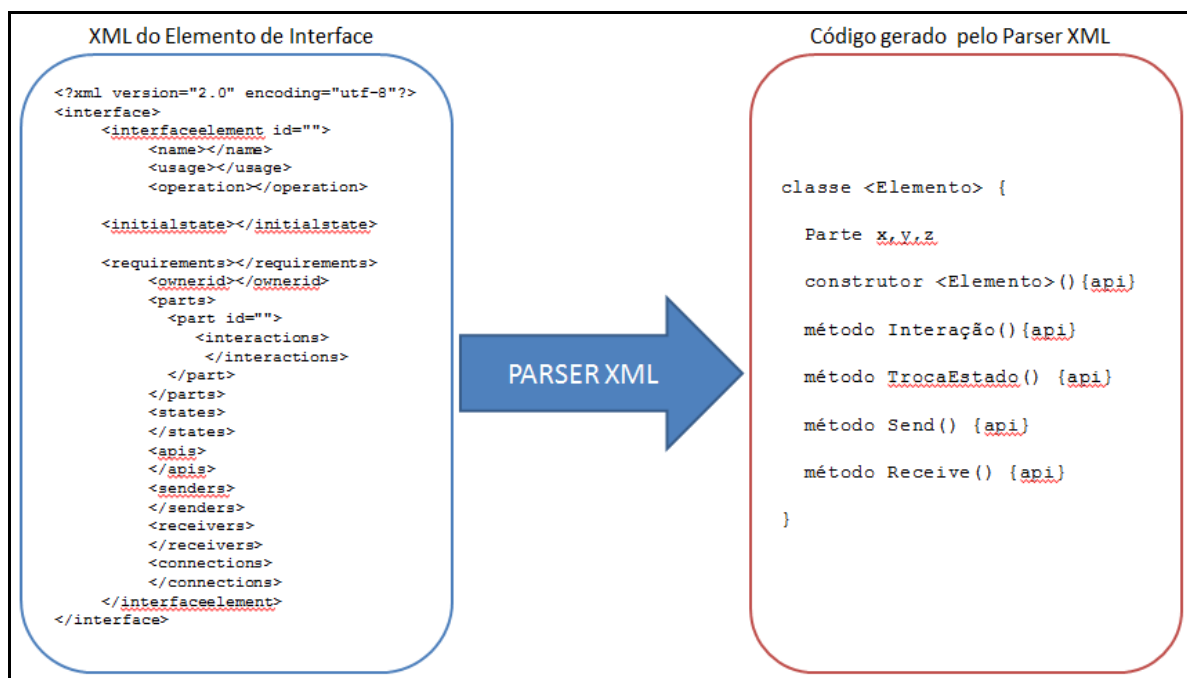


Figura 48 - Geração de Código por meio do XML

5.3 Modelo TTUI-SM e Metodologia de Desenvolvimento

Em termos de metodologia de desenvolvimento, o TTUI-SM é inserido nas fases de Design que são encontradas nos diversos modelos existentes. Dois desses modelos clássicos são o Modelo Cascata (PRESSMAN, 2001) e o Modelo Estrela (HARTSON; HIX, 1989), que foram apresentados na seção 3.2 deste trabalho. Estes modelos são modificados para mostrar como o TTUI-SM se integra nas metodologias de software.

No Modelo Cascata, o modelo TTUI-SM é inserido na fase “Design” recebendo os dados da análise, especificando os elementos de interface, gerando as fichas e o XML com as definições, e em seguida continuando para a fase de Código. A Figura 49 exhibe o Modelo Cascata original (Figura 49a) e o Modelo Cascata utilizando o TTUI-SM (Figura 49b).

Com a aplicação do modelo TTUI-SM no Modelo Estrela, as fases Prototipagem e Design Conceitual / Design Formal se tornam uma só, pois utilizando-se os *Drafts* presentes na especificação TTUI-SM, é possível realizar uma pequena prototipagem da interface. A Figura 50 exhibe o Modelo Estrela Original (Figura 50a) e o Modelo Estrela com o TTUI-SM (Figura 50b).

5.4 Considerações Finais

Neste capítulo, o modelo de especificação TTUI-SM de interfaces tangíveis *tabletop* foi apresentado com suas características e componentes. A ferramenta de especificação TTUI-SMT também foi apresentada como um recurso para agilizar a especificação de interfaces *tabletop*, que oferece a capacidade de geração automática das fichas de elemento de interface e também definição da especificação em arquivo XML. Por fim, foi demonstrado como o modelo TTUI-SM se insere em dois modelos clássicos de ciclo de vida de software: os modelos Cascata e Estrela.

Dois estudos de caso foram especificados para avaliação do modelo TTUI-SM, além de um experimento conduzido com 20 participantes também com a

finalidade de avaliar o modelo TTUI-SM e sua ferramenta. O próximo capítulo apresenta ambos os estudos de caso e também o experimento com seus resultados.

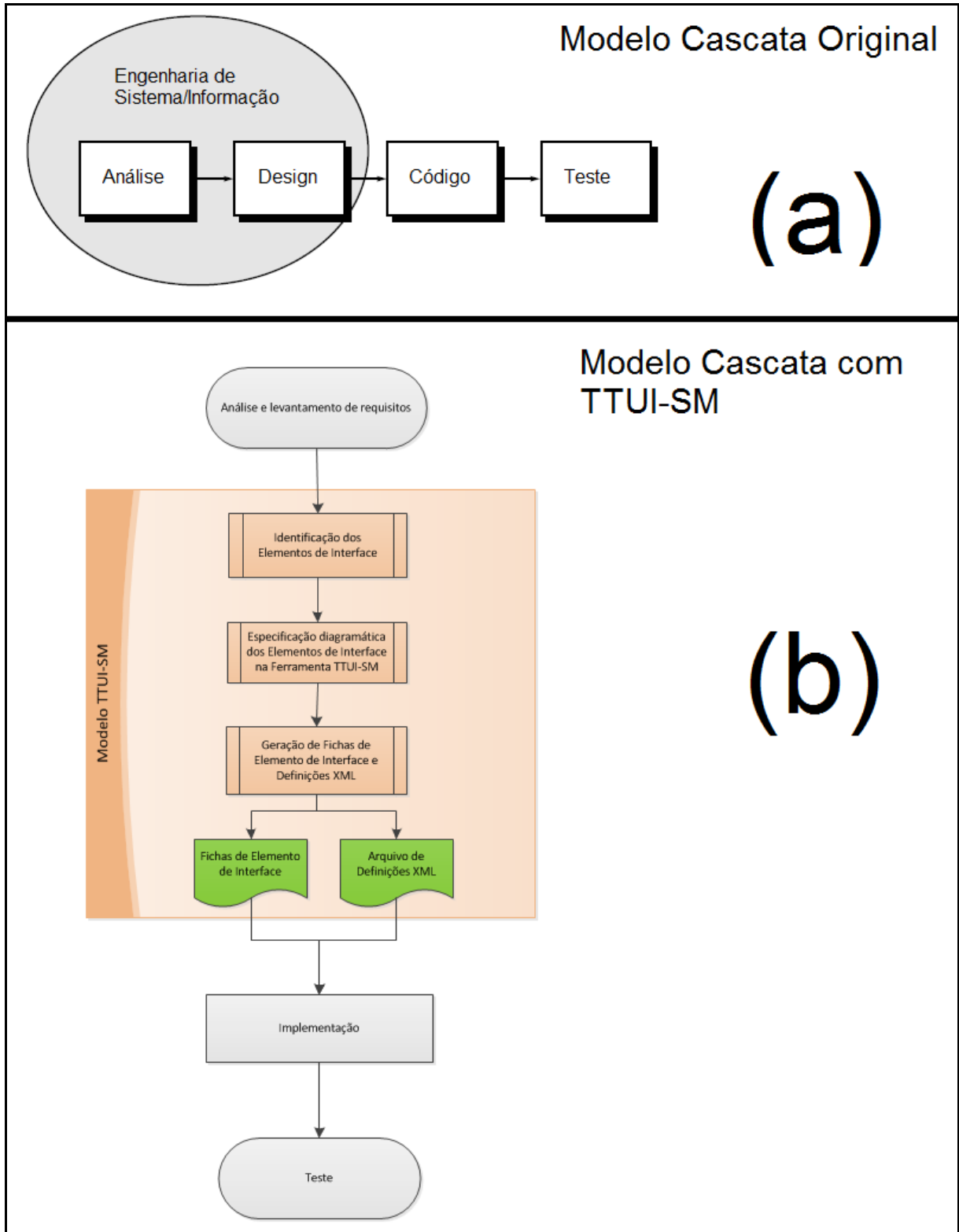


Figura 49 - (a) Modelo Cascata Original (b) Modelo Cascata com TTUI-SM (Adaptado de PRESSMAN, 2001)

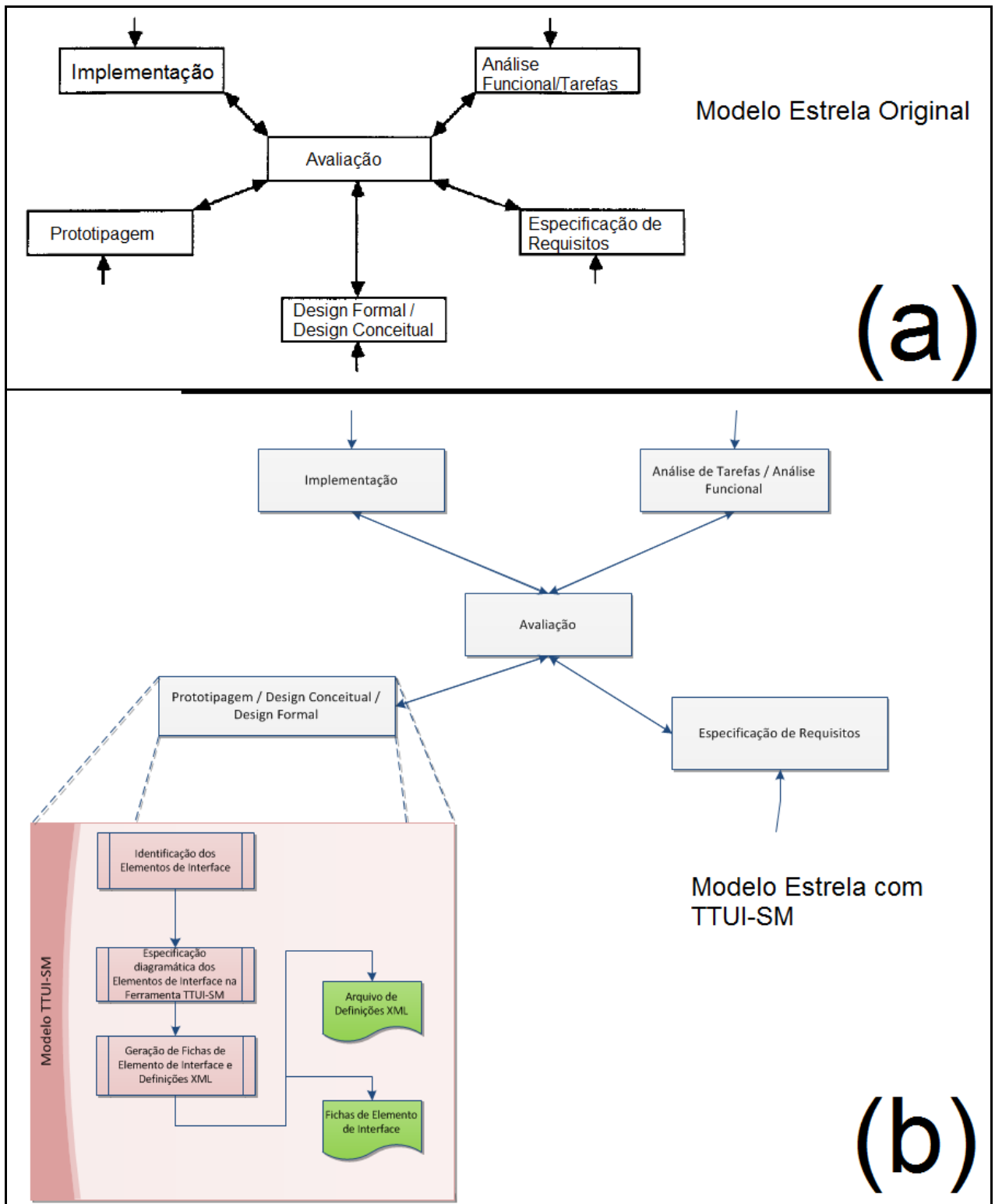


Figura 50 - (a) Modelo Estrela original (b) Modelo Estrela com TTUI-SM (Adaptado de HARTSON e HIX, 1989)

Capítulo 6

AVALIAÇÃO DO MODELO TTUI-SM

Visando demonstrar a utilização do modelo de especificação TTUI-SM juntamente com sua ferramenta de especificação, os estudos de caso “**Vigilância e Segurança de Infraestruturas Críticas**” e “**Voo Interativo**” terão parte de seus elementos de interface especificados utilizando a ferramenta TTUI-SMT. Estes estudos de caso fazem parte do Projeto Instituto Nacional de Ciência e Tecnologia em Sistemas Embarcados Críticos (INCT-SEC). Além dos estudos de caso, com o intuito de obter retorno de usuários e avaliar o modelo TTUI-SM, juntamente com sua ferramenta de especificação, foi conduzido um experimento envolvendo 20 alunos de diferentes níveis acadêmicos, entre iniciação científica e doutorado, consistindo da apresentação do modelo TTUI-SM e sua ferramenta, duas atividades práticas e um questionário.

6.1 Estudo de Caso: Vigilância e Segurança de Infraestruturas Críticas

Considera-se como uma infraestrutura crítica toda e qualquer estrutura ou edificação que, caso seja danificada, destruída, paralisada ou comprometida, poderá causar problemas para a sobrevivência de uma população local, regional ou nacional.

Visando a segurança deste tipo de estrutura, há um protocolo militar para operações de proteção de infraestruturas críticas, denominado “Protocolo Padrão”. O

Protocolo Padrão pode ser dividido em cinco áreas principais: Controle de Pessoal, Inteligência, Operações, Logística e Relações Públicas. Além disso, as tropas utilizadas neste tipo de operação são separadas em quatro grupos: Grupo de Comando, formado pelo comandante da operação e seus auxiliares; Grupo de Sentinelas, formado por soldados ocupando posições fixas e estratégicas dentro do perímetro; Grupo de Patrulha, formado por soldados que realizam patrulhas por rotas pré-determinadas dentro do perímetro; Força de Reação, formado por soldados que aguardam ordens para reagir em caso de invasão (BOSSONARO et al., 2011). A Figura 51 ilustra a organização básica do Protocolo Padrão.

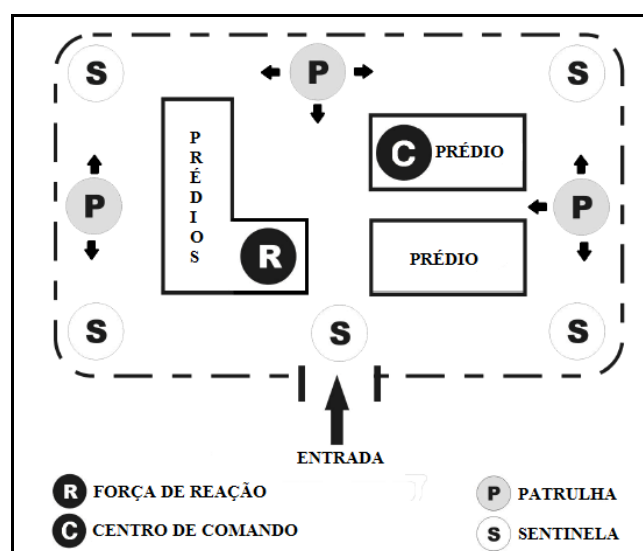


Figura 51 - Organização básica do protocolo padrão (Adaptado de BOSSONARO et al., 2011)

Com a intenção de potencializar o Protocolo Padrão, foi proposto o uso de alta tecnologia na missão de proteção das infraestruturas críticas (BOSSONARO et al., 2011). Os recursos tecnológicos propostos foram: Redes de Sensores Sem Fio (RSSF), Veículo Aéreo Não Tripulado (VANT), Robôs Táticos Internos (RTI), Identificação Rádio Frequência (RFID) e Interface de Comando e Controle, que trata-se de uma Interface Tangível *Tabletop*. A Figura 52 ilustra a organização do Protocolo Padrão potencializado.

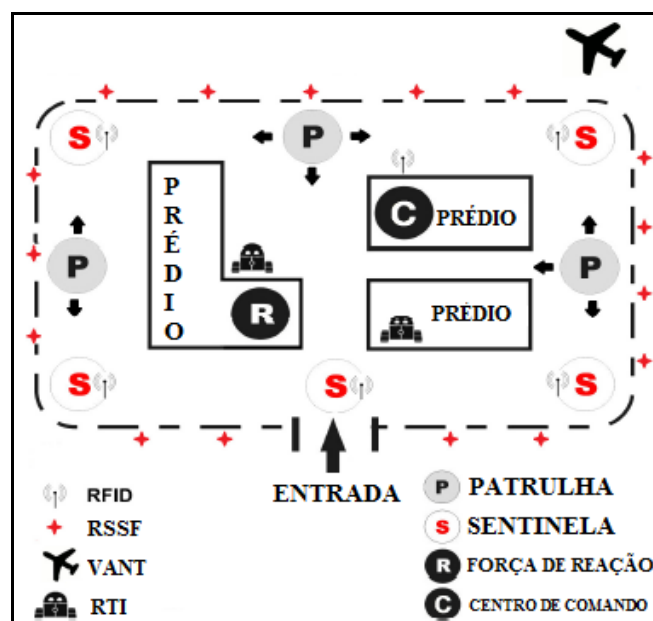


Figura 52 – Protocolo Padrão potencializado (Adaptado de BOSSONARO et al., 2011)

A Interface de Comando e Controle do Protocolo Padrão potencializado será especificada no modelo TTU-SM. Esta interface é responsável por exibir todas as informações provenientes de sensores sem fio, VANT, RFID, RA e Robôs Táticos depositados em um ambiente de infraestrutura crítica. A informação será exibida em camadas de visualização que podem ser sobrepostas, ou seja, é possível visualizar mais de uma camada simultaneamente. As camadas de visualização são:

- Topográfica: Visão apenas do mapa da região
- Controle de Pessoal: Visualização de informações relativas ao pessoal envolvido na operação.
- Inteligência: Visualização de alertas e ocorrências provenientes dos recursos tecnológicos internos e externos, por exemplo, o alerta de um sensor de movimento.
- Operações: Visualização da estrutura operacional do perímetro, ou seja, rotas de sentinelas, posicionamento de sensores e etc.
- Logística: Visualização dos recursos não tecnológicos da operação como helicópteros, viaturas, munições e etc.
- Configuração: Visualização de informações técnicas dos recursos tecnológicos agregados à operação.

Apresentadas as camadas de visualização, a Figura 53 ilustra um esboço da visualização das camadas de Controle de Pessoal (a), Inteligência(b), Operações (c) e Logística(d).

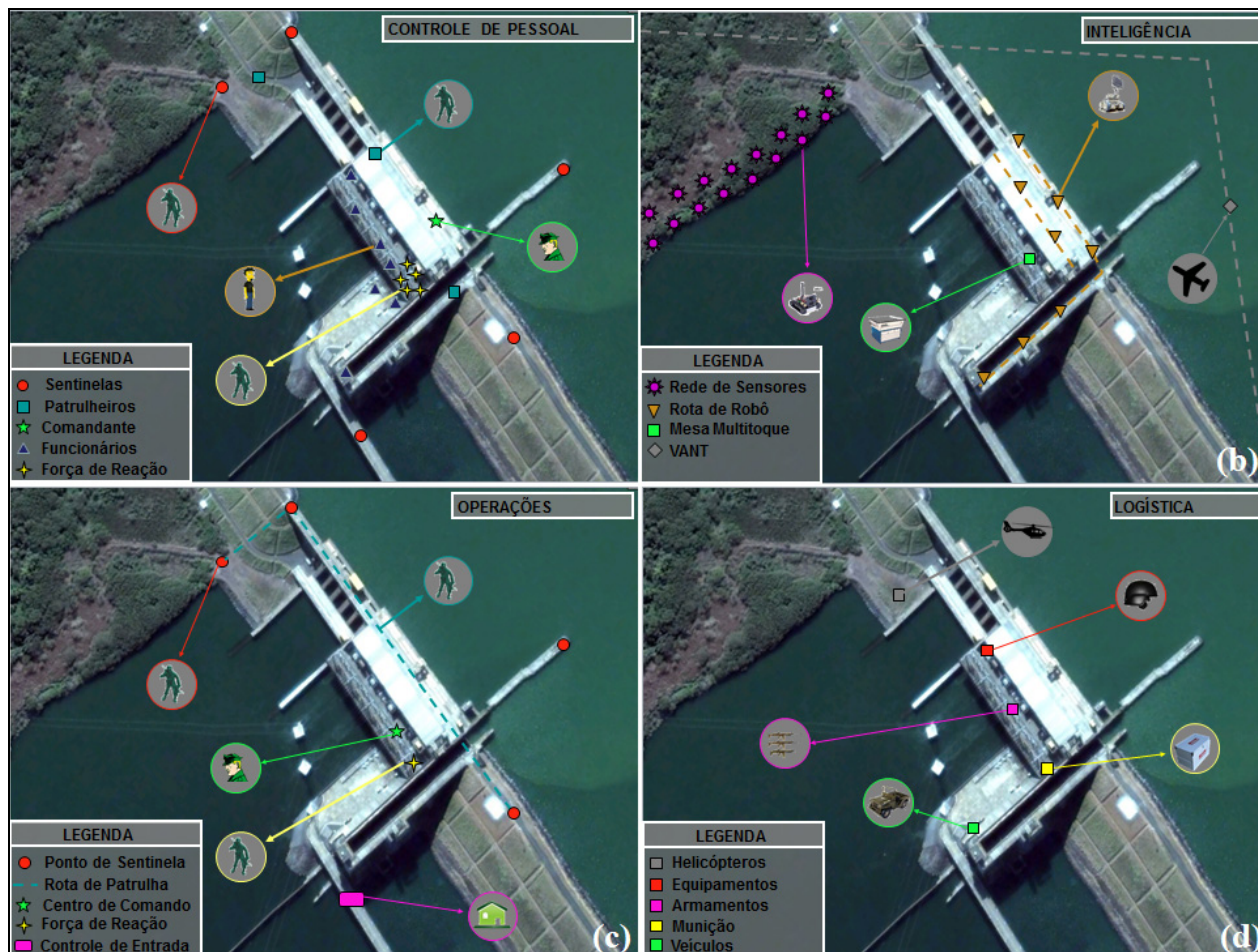


Figura 53 – Camadas de Visualização da Interface Avançada de Controle e Comando. (a) Controle de Pessoal (b) Inteligência (c) Operações (d) Logística.

6.1.1 Identificação dos Elementos de Interface

Antes de realizar a especificação dos elementos de interface, é necessário identificar quais são os pertencentes à aplicação a ser especificada e, se possível, suas *Parts*. Portanto, nesta seção os elementos de interface do estudo de caso Vigilância e Segurança de Infraestruturas Críticas serão identificados.

É importante ressaltar que as camadas também são consideradas como elementos de interface por serem containers de outros elementos de interface, como definido na seção 5.1.1 deste trabalho. Neste estudo de caso, apenas as camadas de Topografia, Controle de Pessoal, Inteligência e Operações foram especificadas.

6.1.1.1 Camada de Topografia

Elementos de Interface:

- **Camada de Topografia** contendo os botões de visualização de outras camadas e também o mapa topográfico como *Parts* do elemento de interface.

6.1.1.2 Camada de Controle de Pessoal

Elementos de Interface:

- **Camada de Controle de Pessoal** contendo a legenda da camada e *Connections* para os elementos de interface desta camada.
- **Sentinelas** representa o posicionamento das sentinelas.
- **Comandante** representa o posicionamento do comandante das operações.
- **Patrulheiros** representa o posicionamento dos patrulheiros.
- **Força de Reação** representa o posicionamento das forças de reação da operação.
- **Funcionários** representa o posicionamento dos funcionários.

Pode-se visualizar as informações dos elementos Sentinelas, Comandante, Patrulheiros, Força de Reação e Funcionários ao realizar um toque sobre o elemento de interface desejado.

6.1.1.3 Camada de Inteligência

- **Camada de Inteligência** contendo a legenda da camada e *Connections* para os elementos de interface desta camada.
- **Redes de Sensores** exibe onde estão posicionados os nós sensores da operação.
 - Ao perceber um evento, o nó sensor envia um sinal para a mesa de controle exibindo um sinal de alerta.
- **Rota de Robô** exibe os pontos por onde os robôs móveis deverão realizar sua patrulha.
 - Ao perceber um evento, o robô móvel envia um sinal para a mesa de controle exibindo um sinal de alerta.

- **Mesa Multitoques** (Interface de Comando e Controle) contendo a representação da mesa e sua janela de informações.
- **VANT** exibe o posicionamento atual do VANT.
 - Ao perceber um evento, o VANT envia um sinal para a mesa de controle exibindo um sinal de alerta.
 - Menu de solicitação de imagens do VANT pode ser aberto ao dar dois toques sobre o VANT.
 - Ao receber imagens do VANT, uma janela pop-up se abre para exibir as imagens.

Pode-se visualizar as informações dos elementos Redes de Sensores, Rota de Robô e VANT por meio de um toque sobre o elemento de interface.

6.1.1.4 Camada de Operações

- **Camada de Operação** contendo a legenda, representação dos pontos de sentinela, rotas de patrulha, centro de comando, força de reação e controle de entrada.

6.1.2 Especificação dos Elementos de Interface

Uma vez que os elementos de interface das camadas de Topografia, Controle de Pessoal, Inteligência e Operações foram identificados, a especificação dos elementos de interface pode ser iniciada.

Pelo fato de a camada de Topografia ser a camada base da aplicação, ou seja, todas as outras camadas são exibidas acima desta, a camada de topografia foi especificada de forma a ser um “nó raiz” com *Connections* de ativação e desativação das outras camadas. Desta forma, a visão geral da especificação diagramática ficou de acordo com a Figura 54.

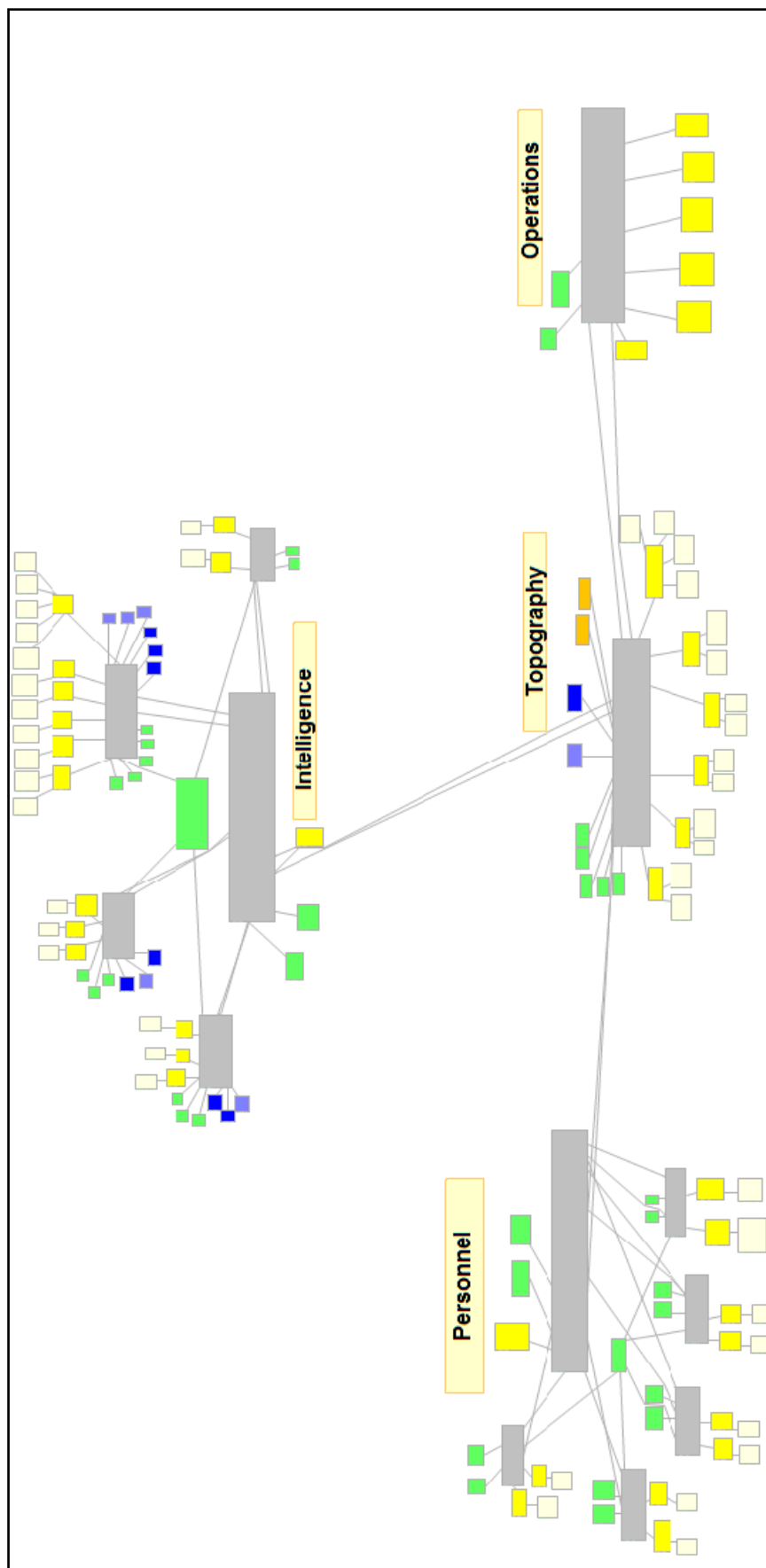


Figura 54 - Disposição do Diagrama de Especificação

Na Figura 54 não é possível visualizar os nós preenchidos com as especificações dos elementos de interface. Portanto, as Figuras 55, 56 e 57 exibem a *Part* “Media Frame” do Elemento de Interface “UAV” com suas quatro interações, o Elemento de Interface “Sink” (Mesa de Controle) com seus dois *States* e *Listeners* *Send* e *Receive* do Elemento de Interface “Sensor Network”, respectivamente.

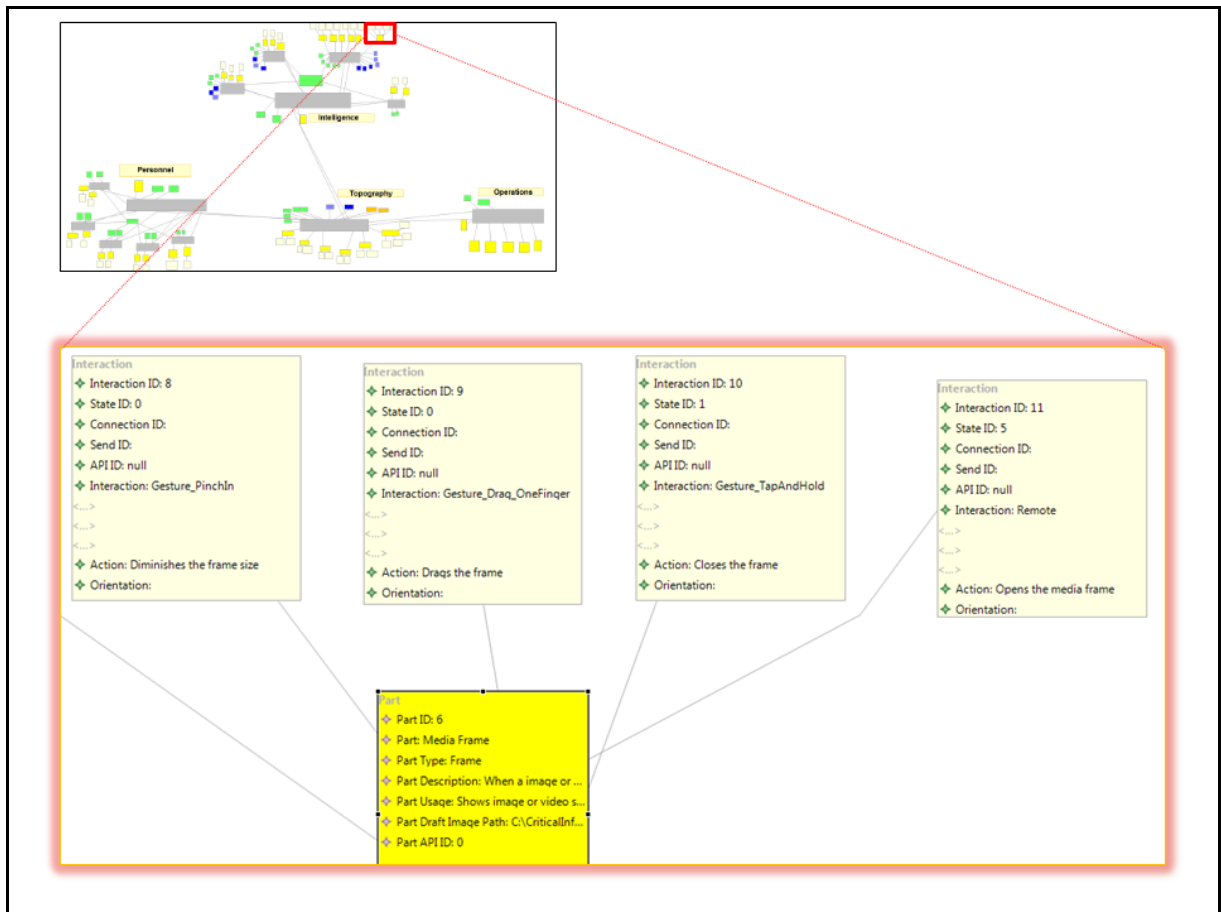


Figura 55 – *Part* “Media Frame” e suas interações

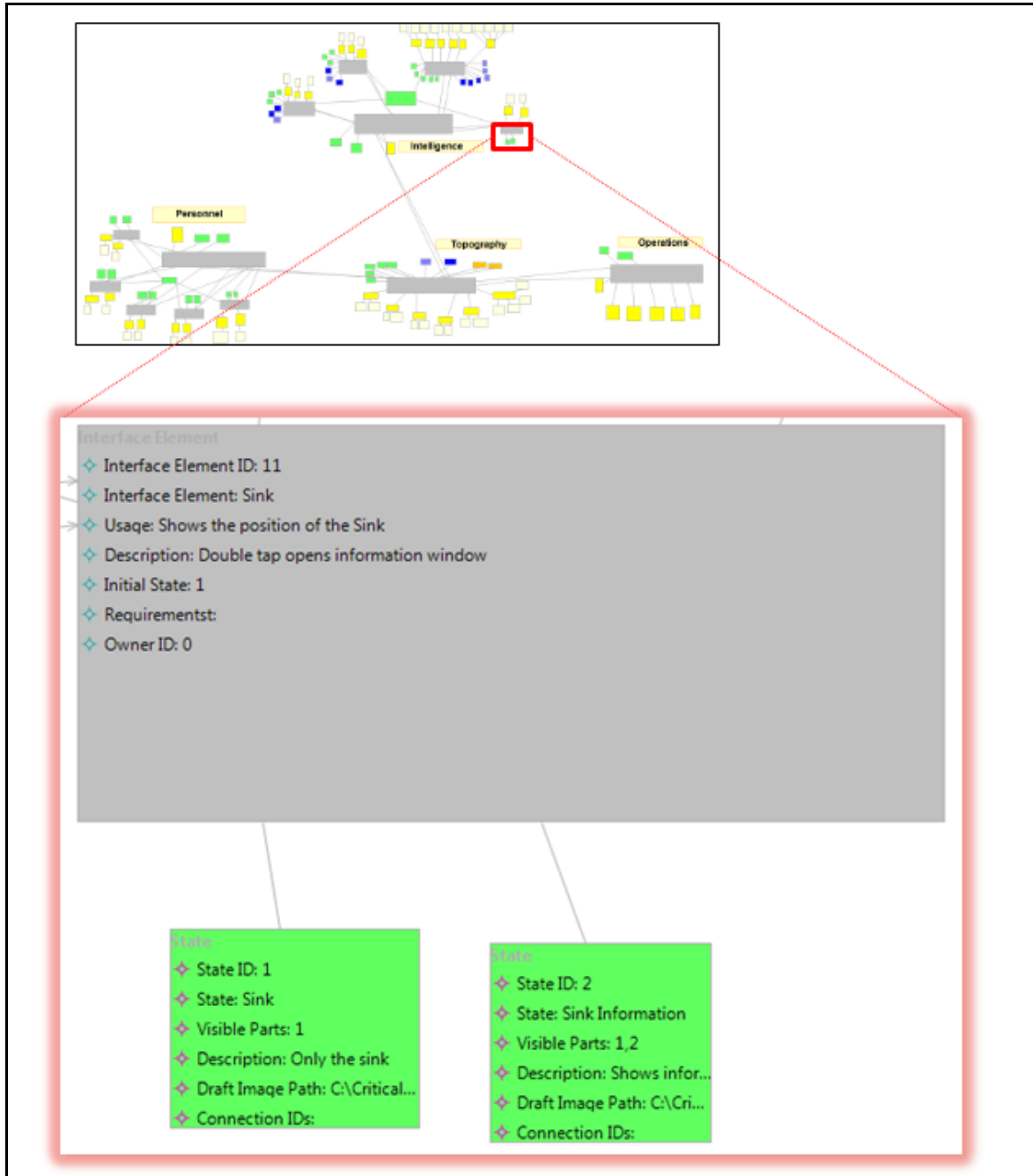


Figura 56 – Elemento de Interface “Sink” e seus States

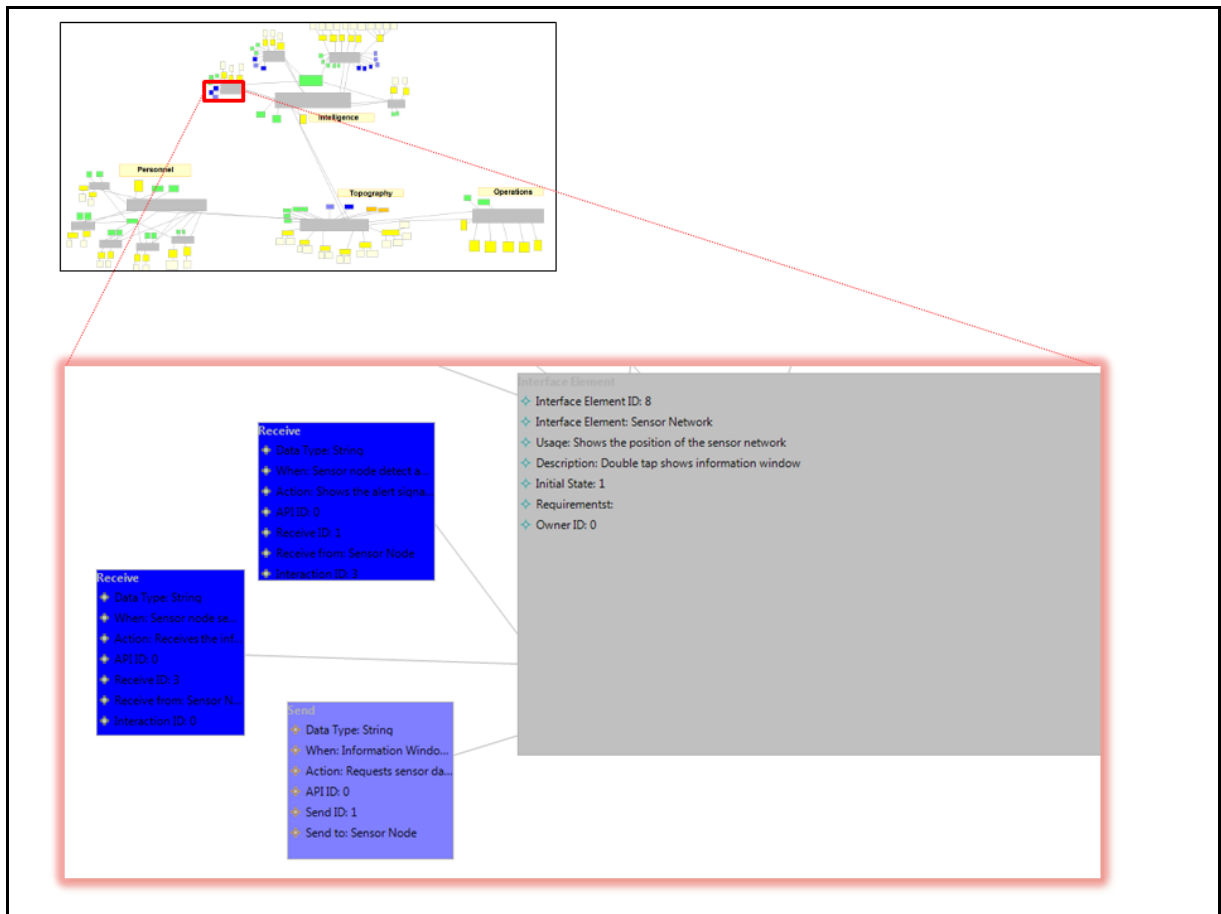



Figura 57 – Elemento de Interface “Sensor Network” e seus Listeners.

De acordo com a Figura 55, a *Part* “Media Frame” é especificada com suas quatro interações, sendo três gestos multitoques e uma interação remota. Essas interações destacam o potencial do modelo TTUI-SM em especificar gestos multitoques

A ficha gerada a partir da especificação do elemento de interface Topografia é exibida na figura 58.

Interface Element: Topography									
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description					
1	Topography	1	0	Usage			Operation		
Requirements				Used as topographic layer to exchange other layers			Possible to use buttons to switch layers and pan, zoom in/out the map		
Parts									
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
6	Map	Image	Shows the area map			Zoom and Pan		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
11	0				Loads Mission Data	Tangible_PutIn	1		
12	0				Pans the Map	Gesture_Drag_OneFinger	-		
13	0				Zooms the Map	Gesture_PinchIn	-		
14	0				Zooms out the Map	Gesture_PinchOut	-		
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
2	Personnel Button	Button	Button to exhibit Personnel Control Layer			Double Tap exhibits/hides Personnel Control layer. Tap and hold for a time activates Personnel Control Layer		0	
Draft									
<div style="border: 1px solid black; padding: 5px; display: inline-block;">Personnel Control</div>									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
3	0		2		Exhibit/Hide Personnel Control Layer	Gesture_DoubleTap	-		
4	0		1		Activate Personnel Control Layer	Gesture_TapAndHold	-		
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
1	Map Button	Button	Button that exhibit the topography layer (map)			Double tap hides all layers. Tap and hold for a time, deactivate and hide them.		0	
Draft									
<div style="border: 1px solid black; padding: 5px; display: inline-block;">Topography</div>									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
1	0		2		Deactivates and hides all layers	Gesture_TapAndHold	-		
2	0		1		Stop exhibiting all layers	Gesture_DoubleTap	-		
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	

3	Inteligente Button	Button	Button to exhibit Intelligence Layer	Double Tap exhibits/hides Intelligence layer. Tap and hold for a time activates Intelligence Layer	0				
Draft									
Intelligence									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
5	0		2		Exhibit/Hide Intelligence Layer	Gesture_DoubleTap	-		
6	3		1		Activate Intelligence Layer	Gesture_TapAndHold	-		
Part ID	Part Name	Part Type	Part Description	Part Usage	API ID				
4	Operations Button	Button	Button to exhibit Operations Layer	Double Tap exhibits/hides Operations layer. Tap and hold for a time activates Operations Layer	0				
Draft									
Operations									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
7	0		2		Exhibit/Hide Operations Layer	Gesture_DoubleTap	-		
8	0		1		Activate Operations Layer	Gesture_TapAndHold	-		
Part ID	Part Name	Part Type	Part Description	Part Usage	API ID				
5	Logistics Button	Button	Button to exhibit Logistics Layer	Double Tap exhibits/hides Logistics layer. Tap and hold for a time activates Logistics Layer	0				
Draft									
Logistics									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
9	0		2		Exhibit/Hide Logistics Layer	Gesture_DoubleTap	-		
10	0		1		Activate Logistics Layer	Gesture_TapAndHold	-		
States									
State ID	State Name	Visible Parts	Connection IDs	State Description					
1	Disabled Buttons	1,2,3,4,5,6		All layer buttons disabled					
Draft									
State ID	State Name	Visible Parts	Connection IDs	State Description					
2	Personnel Active	1,2,3,4,5,6		Personnel Control Layer Active					
Draft									
State ID	State Name	Visible Parts	Connection IDs	State Description					
3	Intelligence Active	1,2,3,4,5,6		Intelligence Layer Active					
Draft									
State ID	State Name	Visible Parts	Connection IDs	State Description					
4	Operations Active	1,2,3,4,5,6		Operations Layer Active					
Draft									
State ID	State Name	Visible Parts	Connection IDs	State Description					
5	Logistics Active	1,2,3,4,5,6		Logistic Layer Active					
Draft									

API						
API ID	Method Name	Method Parameters	Method Type	Method Description		
1	switchLayer	int IID	void	Switch Active Layers		
Algorithm						
API ID	Method Name	Method Parameters	Method Type	Method Description		
2	toggleLayer	int IID	void	Exhibits/Hides a Layer		
Algorithm						
Listeners						
SEND	Send ID	To	When	Data Type	Action	API ID
	1	Middleware	When load data tangible object is placed	String	Requests application data	0
RECEIVE	Receive ID	From	When	Data Type	Action	API ID
	1	Middleware	When middleware sends data	AbstractObject	Receives application data	0
Connections						
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action	
1	2	Active	1	0	Enables personnel control layer	
2	2	Active	1000	0	Disables personnel control layer	
3	7	Active	1	0	Exhibits Intelligence Layer	
4	7	Active	1000	0	Hides Intelligence Layer	
5	12	Active	1	0	Enable Operations Layer	
6	12	Active	1000	0	Disable operations layer	

Figura 58 - Ficha do Elemento de Interface Topografia

A partir da ficha de elemento de interface, é possível observar que a especificação é dividida em componentes distintos, facilitando a visualização das propriedades.

As demais fichas dos elementos de interface especificados para este estudo de caso são encontradas no Anexo B.

6.2 Estudo de Caso: Voo Interativo

A aplicação Voo Interativo tem como objetivo fornecer uma interface de visualização interativa para dar suporte às missões dos Veículos Aéreos Não Tripulados (VANTs) desenvolvidos pelo grupo GT3 – Sistemas Aéreos Não Tripulados que faz parte do Instituto Nacional de Ciência e Tecnologia em Sistemas Embarcados Críticos (INCT-SEC).

Os VANTs são utilizados com diversas finalidades. Em geral, sua utilização tem funções militares para, por exemplo, vigilância. Além disso, existem aplicações

não militares para os VANTS, podendo ser citada a manutenção de infraestruturas (EISENBEISS, 2004).

Neste estudo de caso, os VANTS serão utilizados para realizar o monitoramento aéreo de qualquer tipo de área, seja um rio, um campo, uma fronteira. Contarão com *Waypoints*, pontos que compõe sua rota de forma que o usuário poderá distribuir *waypoints* sobre o mapa da região. Por meio de um protocolo de comunicação, os VANTS trocarão informações com a aplicação e suas posições serão atualizadas constantemente bem como as informações sobre waypoints que lhe serão enviadas. Paralelamente, poderão enviar dados oriundos de seus payloads (como câmera e sensores) para a aplicação e a mesa de comando (interface tangível *tabletop* desenvolvida no laboratório WINDIS) tratará e exibirá essas informações. A Figura 59 exibe um esboço da interface deste estudo de caso.

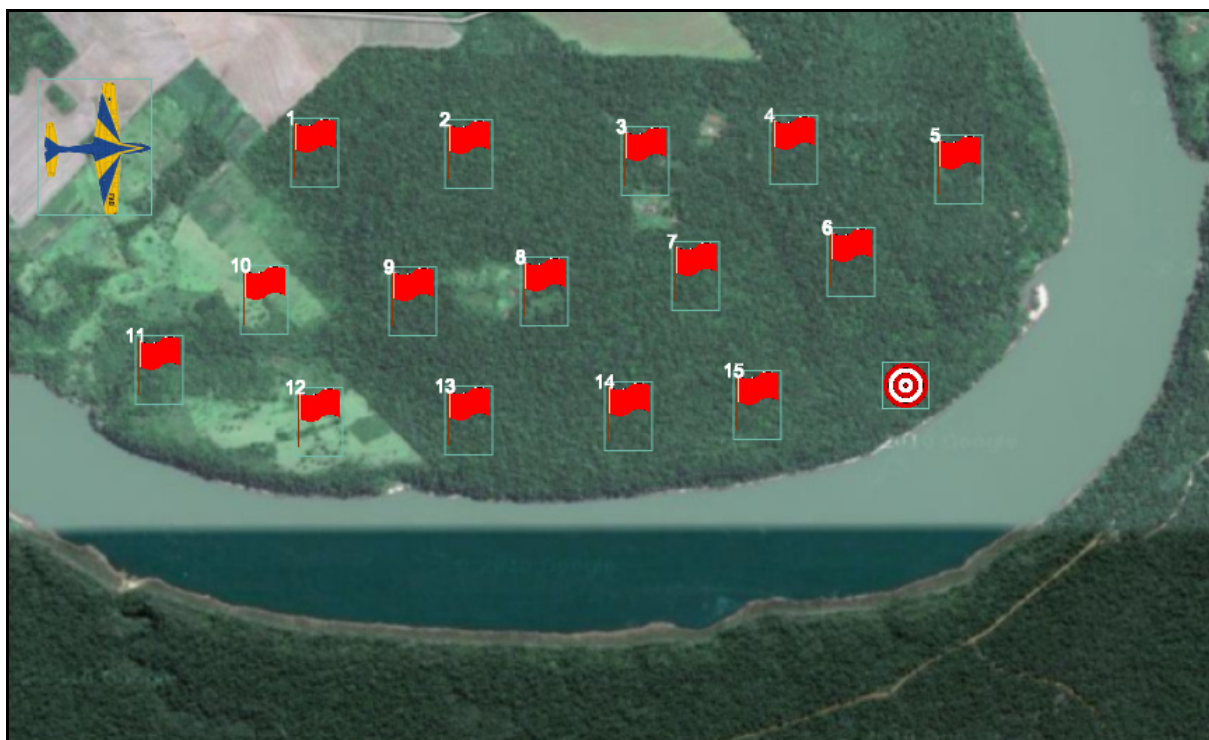


Figura 59 - Esboço da Interface Voo Interativo

6.2.1 Identificação dos Elementos de Interface

Assim como na seção 6.1.1 deste capítulo, os elementos deste estudo de caso também serão especificados de acordo com modelo TTUI-SM. Por se tratar de uma interface mais simples, quando comparada à interface do estudo de caso anterior, esta especificação contará com poucos elementos de interface.

6.2.1.1 Elemento de Interface Mapa (*Map*)

O elemento de interface mapa é responsável por exibir a região geográfica na qual o VANT irá realizar sua missão. Este elemento de interface conta com apenas uma *Part*, que é a representação visual do Mapa.

6.2.1.2 Elemento de Interface VANT (*UAV*)

O elemento de interface VANT representa no mapa o posicionamento atual do VANT real. Ele conta com quatro *Parts*: a imagem do VANT, uma janela de informações sobre o VANT e duas opções de menu, sendo a primeira para solicitar que o VANT tire uma foto e outra para solicitar o envio de imagens.

6.2.1.3 Elemento de Interface Waypoint

O elemento de interface Waypoint exibe um ponto no mapa por onde o VANT deverá passar durante a sua rota. Possui duas *Parts* que são a imagem do waypoint e sua janela de informações.

6.2.1.4 Elemento de Interface Frame de Mídia (*MEDIA FRAME*)

O elemento de interface Frame de Mídia é responsável por receber fotos e imagens do VANT e os exibir. Ele possui uma *Part*, sendo esta o frame que se abrirá ao receber tais mídias.

6.2.2 Especificação dos Elementos de Interface

Com os elementos de interface do estudo de caso “Voo Interativo” identificados, foi possível então realizar a especificação dos mesmos utilizando a ferramenta TTUI-SM. Assim, a Figura 60 exibe a visão geral do diagrama construído com a especificação.

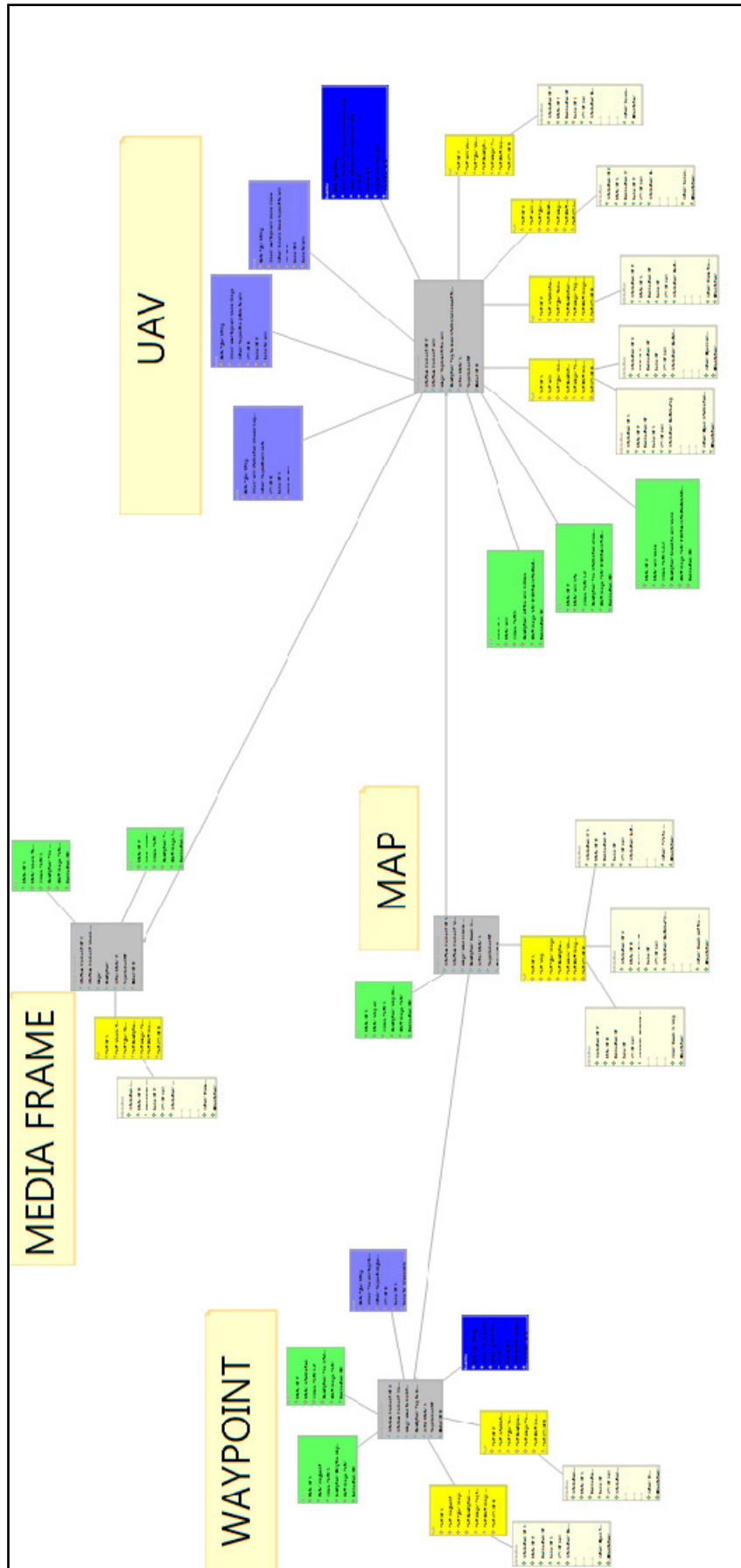


Figura 60 - Diagrama de Especificação do Estudo de Caso Voo Interativo

A fim de exibir melhor os nós do diagrama de especificação do estudo de caso “Voo Interativo” (Figura 60), as Figuras 61, 62 e 63 mostram, respectivamente, o Elemento de Interface “UAV” com duas de suas *Parts*, o Elemento de Interface “Waypoint” com duas *Parts* e dois *Listeners*, e a *Part* “Map” do Elemento de Interface “Map” com três interações.

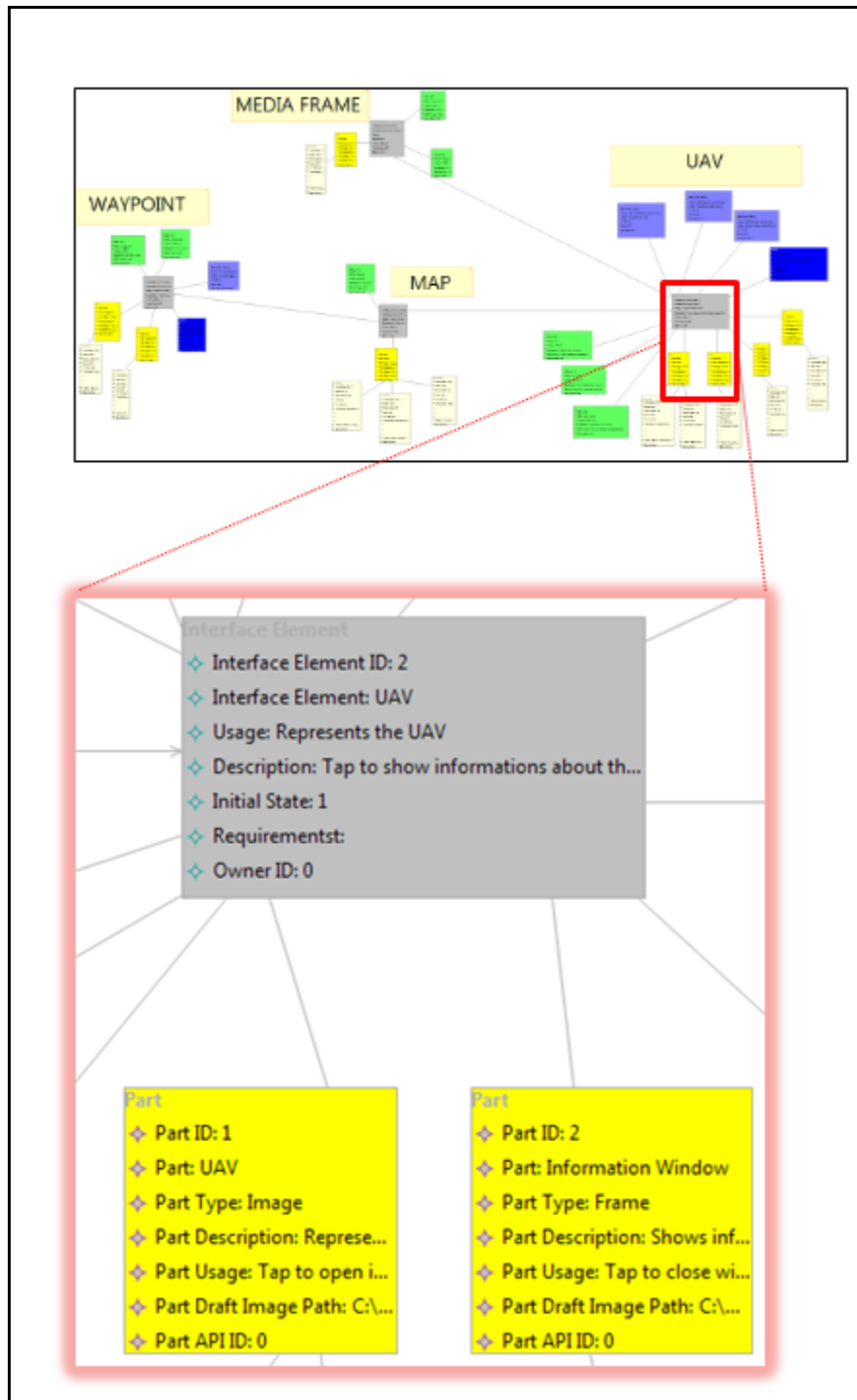


Figura 61 - Elemento de Interface UAV com duas *Parts*

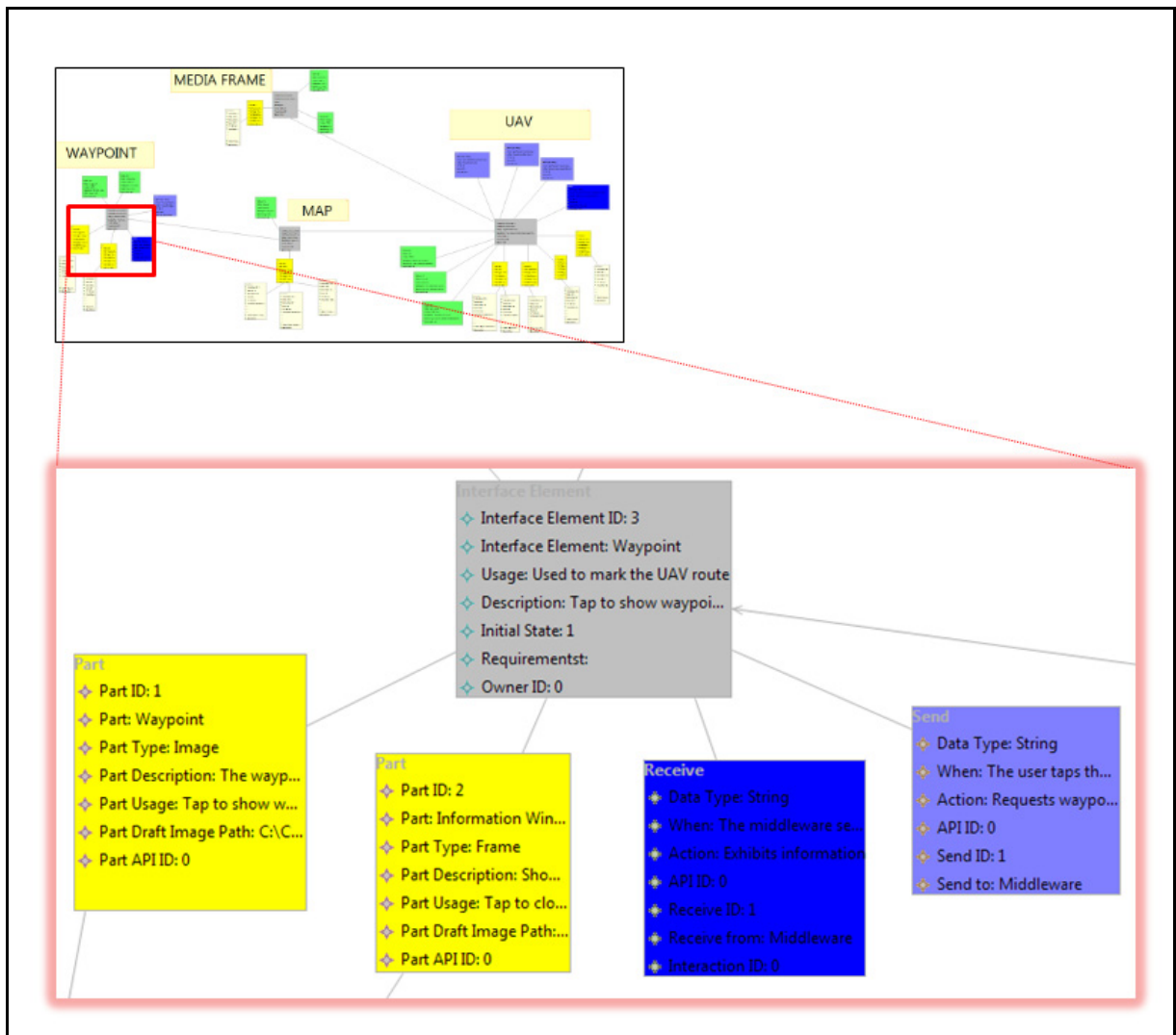


Figura 62 - Elemento de Interface Waypoint com suas *Parts* e Listeners

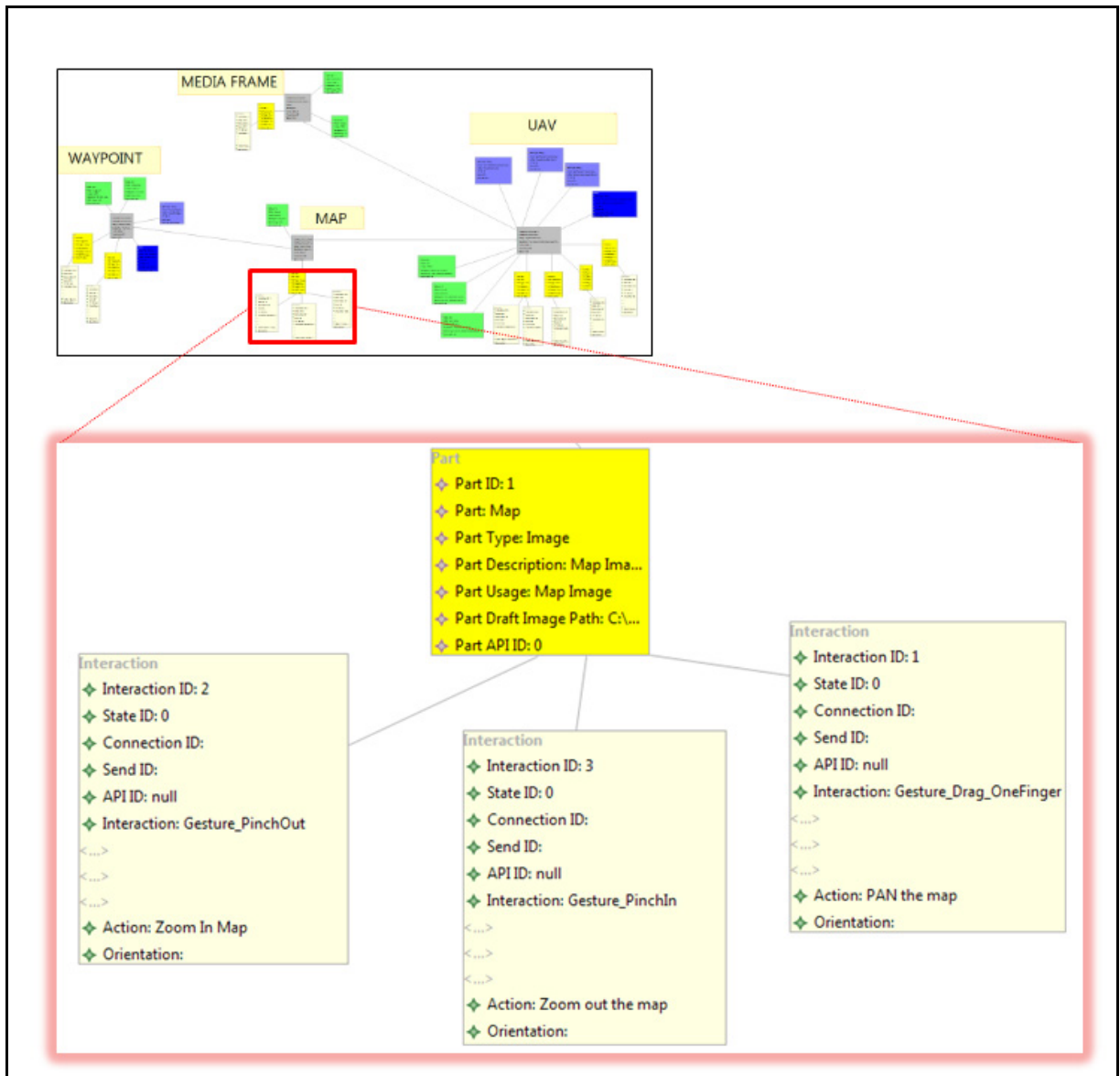


Figura 63 - Part Map do Elemento de Interface Map com suas interações

É importante ressaltar a especificação de gestos multitoques na Figura 63, em que os gestos “Zoom In”, “Zoom Out” e “PAN” da Parts “Map” são especificados, mostrando assim o suporte do TTUI-SM a gestos.

Assim como no estudo de caso anterior, para mostrar um resultado da especificação deste estudo de caso, a Figura 64 exibe a Ficha do Elemento de Interface “Waypoint”.


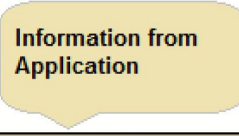
Interface Element: Waypoint									
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description					
3	Waypoint	1	0	Usage			Operation		
Requirements				Used to mark the UAV route			Tap to show waypoint informations		
Parts									
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
1	Waypoint	Image	The waypoint composes the UAV route			Tap to show waypoint informations		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
1	2				Open the information window	Gesture_Tap	-		
Parts									
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
2	Information Window	Frame	Shows information about the waypoint			Tap to close information window		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
2	1				Closes the information window	Gesture_Tap	-		
States									
State ID	State Name	Visible Parts	Connection IDs	State Description					
1	Waypoint	1		Only the waypoint is visible					
Draft									
State ID	State Name	Visible Parts	Connection IDs	State Description					
2	Information	1,2		The information window is opened					
Draft									
API									
Listeners									
SEND	Send ID	To	When	Data Type	Action	API ID			
	1	Middleware	The user taps the Waypoint part	String	Requests waypoint information	0			
RECEIVE	Receive ID	From	When	Data Type	Action	API ID			
	1	Middleware	The middleware sends information data	String	Exhibits information	0			
Connections									
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action				

Figura 64 - Ficha do Elemento de Interface Waypoint

As fichas dos demais elementos de interface podem ser encontradas no Anexo D.

6.3 Experimento TTUI-SM

6.3.1 Etapas do Experimento de Avaliação

O experimento foi dividido em três etapas. Na primeira etapa, foi dada uma introdução sobre o que são as interfaces tangíveis e seus tipos, incluindo as interfaces tangíveis *tabletop*. Em seguida, foi apresentado o conceito geral sobre especificação de interfaces e também algumas abordagens encontradas na literatura para realizar essa especificação e suas interações. Por fim, a motivação do modelo TTUI-SM foi explicada e o modelo apresentado juntamente com sua ferramenta e alguns exemplos.

Na segunda etapa, foi solicitado que os participantes realizassem, em duplas, a especificação de duas aplicações simples que explorassem tanto as interações multitoques quanto as interações com objetos tangíveis, com o objetivo de fornecer aos participantes um contato com diferentes aspectos de aplicações *tabletop*. A primeira aplicação foi a “*Pictures Demo*”, uma demonstração do framework Kivy (KIVY, 2010) em que é possível arrastar, redimensionar e rotacionar as fotos presentes na tela. Os participantes receberam instruções para especificar os elementos de interface presentes na aplicação sendo possível arrastar, rotacionar e redimensionar as fotos além de reestabelecer o tamanho original das fotos com dois toques sobre ela. A Figura 65a exibe a imagem da aplicação.

A segunda aplicação foi similar à aplicação encontrada na Reactable (JORDÀ, 2010), porém simplificada. Nela, é possível colocar um objeto quadrado sobre a superfície para iniciar um ritmo musical e o objeto recebe um contorno quadrado. É também possível colocar um objeto circular sobre a superfície e, com a rotação em sentido horário e anti-horário, o volume da música aumenta e diminui, respectivamente, e o objeto recebe um contorno circular. Em ambas as interações, uma mensagem era enviada para um controlador multimídia externo. A Figura 65b exibe a imagem da Reactable completa utilizada como base para a segunda aplicação.



Figura 65 – (a) *Pictures Demo* do Kivy (KIVY, 2010) e (b) Reactable

Na terceira e última etapa do experimento, os participantes receberam, individualmente, um questionário para avaliarem o modelo TTUI-SM e sua ferramenta de especificação em relação à sua facilidade de uso, compreensão, recursos e completude. Foram questionados o grau de conhecimento dos participantes em relação às interfaces tangíveis *tabletop* antes do experimento e se o experimento contribuiu para o aprendizado deste tipo de interface.

O questionário utilizou a escala de Likert, uma escala conhecida por sua simplicidade, tanto para construir as questões, quanto para respondê-las (SEASHORE; KATZ, 1982). As questões elaboradas utilizando a escala de Likert têm cinco possíveis respostas sobre as impressões do participante: Discordo Totalmente (1 ponto), Discordo (2 pontos), Indiferente (3 pontos), Concordo (4 Pontos) e Concordo Totalmente (5 pontos). Além da escala de Likert, o questionário continha quatro questões abertas onde o usuário pôde apontar os pontos positivos e negativos em relação ao modelo, além de sugestões de melhoria.

6.3.2 Resultados e Discussão

Com base nas respostas dos questionários do experimento (Anexo C), que nas questões de 2 a 10 utilizaram a escala de Likert (com seus valores variando de 1 a 5), as médias foram calculadas a fim de estabelecer notas, para cada uma delas, em relação à utilidade, qualidade e facilidade do modelo TTUI-SM e sua ferramenta. Assim, a Tabela 3 apresenta as questões e suas notas finais. Da mesma forma, a Figura 66 exhibe graficamente os resultados das questões.

Tabela 3 - Resultados do Questionário

Questão	Nota
Q2 - O modelo TTUI-SM abrange as necessidades (em termos de especificação) das interfaces tangíveis <i>tabletop</i>.	4,55
Q3 - A divisão do modelo TTUI-SM em seções facilita a compreensão do mesmo.	4,5
Q4 - O Diagrama de Elemento de Interface hierárquico e abstrato da primeira fase do modelo TTUI-SM é de fácil compreensão.	4,15
Q5 - O layout da Ficha de Elemento de Interface simplifica a organização dos dados especificados e é compreensível.	4,25
Q6 - Ao desenvolver uma aplicação do domínio das interfaces tangíveis <i>tabletop</i>, as Fichas de Elemento de Interface podem facilitar a codificação ao serem utilizadas como documentação.	4,45
Q7 - A ferramenta de especificação do TTUI-SM facilita e agiliza a especificação dos elementos de interface.	4,5
Q8 - É fácil utilizar a ferramenta de especificação do TTUI-SM.	4,35
Q9 - A geração de arquivo de definições XML da ferramenta do TTUI-SM é um recurso importante e facilita na codificação da aplicação.	4,35
Q10 - De forma geral e com o uso da ferramenta, o modelo TTUI-SM é de fácil utilização.	4,45

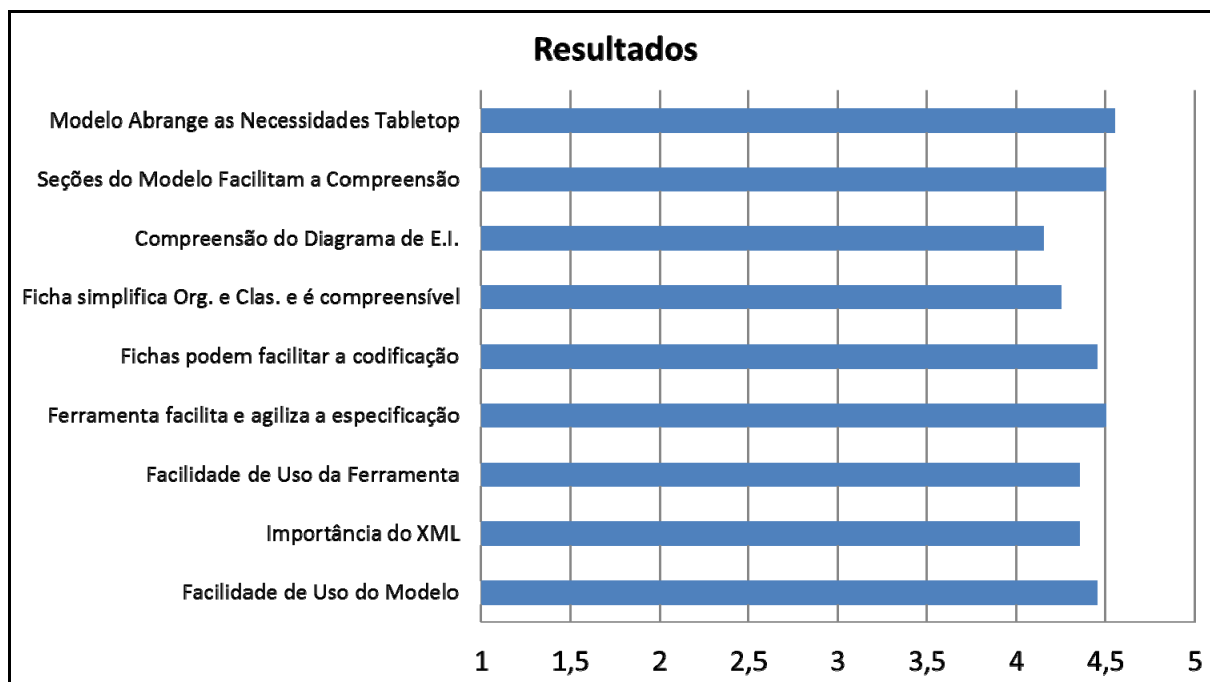


Figura 66 - Resultado do Questionário TTUI-SM

De acordo com os resultados apresentados na Tabela 3, os participantes, em sua maioria, concordam sobre a relevância, facilidade, utilidade, importância e capacidade de compreensão do modelo TTUI-SM e sua ferramenta. Entretanto, a questão 3 com nota 4,15 sugere que a maneira pela qual o modelo TTUI-SM é dividido, ou seja, em porções, pode fazer com que haja dúvidas em relação à ligação entre elas, embora ainda assim facilite a compreensão do modelo. Com base nas questões sobre as porções, a Tabela 4 exhibe quais delas foram apontadas como mais difíceis e mais fáceis (com mais de um voto) de serem especificadas de acordo com o nível de conhecimento dos participantes.

Tabela 4 - Seções difíceis e fáceis

Nível de Conhecimento	Difícil	Fácil
Nenhum	<i>Parts</i>	Descrição Geral
Mínimo	<i>Parts</i>	Descrição Geral
Básico	<i>Parts e States</i>	Descrição Geral e <i>Parts</i>
Avançado	-	<i>Parts</i>

A Tabela 4 aponta que os participantes com menor nível de conhecimento tiveram problemas em compreender a especificação das *Parts* dos elementos de

interface. O autor julga que este fato está diretamente ligado à falta de conhecimento sólido do paradigma de interação das interfaces tangíveis *tabletop*, uma vez que alguns participantes, com conhecimento básico e avançado, responderam que a especificação das *Parts* é a mais fácil. As seções mais difíceis, votadas pelos participantes de conhecimento avançado (Descrição Geral, API, *Parts*, *States* e *Connections*), não foram listadas na tabela por não receberam mais de um voto cada. De forma gráfica, as seções mais difíceis e mais fáceis de acordo com o nível de conhecimento dos participantes são exibidas nas Figuras 67 e 68, respectivamente.

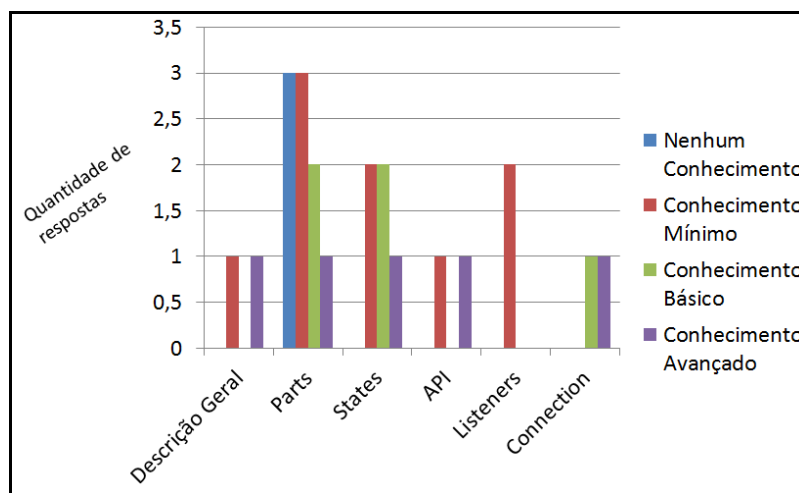


Figura 67 - Seções mais difíceis por nível de conhecimento

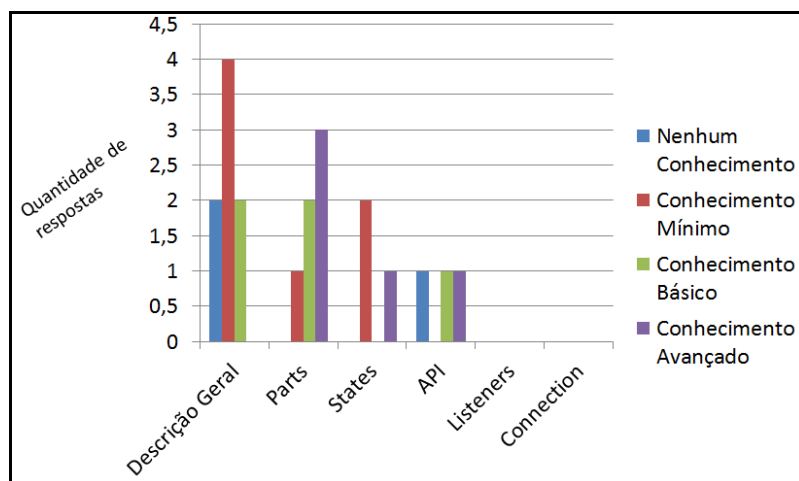


Figura 68 - Seções mais fáceis por nível de conhecimento

Devido à diversidade de níveis de conhecimento sobre interfaces tangíveis *tabletop* (desde “Nenhum Conhecimento” até “Conhecimento Avançado”), o autor

observou que o entendimento e dúvidas geradas durante a execução do experimento variaram de participante para participante, independentemente do nível acadêmico. Dos 20 participantes, três alegaram nenhum conhecimento, sete alegaram conhecimento mínimo, cinco alegaram conhecimento básico e cinco alegaram conhecimento avançado no assunto.

As seções *Listeners* e *Connection* não foram apontadas como fáceis por nenhum dos participantes enquanto *Connection* foi apontada por difícil por alguns. Este fato deixa claro que além da seção *Parts*, os participantes também tiveram dificuldades com as seções *Listeners* e *Connection*.

Foi observado que, dentre todos os participantes, as dúvidas mais específicas sobre o modelo TTUI-SM foram por aqueles que alegaram conhecimento avançado na interface alvo do experimento, enquanto aqueles que alegaram nenhum conhecimento ou conhecimento mínimo apresentaram dúvidas mais conceituais em relação ao paradigma de interação e o funcionamento da ferramenta TTUI-SM.

Outro ponto observado foi em relação à exploração da flexibilidade oferecida pelo modelo TTUI-SM, que possibilita realizar a especificação dos elementos de interface de diversas formas, diferentes e corretas. Uma das duplas participantes da atividade prática do experimento, por exemplo, especificou a interação de aumento e diminuição de volume, da segunda aplicação, utilizando apenas uma interação de rotação por meio da verificação da alteração de graus enquanto os demais grupos utilizaram duas interações de rotação com orientação em sentido horário e anti-horário.

Um dos participantes apontou como ponto positivo do modelo TTUI-SM e sua ferramenta de especificação que **“Organiza de uma forma hierárquica as especificações”** enquanto outros participantes disseram que é positiva **“a facilidade de compreensão, pela forma que foi organizado e a simplicidade do uso da ferramenta de especificação”** e **“utilizando a ferramenta, o designer pode ganhar tempo e também poderá ter em mãos uma ótima documentação, pois a ferramenta gera a ficha em pdf com os detalhes de cada interface”**.

Em relação aos pontos negativos, os participantes destacaram o esquema de cores dos nós da ferramenta de especificação, a falta de uma legenda entre as ligações dos nós da ferramenta e alguns outros quesitos de usabilidade como ordem de atributos e visualização icônica da especificação (que o autor explica melhor na sessão de trabalhos futuros).

Em sua maioria, os participantes do experimento afirmaram que obtiveram ganho de conhecimento sobre este tipo de interface. Dois dos participantes, que alegaram conhecimento avançado, disseram: “**Aprendi uma possível metodologia de especificação e acredito que essa é uma ótima ferramenta para uso.**” e “**Foi bom, considerando eu ter uma prévia experiência em TUI, saber de um modelo de especificação que ajudaria muito o desenvolvedor**”. Um dos participantes que alegou ter nenhum conhecimento sobre as interfaces tangíveis *tabletop* afirmou sobre a experiência obtida: “**Conhecimento sobre interfaces tangíveis tabletop e especificação da mesma**”.

O autor observou durante a apresentação do modelo para os participantes, uma possível dificuldade da especificação em relação à classificação hierárquica das aplicações, ou seja, quando uma aplicação tem seus elementos gráficos apresentados de forma hierárquica (em camadas, por exemplo), pode haver alguma dificuldade para a realização da especificação dos elementos de interface, uma vez que o modelo TTUI-SM não possibilita a especificação direta da hierarquia. Entretanto, é perfeitamente possível a realização da especificação de elementos de interface de forma hierárquica e em camadas, como é mostrado nos estudos de caso deste capítulo.

Por fim, por questões de tempo, não foi possível realizar a implementação das interfaces especificadas durante o experimento e estudos de caso. Entretanto, analisando as respostas dadas pelos participantes por meio do questionário sobre a geração dos arquivos XML e das Fichas de Elemento de Interface, foi identificado um grande potencial de automatização por meio da geração de código.

6.4 Considerações Finais

Neste capítulo, os estudos de caso “Voo Interativo” e “Vigilância e Segurança de Infraestruturas Críticas” foram apresentados e explicados. Em seguida seus elementos de interface foram identificados e especificados utilizando a ferramenta TTUI-SM.

Foi também apresentado o experimento realizado com 20 participantes, com o objetivo de avaliar o modelo TTUI-SM e sua ferramenta de especificação, em que foi apresentado um roteiro explicativo do modelo TTUI-SM, seguido pela especificação de duas interfaces tangíveis *tabletop*. No final do experimento, os usuários preencheram questionários e o resultado foi apresentado e discutido neste capítulo.

No próximo capítulo, serão feitas as conclusões finais sobre este trabalho, bem como a apresentação das limitações encontradas durante seu desenvolvimento e também os trabalhos futuros.

Capítulo 7

CONCLUSÕES

O modelo de especificação de interfaces *tabletop* TTUI-SM foi desenvolvido para agilizar a especificação de interfaces para diferentes aplicações. Para apoiar o modelo de especificação, foi desenvolvida uma ferramenta diagramática, TTUI-SMT, que se baseia no modelo TTUI-SM e agiliza o processo de especificação. A ferramenta, por sua vez, realiza a geração automática das fichas de elemento de interface e também de um arquivo XML com as definições dos elementos de interface. Para validar o modelo e a ferramenta, dois estudos de caso foram realizados e, para avaliar a ambos, um experimento com 20 participantes foi conduzido.

Por meio dos resultados obtidos com a análise das respostas dos questionários entregues aos participantes do experimento, e com a construção dos estudos de caso, é possível concluir que o modelo TTUI-SM cumpre seu objetivo de oferecer um modelo de especificação para interfaces tangíveis *tabletop* que suporte a especificação de gestos, desde os mais simples, como um simples toque sobre a superfície, até gestos avançados, que podem ser customizados, permitindo que o designer os especifique com quantos toques desejar e com qualquer padrão de gesto.

O modelo também cumpre o objetivo de simplificar a organização dos dados especificados por meio das fichas de elemento de interface das quais, a partir de seções distintas, é possível compreender como os elementos gráficos da interface se comportam, como se interligam e quais ações podem ser realizadas com eles para atingir as tarefas do usuário.

Embora os objetivos tenham sido cumpridos, limitações e ideias foram identificadas durante a realização do estudo de caso e do experimento.

7.1 Limitações

Por ser um plug-in da IDE Eclipse, a ferramenta de especificação do TTUI-SM necessita da utilização do Eclipse. Desta forma, todas as limitações presentes na IDE Eclipse se aplicam à ferramenta de especificação TTUI-SM. Entre elas, algumas relativas à usabilidade como, por exemplo, o recurso Copiar/Colar para os nós, e também a edição de campos diretamente nos nós, por meio de duplo clique, são afetados devido à versão da IDE mencionada. Por exemplo, constatou-se que na versão Ganymede do Eclipse era possível editar as propriedades dos nós por meio de duplo clique, porém na versão Indigo, não.

O modelo TTUI-SM tem seu foco limitado apenas aos elementos de interface da aplicação e, desta forma, não é possível, no momento, especificar seu posicionamento absoluto na interface em relação a outros elementos. É considerado que o posicionamento entre elementos de interface diferentes será realizado no momento da codificação da aplicação.

7.2 Trabalhos Futuros

Por questões de tempo, alguns recursos adicionais e ideias que foram identificadas como potencializadores do modelo TTUI-SM e de sua ferramenta não puderam ser implementados. Entretanto, tais recursos foram discutidos e suas viabilidades estudadas a fim de decidir sua real importância dentro do contexto da especificação TTUI-SM e então classificados como trabalhos a serem desenvolvidos futuramente e descritos a seguir.

7.2.1 Tratamento de Exceções

É possível que interações diretas (feitas pelos usuários) ou indiretas (disparadas pela rede) ocasionem erros de aplicação e gerem exceções que podem necessitar um tratamento especial que implique em alteração de *States* dos elementos de interface da aplicação. Assim, o tratamento de exceções da interface foi identificado como potencial trabalho futuro, a fim de prover maior estabilidade das aplicações de interfaces tangíveis *tabletop* já no momento da especificação.

7.2.2 Modo de Visualização

Com base na limitação sobre posicionamento absoluto dos elementos de interface, um importante recurso para a ferramenta de especificação TTUI-SM foi identificado como trabalho futuro. Tal recurso consiste em prover um modo de visualização para o designer durante o momento da especificação dos elementos de interface que lhe permita posicioná-los e suas *Parts* pela área de diagramação, podendo estimar o seu posicionamento em relação à interface como um todo.

A visualização será controlada por meio dos *States* dos elementos de interface especificados, dos quais o designer poderá selecionar um *State* e visualizar as *Parts* visíveis daquele *State* a partir dos *Drafts* especificados, podendo então construir a interface por meio do posicionamento das *Parts*. Assim, para cada *State*, é possível visualizar as *Parts* e alterar seus posicionamentos. A Figura 69 ilustra um esboço do modo de visualização na ferramenta TTUI-SM.

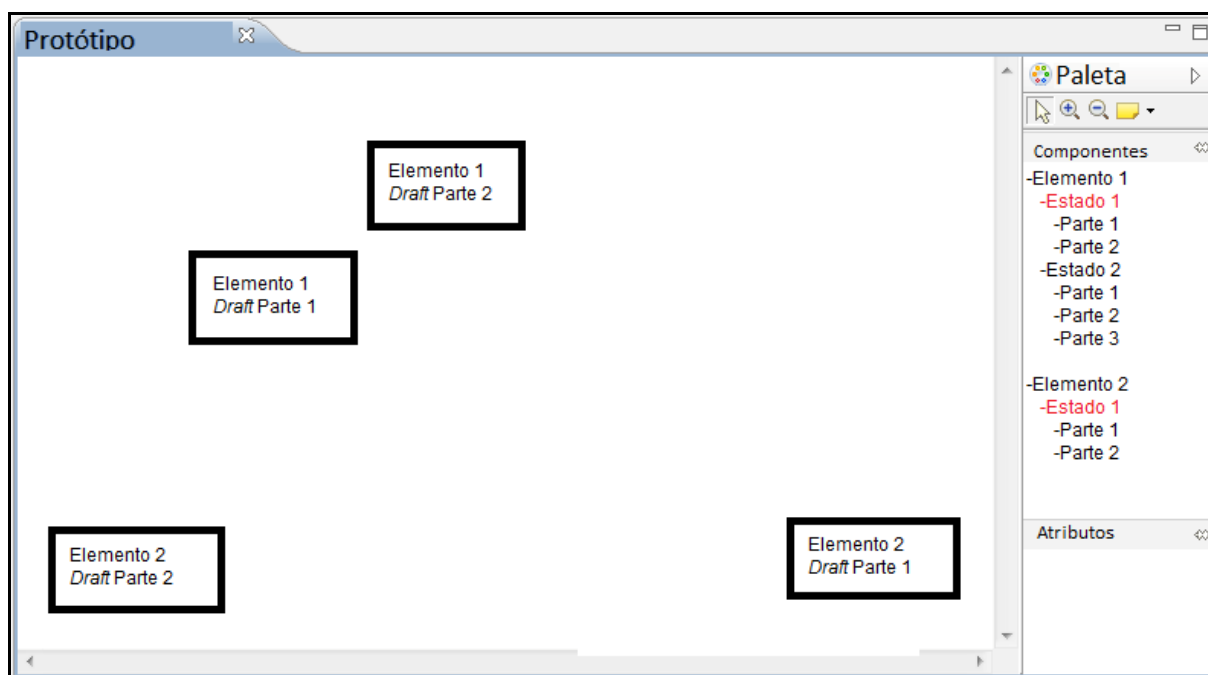


Figura 69 - Ilustração do Modo de Visualização da Ferramenta TTUI-SM

Na Figura 69, a paleta apresenta a lista dos elementos de interface especificados, juntamente com seus *States* e *Parts*. No “Elemento1”, o “Estado 1” está ativo e o *Draft* de suas duas *Parts* foram posicionadas na área à esquerda. O mesmo se repete para o “Elemento 2”. As posições das *Parts* são totalmente ditadas pelo designer.

Além de fornecer uma prévia da interface para o designer, outro objetivo do modo visualização é também possibilitar, futuramente, a especificação da posição das *Parts* dos elementos de interface e fornecê-la no arquivo de definições XML.

7.2.3 Validação automatizada da ferramenta TTUI-SM

No momento, a ferramenta TTUI-SMT não realiza a validação da modelagem diagramática dos elementos de interface e dos dados especificados. Portanto, cabe ao designer realizar a especificação com os dados corretos. Como trabalho futuro, propõe-se a validação automática desses elementos, a fim de evitar a geração de fichas incompletas por falta de atenção do designer.

7.2.4 Criação de Parsers e Implementação de Interfaces

Como mencionado na seção 6.3.2, não foram realizadas implementações de interfaces a partir das fichas de elemento de interface ou arquivos XML. Visando encontrar possíveis ideias de melhorias e para adaptar o modelo TTUI-SM para as necessidades do laboratório WINDIS, serão desenvolvidos *parsers* para a conversão dos arquivos XML para diversas linguagens de programação que implementem interfaces para *tabletops* e, também, a especificação das interfaces tangíveis *tabletop* dos projetos do laboratório.

7.3 Contribuições

Uma das maiores contribuições deste trabalho é, como apresentado nos objetivos, organizar os dados de maneira simplificada. Assim, diversos aspectos das interfaces tangíveis *tabletop* foram organizados e interligados a fim de fornecer uma especificação mais robusta e completa. Além de fornecer suporte para a especificação de itens, já bastante conhecidos na literatura como interações e estados, o modelo TTUI-SM também oferece a possibilidade de especificar assinatura de métodos de código e envio/recebimento de dados a partir da rede (internet ou intranet) de forma coesa com o restante da especificação.

A ferramenta de especificação também é um recurso importante para o modelo TTUI-SM, pois facilita e agiliza a especificação dos elementos de interface e automatiza grande parte do processo com a geração das fichas de elemento de interface em PDF e das definições dos elementos interface em um arquivo XML. Os desenvolvedores podem, em posse do arquivo XML, desenvolver um *parser* que interprete as definições XML, gerando automaticamente classes e métodos e, desta forma, diminuir o tempo gasto na codificação das aplicações.

Em termos de publicações, a versão inicial deste trabalho foi publicada em 2011 na 1ª Conferência Brasileira de Sistemas Embarcados Críticos (CBSEC) como parte do grupo GT4 de Aplicações Integradoras.

Está sendo preparada uma versão deste trabalho para um dos grandes eventos internacionais como o *ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (EICS), que se trata de um grande evento sobre sistemas interativos com o foco na engenharia. O EICS busca artigos que visem métodos, técnicas e ferramentas que auxiliem o design e o desenvolvimento de novas modalidades de interfaces. Além deste evento, o autor visa submeter também para o renomado evento nacional *Conference on Graphics, Patterns and Images* (SIBGRAPI) no qual, todo ano, diversos artigos relacionados a várias modalidades de interface são publicados.

REFERÊNCIAS BIBLIOGRÁFICAS

ARYA, A.; HAMIDZADEH, B. Face Animation: A Case Study for Multimedia Modeling and Specification Languages. **Multimedia systems and content-based**, 2004.

AZEVEDO, P.; MERRICK, R.; ROBERTS, D. OVID to AUIML-user-oriented interface modelling. **TUPIS'2000 Workshop at the UML'2000**, 2000.

AZUMA, R.; BAILLOT, Y.; BEHRINGER, R.; FEINER, S.; JULIER, S.; MACINTYRE, B. Recent advances in augmented reality. **IEEE Computer Graphics and Applications**, v. 21, n. 6, p. 34-47, 2001.

BEGAULT, D. R. **3-D sound for virtual reality and multimedia**. Disponível em: <http://www.macs.hw.ac.uk/modules/F24VS2/Resources/Begault_2000_3d_Sound_Multimedia.pdf>. Acesso em: 20 mar. 2011.

BILLINGHURST, M.; KATO, H.; POUPYREV, I. The magicbook-moving seamlessly between reality and virtuality. **Computer Graphics and Applications, IEEE**, v. 21, n. 3, p. 6-8, 2001.

BOSSONARO, A. A. .; VILELA, M. A. .; MORAES, J. L. C. .; PRADO, A. F. .; ARAÚJO, R. B. . **An Integrated System to Support Critical Infrastructure Security**^{1º Congresso Brasileiro de Sistemas Embarcados Críticos. Anais...2011}

BOTEGA, L. C.; DOURADO, A. M. B.; JR, O. T.; ARAÚJO, R. B. **A New Specification Model for the Design of Tangible User Interfaces**^{1º Congresso Brasileiro de Sistemas Embarcados Críticos. Anais...2011}

BREWER, J.; WILLIAMS, A.; DOURISH, P. A Handle on What ' s Going On : Combining Tangible Interfaces and Ambient Displays for Collaborative Groups. **Computer**, p. 15-17, 2007.

BROWNE, D. P.; SHARRATT, B.; NORMAN, M. **The formal specification of adaptive user interfaces using command language grammar**Proceedings of the SIGCHI conference on Human factors in computing systems. **Anais...ACM**, 1986

CALVARY, G.; COUTAZ, J.; THEVENIN, D.; LIMBOURG, Q.; BOUILLON, L.; VANDERDONCKT, J. A unifying reference framework for multi-target user interfaces. **Interacting with Computers**, v. 15, n. 3, p. 289-308, jun. 2003.

CALVILLO-GÁMEZ, E. H.; LELAND, N.; SHAER, O.; JACOB, R. J. K. The TAC paradigm: unified conceptual framework to represent Tangible User Interfaces. **Proceedings of the Latin American conference on Human-computer interaction**, 2003.

CARR, D. A. **Specification of interface interaction objects** Proceedings of the SIGCHI conference on Human factors in computing systems: celebrating interdependence. **Anais...**ACM, 1994

CHEN, I. Y. H.; MACDONALD, B.; WUNSCHE, B. **Markerless augmented reality for robotic helicopter applications** Proceedings of the 2nd international conference on Robot vision. **Anais...**Springer-Verlag, 2008

COHEN, M.; GIANGOLA, J.; BALOGH, J. **Voice user interface design**. [S.l.] Addison-Wesley Professional, 2004.

DUMAS, B.; LALANNE, D.; OVIATT, S. Multimodal interfaces: A survey of principles, models and frameworks. **Human Machine Interaction**, p. 3–26, 2009.

ECHTLER, F.; KLINKER, G.; BUTZ, A. Towards a unified gesture description language. **Proceedings of the 13th International Conference on Humans and Computers**, p. 177-182, 2010.

ECLIPSE. **Eclipse IDE**. Disponível em: <www.eclipse.org>. Acesso em: 22 jan. 2012.

EDWARDS, A. D. N.; MITSOPOULOS, E. A principled methodology for the specification and design of nonvisual widgets. **ACM Transactions on Applied Perception**, v. 2, n. 4, p. 442-449, 1 out. 2005.

EHNES, J.; HIROTA, K.; HIROSE, M. **Projected augmentation-augmented reality using rotatable video projectors** Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality. **Anais...**IEEE Computer Society, 2004

EISENBEISS, H. A mini unmanned aerial vehicle (UAV): system overview and image acquisition. **International Archives of Photogrammetry. Remote Sensing and Spatial Information Sciences**, v. 36, n. 5/W1, 2004.

FIGUEROA, P.; GREEN, M.; HOOVER, H. J. InTml: a description language for VR applications. **Proceedings of the 7th International Conference on 3D Web Technology**, 2002.

FISHKIN, K. P.; GUJAR, A.; HARRISON, B. L.; MORAN, T. P.; WANT, R. Embodied user interfaces for really direct manipulation. **Communications of the ACM**, v. 43, n. 9, p. 74-80, 1 set. 2000.

FITZMAURICE, G. W.; ISHII, H.; BUXTON, W. A. S. **Bricks: laying the foundations for graspable user interfaces** Proceedings of the SIGCHI conference on Human factors in computing systems. **Anais...**ACM Press/Addison-Wesley Publishing Co., 1995. Acesso em: 20 mar. 2011

GOBBETTI, E.; SCATENI, R. Virtual reality: past, present and future. **Studies in Health Technology and Informatics**, v. 58, p. xxi, jan. 1998.

- GOMEZ, D.; BURDEA, G.; LANGRANA, N. Integration of the Rutgers Master II in a virtual reality simulation. **Proceedings Virtual Reality Annual International Symposium '95**, p. 198-202, 1995.
- HAREL, D. Statecharts: A visual formalism for complex systems. **Science of computer programming**, v. 8, p. 231-274, 1987.
- HARTSON, H.; HIX, D. Human-computer interface development: concepts and systems for its management. **ACM Computing Surveys (CSUR)**, v. 21, n. 1, p. 5-92, 1 mar. 1989.
- HAYASHI, K.; KATO, H.; NISHIDA, S. Occlusion detection of real objects using contour based stereo matching. **Proceedings of the 2005 international conference on Augmented tele-existence - ICAT '05**, p. 180, 2005.
- HENG, X.; LAO, S.; LEE, H.; SMEATON, A. F. A touch interaction model for tabletops and PDAs. **Proc. PPD**, p. 1-5, 2008.
- HOLMQUIST, L.; REDSTRÖM, J.; LJUNGSTRAND, P. Token-based access to digital information. **Handheld and Ubiquitous Computing**, p. 234-245, 1999.
- HUA, Z.; WHENG, L. N. Speech recognition interface design for in-vehicle system. **Proceedings of the 2nd International Conference on**, n. AutomotiveUI, p. 29-33, 2010.
- HUCKAUF, A.; URBINA, M. H.; GRUBERT, J.; BÖCKELMANN, I.; DOIL, F.; SCHEGA, L.; T"UMLER, J.; MECKE, R. **Perceptual issues in optical-see-through displays** Proceedings of the 7th Symposium on Applied Perception in Graphics and Visualization. **Anais...ACM**, 2010
- HUNTER, S.; MERRILL, D. Make a Riddle and TeleStory : Designing Children's Applications for the Siftables Platform. **Human Factors**, p. 6-9, 2010.
- ISHII, H.; ULLMER, B. **Tangible bits: towards seamless interfaces between people, bits and atoms** Proceedings of the SIGCHI conference on Human factors in computing systems. **Anais...ACM**, 1997
- JACOB, R. J. K.; DELIGIANNIDIS, L.; MORRISON, S. A software model and specification language for non-WIMP user interfaces. **ACM Transactions on Computer-Human Interaction (TOCHI)**, v. 6, n. 1, p. 1-46, 1 mar. 1999.
- JOHNSON, J.; ROBERTS, T. L.; VERPLANK, W.; SMITH, D. C.; IRBY, C. H.; BEARD, M.; MACKEY, K. The xerox star: A retrospective. **Computer**, v. 22, n. 9, p. 11-26, mar. 1989.
- JORDÀ, S. **The reactable: tangible and tabletop music performance** Proceedings of the 28th of the international conference extended abstracts on Human factors in computing systems. **Anais...ACM**, 2010
- KALTENBRUNNER, M.; BOVERMANN, T.; BENCINA, R.; COSTANZA, E. **TUIO: A protocol for table-top tangible user interfaces** Proc. of the The 6th Int'l Workshop on Gesture in Human-Computer Interaction and Simulation. **Anais...Citeseer**, 2005

KAMMER, D.; KECK, M.; FREITAG, G.; WACKER, M. Taxonomy and Overview of Multi-touch Frameworks: Architecture, Scope and Features. **Engineering Patterns for Multi-Touch Interfaces**, 2010.

KAY, A.; GOLDBERG, A. Personal dynamic media. **Computer**, v. 10, n. 3, p. 31–41, mar. 1977.

KIM, S.; SCHAP, T.; BOSCH, M.; MACIEJEWSKI, R.; DELP, E. J.; EBERT, D. S.; BOUSHEY, C. J. Development of a mobile user interface for image-based dietary assessment. **Conference on Mobile**, p. 1-7, 2010.

KIRNER, C.; SISCOOTTO, R. Realidade Virtual e Aumentada: Conceitos, Projeto e Aplicações. **Porto Alegre-RS: Sociedade Brasileira de Computação-SBC**, v. 1, p. 13–18, 2007.

KIVY. **Kivy.org**. Disponível em: <<http://kivy.org>>. Acesso em: 20 jun. 2012.

KLINKER, G.; REICHER, R.; BRUGGE, B. **Distributed user tracking concepts for augmented reality applications** Augmented Reality, 2000.(ISAR 2000). Proceedings. IEEE and ACM International Symposium on. **Anais...IEEE**, 2000

LEE, G. A.; NELLES, C.; BILLINGHURST, M.; KIM, G. J. Immersive Authoring of Tangible Augmented Reality Applications. **Third IEEE and ACM International Symposium on Mixed and Augmented Reality**, n. Ismar, p. 172-181, 2004.

LIMBOURG, Q.; VANDERDONCKT, J.; MICHOTTE, B.; BOUILLON, L.; FLORINS, M.; TREVISAN, D. **Usixml: A user interface description language for context-sensitive user interfaces** Proceedings of the ACM AVI'2004 Workshop "Developing User Interfaces with XML: Advances on User Interface Description Languages." **Anais...2004**

LOPUCK, L. **Designing multimedia: A visual guide to multimedia and online graphic design**. [S.l.: s.n.].

MARCHIONINI, G.; SIBERT, J. An agenda for human-computer interaction: science and engineering serving human needs. **ACM SIGCHI Bulletin**, v. 23, n. 4, p. 17–32, 1991.

MEDHI, I.; PATNAIK, S.; BRUNSKILL, E.; GAUTAMA, S. N. N.; THIES, W.; TOYAMA, K. Designing mobile interfaces for novice and low-literacy users. **ACM Transactions on Computer-Human Interaction**, v. 18, n. 1, p. 1-28, 1 abr. 2011.

MERRILL, D.; KALANITHI, J.; MAES, P. Siftables: towards sensor network user interfaces. **Proceedings of the 1st international**, p. 15-17, 2007.

MICROSOFT. **Microsoft Kinect Webpage**. Disponível em: <<http://www.xbox.com/kinect>>.

MOHRING, M.; LESSIG, C.; BIMBER, O. **Video see-through AR on consumer cell-phones** Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality. **Anais...IEEE Computer Society**, 2004. Acesso em: 20 mar. 2011

- MYERS, B. A. A brief history of human-computer interaction technology. **Interactions**, v. 5, n. 2, p. 44–54, 1998.
- MÁRQUEZ, J. J. An Introduction to Virtual Reality. **Humans and Automation Seminar**, out. 2002.
- NACENTA, M.; BAUDISCH, P.; BENKO, H. Separability of spatial manipulations in multi-touch interfaces. **Proceedings of Graphics**, p. 175-182, 2009.
- NAIMARK, L.; FOXLIN, E. Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. **Proceedings. International Symposium on Mixed and Augmented Reality**, n. Ismar, p. 27-36, 2002.
- NEWMAN, J.; INGRAM, D.; HOPPER, A. **Augmented reality in a wide area sentient environment** Augmented Reality, 2001. Proceedings. IEEE and ACM International Symposium on. **Anais...IEEE**, 2001
- OVIATT, S. Advances in robust multimodal interface design. **Computer Graphics and Applications, IEEE**, v. 23, n. 5, p. 62–68, 2003.
- PARNAS, D. L. **On the use of transition diagrams in the design of a user interface for an interactive computer system** Proceedings of the 1969 24th national conference. **Anais...ACM**, 1969. Acesso em: 16 ago. 2011
- PAYNE, S.; GREEN, T. R. G. Task-Action Grammars: A Model of the Mental Representation of Task Languages. **Human-Computer Interaction**, v. 2, n. 2, p. 93-133, 1 jun. 1986.
- PERLMAN, R. Using computer technology to provide a creative learning environment for preschool children. 1976.
- PERRY, L. D. S.; SMITH, C. M.; YANG, S. An investigation of current virtual reality interfaces. **Crossroads**, v. 3, n. 3, p. 23-28, abr. 1997.
- PHANOURIOU, C. UIML: A Device-Independent User Interface Markup. **Phd Thesis. Virginia Polytechnic Institute and State University**, 2000.
- POYNOR, R. The hand that rocks the cradle. **ID Magazine**, p. 60-65, 1995.
- PRESSMAN, R. **Software engineering: a practitioner's approach**. 5^a. ed. [S.l.: s.n.].
- PUBLICATION, J. Department of Defense Dictionary of Military and Associated Terms. v. 2010, n. March, 2012.
- PUERTA, A.; EISENSTEIN, J. XIIML: A universal language for user interfaces. **RedWhale Software**, 2001.
- RADDICHI, A. O. DESENVOLVIMENTO DE UM PROTÓTIPO DE HARDWARE DE. **Trabalho de Conclusão de Curso. UNIVEM**, 2010.

RANEBURGER, D.; POPP, R.; KAINDL, H.; JURGEN, F.; DOMINIK, E. Automated generation of device-specific WIMP UIs: weaving of structural and behavioral models. **Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems**, p. 41-46, 2011.

REACTIVISION. **Site oficial do ReactIVision**. Disponível em: <<http://reactivision.sourceforge.net/>>.

REITMAYR, G.; DRUMMOND, T. Going out: robust model-based tracking for outdoor augmented reality. **2006 IEEE/ACM International Symposium on Mixed and Augmented Reality**, p. 109-118, out. 2006.

RINTAKOSKI, K.; ALHO, B. G. S. **Improving the Coherence of Crisis Management: New Technologies for Command and Control Systems**. Disponível em: <www.europarl.europa.eu>. Acesso em: 25 abr. 2012.

ROGERS, Y.; SHARP, H.; PREECE, J. **Interaction design: Beyond human-computer interaction**. 2nd. ed. [S.l.] Wiley, 2006.

ROLLAND, J. P.; FUCHS, H. Optical versus video see-through head-mounted displays in medical visualization. **Presence: Teleoperators & Virtual Environments**, v. 9, n. 3, p. 287-309, jun. 2000.

SAUER, S.; ENGELS, G. Extending UML for modeling of multimedia applications. **Visual Languages, 1999. Proceedings.**, 1999.

SCAIFE, M.; ROGERS, Y. External cognition: how do graphical representations work? p. 185-213, 1996.

SEASHORE, S.; KATZ, D. Obituary: Rensis Likert (1903-1981). v. 37, n. 7, p. 851-853, 1982.

SHAER, O. Tangible User Interfaces: Past, Present, and Future Directions. **Foundations and Trends® in Human-Computer Interaction**, v. 3, n. 1-2, p. 1-137, 2009.

SHAER, O.; HORNECKER, E. Tangible User Interfaces: Past, Present, and Future Directions. **Foundations and Trends® in Human-Computer Interaction**, v. 3, n. 1-2, p. 1-137, 2009.

SHAER, O.; JACOB, R. J. K. A specification paradigm for the design and implementation of tangible user interfaces. **ACM Transactions on Computer-Human Interaction**, v. 16, n. 4, p. 1-39, nov. 2009.

SUTHERLAND, I. E. **Sketchpad: a man-machine graphical communication system** Proceedings-Spring Joint Computer Conference, Michigan. **Anais...dez.** 1963

SUTHERLAND, I. E. **The ultimate display** Proceedings of the IFIP Congress. **Anais...Citeseer**, 1965

TUIO. **Site oficial do TUIO**. Disponível em: <www.tuio.org>.

ULLMER, B. A. Tangible Interfaces for Manipulating Aggregates of Digital Information. **PhD Thesis. Massachusetts Institute of Technology**, 2002.

ULLMER, B.; ISHII, H. Emerging frameworks for tangible user interfaces. **IBM Systems Journal**, v. 39, n. 3, p. 915-931, 2000.

VAN DAM, A. Beyond WIMP. **Computer Graphics and Applications, IEEE**, v. 20, n. 1, p. 50-51, 2000.

WINGRAVE, C.; JR, J. L. A natural, tiered and executable UIDL for 3D user interfaces based on Concept-Oriented Design. **ACM Transactions on**, v. 16, n. 4, p. 1-36, 2009.

ZAFRULLA, Z.; BRASHEAR, H.; STARNER, T. American sign language recognition with the kinect. **Proceedings of the 13th**, p. 279, 2011.

ZHOU, F.; DUH, H. B. L.; BILLINGHURST, M. Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR. **2008 7th IEEE/ACM International Symposium on Mixed and Augmented Reality**, v. 2, p. 193-202, set. 2008.

Anexo A

TEMPLATE XML DE ELEMENTO DE INTERFACE

```
<?xml version="2.0" encoding="utf-8"?>
<interface>
  <interfaceelement id="">
    <name></name>
    <usage></usage>
    <operation></operation>
    <initialstate></initialstate>
    <requirements></requirements>
    <ownerid></ownerid>
    <parts>
      <part id="">
        <partname></partname>
        <parttype></parttype>
        <partdescription></partdescription>
        <partusage></partusage>
        <partdraft></partdraft>
        <apiid></apiid>
        <interactions>
          <interaction id="">
            <interactionaction></interactionaction>
            <interactionorientation></interactionorientation>
            <interactiontype>
              <type></type>
              <!--Caso seja uma interação tangível
              <objectid></objectid> -->
              <!--Caso seja um gesto customizado
              <numberoffingers></numberoffingers>
              <pattern></pattern> -->
            </interactiontype>
            <connectionid></connectionid>
            <stateid></stateid>
            <sendid></sendid>
            <apiid></apiid>
          </interaction>
        </interactions>
      </part>
    </parts>
  </interfaceelement>
</interface>
```



```

        </interaction>
    </interactions>
</part>
</parts>
<states>
    <state id="">
        <statename></statename>
        <visibleparts></visibleparts>
        <statedescription></statedescription>
        <draft></draft>
    </state>
</states>
<apis>
    <api id="">
        <methodname></methodname>
        <methodtype></methodtype>
        <parameters></parameters>
        <description></description>
        <algorithm></algorithm>
    </api>
</apis>
<senders>
    <send id="">
        <to></to>
        <datatype></datatype>
        <when></when>
        <action></action>
        <apiid></apiid>
    </send>
</senders>
<receivers>
    <receive id="">
        <from></from>
        <datatype></datatype>
        <when></when>
        <action></action>
        <apiid></apiid>
    </receive>
</receivers>
<connections>
    <connection id="">
        <connectiontype></connectiontype>
        <interfaceelementid></interfaceelementid>
        <interfaceelementstateid></interfaceelementstateid>
        <apiid></apiid>
    </connection>
</connections>
</interfaceelement>
</interface>

```

Anexo B

FICHAS DO ESTUDO DE CASO VIGILÂNCIA E SEGURANÇA DE INFRAESTRUTURAS CRÍTICAS



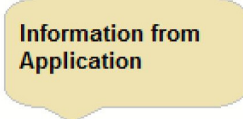


Interface Element: Personnel Control									
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description					
2	Personnel Control	1000	0	Usage	Operation				
Requirements				Used as container of personnel control interface elements					
Parts									
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
1	Legend	Image	Shows the Personnel Control Layer Legend					0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
States									
State ID	State Name	Visible Parts		Connection IDs	State Description				
1	Enabled	1		1,2,3,4,5	The layer is visible				
Draft									
State ID	State Name	Visible Parts		Connection IDs	State Description				
1000	Disabled			6,7,8,9,10	Layer disabled				
Draft									
API									
Listeners									
SEND	Send ID	To		When	Data Type	Action	API ID		
RECEIVE	Receive ID	From		When	Data Type	Action	API ID		
Connections									
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action				
1	3	Active	1	0					
2	4	Active	1	0					
3	5	Active	1	0					
4	6	Active	1	0	Exhibits Reaction Force				
5	7	Active	1	0	Exhibits employee				
6	3	Active	1000	0	Disables sentinel				
7	4	Active	1000	0	Disables the commander				
8	5	Active	1000	0	Disables the patrolmen				
9	6	Active	1000	0	Disables the reaction force				
10	7	Active	1000	0	Disables the employee				

Figura 70 - Elemento de Interface da Camada de Controle de Pessoal

Interface Element: Sentinel								
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description				
3	Sentinel	1	0	Usage		Operation		
Requirements				Shows the position of sentinels		Possible to double tap to show more informations		
Parts								
Part ID	Part Name	Part Type	Part Description		Part Usage		API ID	
1	Sentinel	Image Button	Shows positioning of the sentinel		If double tapped, shows information about the sentinel		0	
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
1	2				Shows more information	Gesture_DoubleTap	-	
Part ID	Part Name	Part Type	Part Description		Part Usage		API ID	
2	Information Window	Frame	Shows information about the sentinel				0	
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
2	1				Closes information frame	Gesture_Tap	-	
States								
State ID	State Name	Visible Parts	Connection IDs	State Description				
1	Sentinel	1		Only the sentinel is visible				
Draft								
								
State ID	State Name	Visible Parts	Connection IDs	State Description				
2	Show information	1,2		The information frame is opened for information				
Draft								
								
State ID	State Name	Visible Parts	Connection IDs	State Description				
1000	Disabled			Personnel Layer and all its interface elements are hidden				
Draft								
API								
Listeners								
SEND	Send ID	To	When	Data Type	Action	API ID		
RECEIVE	Receive ID	From	When	Data Type	Action	API ID		

Connections					
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action

Figura 71 - Elemento de Interface Sentinela


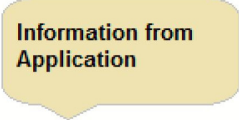


Interface Element: Commander								
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description				
4	Commander	1	0	Usage			Operation	
Requirements				Shows the position of the commander		Double tap shows more informations		
Parts								
Part ID	Part Name	Part Type	Part Description		Part Usage		API ID	
1	Commander	Image Button	Shows the position of the commander		If double tapped, shows information about the commander		0	
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
1	2				Shows the information window	Gesture_DoubleTap	-	
Part ID	Part Name	Part Type	Part Description		Part Usage		API ID	
2	Information Window	Frame	Show more informations about commander		If tapped, closes the information window		0	
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
2	1				Close information window	Gesture_Tap	-	
States								
State ID	State Name	Visible Parts	Connection IDs	State Description				
1000	Disabled			Personnel Layer and all its interface elements are hidden				
Draft								
State ID	State Name	Visible Parts	Connection IDs	State Description				
2	Commander Informations	1,2		The information frame is openend for information				
Draft								
								
State ID	State Name	Visible Parts	Connection IDs	State Description				
1	Commander	1		Only the commander				
Draft								
								
API								
Listeners								
SEND	Send ID	To	When	Data Type	Action	API ID		
RECEIVE	Receive ID	From	When	Data Type	Action	API ID		
Connections								
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action			

Figura 72 – Elemento de Interface Comandante


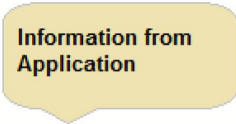

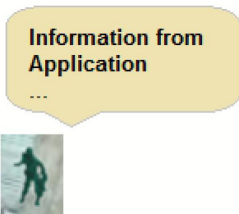
Interface Element: Patrolmen									
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description					
5	Patrolmen	1	0	Usage			Operation		
Requirements				Shows the position of the patrolmen			Double tap shows more information		
Parts									
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
1	Patrolman	Image Button	Shows the position of the patrolmen			Double tap to show more information about		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
1	2				Shows information window	Gesture_DoubleTap	-		
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
2	Information Window	Frame	Shows information about the patrolman			Tap to close the information window		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
2	1				Closes information window	Gesture_Tap	-		
States									
State ID	State Name	Visible Parts		Connection IDs	State Description				
1000	Disabled				Personnel Layer and all its interface elements are hidden				
Draft									
State ID	State Name	Visible Parts		Connection IDs	State Description				
1	Enabled	1			Only the patrolman				
Draft									
									
State ID	State Name	Visible Parts		Connection IDs	State Description				
2	Patrolman Informations	1,2			The information frame is openend for information				
Draft									
									
API									
Listeners									
SEND	Send ID	To	When		Data Type	Action	API ID		
RECEIVE	Receive ID	From	When		Data Type	Action	API ID		
Connections									
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action				

Figura 73 - Elemento de Interface Patrulheiro


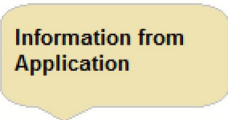


Interface Element: Reaction Force									
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description					
6	Reaction Force	1	0	Usage			Operation		
Requirements				Shows the position of the reaction forces			Double tap shows more information		
Parts									
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
1	Reaction Force	Image Button	Shows the position of reaction forces			Double tap opens information window		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
1	2				Opens the information window	Gesture_DoubleTap	-		
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
2	Information Window	Frame	Shows more information about the reaction force			Tap to close the window		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
2	1				Closes the information window	Gesture_Tap	-		
States									
State ID	State Name	Visible Parts	Connection IDs	State Description					
1000	Disabled			Personnel Layer and all its interface elements are hidden					
Draft									
State ID	State Name	Visible Parts	Connection IDs	State Description					
1	Enabled	1		Only the reaction force					
Draft									
									
State ID	State Name	Visible Parts	Connection IDs	State Description					
2	Reaction Force Informations	1,2		Shows information about the reaction force					
Draft									
									
API									
Listeners									
SEND	Send ID	To	When	Data Type	Action	API ID			
RECEIVE	Receive ID	From	When	Data Type	Action	API ID			
Connections									
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action				

Figura 74 - Elemento de Interface Força de Reação




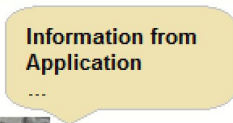




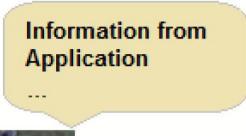

Interface Element: Employee									
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description					
7	Employee	1	0	Usage			Operation		
Requirements				Shows the position of the employee			Double tap to show information window		
Parts									
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
1	Employee	Image Button	Shows the position of the employee			Double tap to open information window		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs		Interaction Action	Interaction Type	Object ID	Orientation
1	2					Opens the information window	Gesture_DoubleTap	-	
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
2	Information Window	Frame	Shows more information about the employee			Tap to close information window		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs		Interaction Action	Interaction Type	Object ID	Orientation
2	1					Closes the information window	Gesture_Tap	-	
States									
State ID	State Name	Visible Parts		Connection IDs		State Description			
1	Enabled	1				Only the employee			
Draft									
									
State ID	State Name	Visible Parts		Connection IDs		State Description			
2	Employee Informativos	1,2				Shows information about the employee			
Draft									
									
State ID	State Name	Visible Parts		Connection IDs		State Description			
1000	Disabled					Personnel Layer and all its interface elements are hidden			
Draft									
API									
Listeners									
SEND	Send ID	To		When		Data Type	Action	API ID	
RECEIVE	Receive ID	From		When		Data Type	Action	API ID	
Connections									
Connection ID	Interface Element ID	Connection Type		Interface Element State ID	API ID	Action			

Figura 75 - Elemento de Interface Funcionário

Interface Element: Intelligence									
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description					
7	Intelligence	1000	0	Usage				Operation	
Requirements				Used as container of intelligence interface elements					
Parts									
Part ID	Part Name	Part Type	Part Description				Part Usage	API ID	
1	Intelligence Legend	Image	Legend of Intelligence Layer					0	
Draft									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs		Interaction Action	Interaction Type	Object ID	Orientation
States									
State ID	State Name	Visible Parts		Connection IDs		State Description			
1	Enabled	1		1,3,5,7		The layer is visible			
Draft									
State ID	State Name	Visible Parts		Connection IDs		State Description			
1000	Disabled			2,4,6,8		Intelligence disabled			
Draft									
API									
Listeners									
SEND	Send ID	To		When		Data Type	Action	API ID	
RECEIVE	Receive ID	From		When		Data Type	Action	API ID	
Connections									
Connection ID	Interface Element ID	Connection Type		Interface Element State ID	API ID	Action			
1	8	Active		1	0	Exhibites sensor network			
2	8	Active		1000	0	Hides sensor network			
3	9	Active		1	0	Exhibites robot route			
4	9	Active		1000	0	Hides the robot route			
5	10	Active		1	0	Exhibites UAV			
6	10	Active		1000	0	Hides UAV			
7	11	Active		1	0	Exhibites Sink			
8	11	Active		1000	0	Hides Sink			

Figura 76 - Elemento de Interface da Camada de Inteligência

Interface Element: Sensor Network								
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description				
8	Sensor Network	1	0	Usage	Operation			
Requirements				Shows the position of the sensor network		Double tap shows information window		
Parts								
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID
1	Sensor Network	Image Button	Shows the position of the sensor network			Double tap to open information window		0
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
1	2				Shows information window	Gesture_DoubleTap	-	
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID
2	Information Window	Frame	Shows information about sensor network			Tap to close information window		0
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
2	1				Closes the information window	Gesture_Tap	-	
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID
3	Alert Signal	Image	Alert signal for intrusion detection			Informs users about an event		0
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
3	3				Shows alert signal	Remote	-	
States								
State ID	State Name	Visible Parts	Connection IDs	State Description				
1	Sensor Network	1		Only the sensor network				
Draft								
								
State ID	State Name	Visible Parts	Connection IDs	State Description				
2	Sensor Network Informations	1,2		Shows information about the sensor network				
Draft								
								
								
State ID	State Name	Visible Parts	Connection IDs	State Description				
1000	Disabled			Hides the interface element				
Draft								
State ID	State Name	Visible Parts	Connection IDs	State Description				
3	Alert Signal	1,3		Sensor detected an event				
Draft								



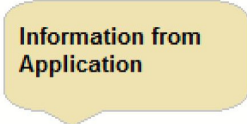


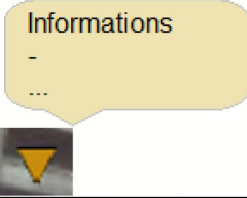


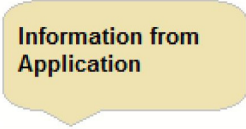

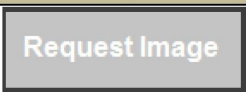
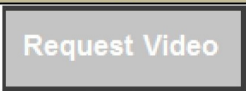
						
API						
Listeners						
SEND	Send ID	To	When	Data Type	Action	API ID
	1	Sensor Node	Information Window is opened	String	Requests sensor data	0
RECEIVE	Receive ID	From	When	Data Type	Action	API ID
	1	Sensor Node	Sensor node detect an event	String	Shows the alert signal of the sensor	0
	3	Sensor Node	Sensor node sends requested data	String	Receives the informations from sensor node	0
Connections						
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action	

Figura 77 - Elemento de Interface Redes de Sensores


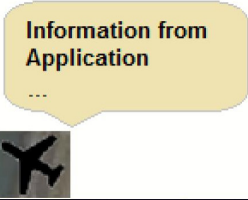

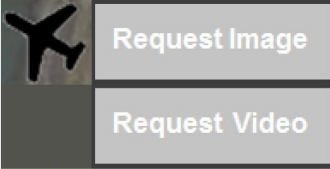

Interface Element: Robot Route									
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description					
9	Robot Route	1	0	Usage			Operation		
Requirements				Shows information about the robot route			Double tap opens information window		
Parts									
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
1	Robot Route	Image Button	Shows the position of robot route			Double tap opens information window		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
1	2				Opens information window	Gesture_DoubleTap	-		
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
2	Information Window	Frame	Shows information of robot route			Tap to close information window		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
2	1				Closes information window	Gesture_Tap	-		
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
3	Alert Signal	Image	Alert signal where robot detect an event			Informs users about an event		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
3	3				Shows alert signal	Remote	-		
States									
State ID	State Name	Visible Parts		Connection IDs	State Description				
1	Robot Route	1			Only the robot route				
Draft									
									
State ID	State Name	Visible Parts		Connection IDs	State Description				
2	Robot Route Informations	1,2			Shows information about robot route				
Draft									
									
State ID	State Name	Visible Parts		Connection IDs	State Description				
1000	Disabled				Hides the interface element				
Draft									
State ID	State Name	Visible Parts		Connection IDs	State Description				
3	Alert Signal	1,3			Point where robot detected an event				
Draft									
									

API						
Listeners						
SEND	Send ID	To	When	Data Type	Action	API ID
	1	Robot	User taps robot route	String	Requests robot position	0
RECEIVE	Receive ID	From	When	Data Type	Action	API ID
	1	Robot	Robot detects an event	String	Receives alert data and shows alert	0
	2	Robot	Robot sends requested data	String	Receives robot informations	0
Connections						
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action	

Figura 78 - Elemento de Interface Rota de Robô

Interface Element: UAV								
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description				
10	UAV	1	0	Usage	Operation			
Requirements				Shows the position of UAV		Double tap shows informations about UAV		
Parts								
Part ID	Part Name	Part Type	Part Description		Part Usage		API ID	
1	UAV	Image Button	Shows the position of the UAV		Double tap opens information window		0	
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
1	2				Opens the information window	Gesture_DoubleTap	-	
3	4				Opens UAV menu	Gesture_TapAndHold	-	
Part ID	Part Name	Part Type	Part Description		Part Usage		API ID	
2	Information Window	Frame	Shows information about UAV		Tap to close information window		0	
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
2	1				Closes information window	Gesture_Tap	-	
Part ID	Part Name	Part Type	Part Description		Part Usage		API ID	
3	Alert Signal	Image	Alert Signal when UAV detects an event		Informs user about an event		0	
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
4	3				Shows alert signal	Remote	-	
Part ID	Part Name	Part Type	Part Description		Part Usage		API ID	
4	UAV Menu Image	Menu Option	Menu option request Image		Requests images from UAV		0	
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
5	0				Sends Image request to UAV	Gesture_Tap	-	
Part ID	Part Name	Part Type	Part Description		Part Usage		API ID	
5	UAV Menu Video	Menu Option	Menu option request Video		Requests video stream from UAV		0	
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
6	0				Sends video stream request from UAV	Gesture_Tap	-	
Part ID	Part Name	Part Type	Part Description		Part Usage		API ID	
6	Media Frame	Frame	When a image or video stream is received, they are showed in this frame		Shows image or video stream fom UAV		0	
Draft								

Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
7	0				Raises the frame size	Gesture_PinchOut	-	
8	0				Diminishes the frame size	Gesture_PinchIn	-	
9	0				Drags the frame	Gesture_Drag_OneFinger	-	
10	1				Closes the frame	Gesture_TapAndHold	-	
11	5				Opens the media frame	Remote	-	

States				
State ID	State Name	Visible Parts	Connection IDs	State Description
1	UAV	1		Only the UAV
Draft				
				
2	UAV informations	1,2		Shows informations about the UAV
Draft				
				
1000	Disabled			Hides the interface element
Draft				
3	UAV Alert	1,3		Shows alert signal from UAV
Draft				
				
4	UAV Menu	1,4,5		Shows UAV Menu Options
Draft				
				
5	Image Frame	6		Shows image data frmo UAV
Draft				
				

API						
Listeners						
SEND	Send ID	To	When	Data Type	Action	API ID
	1	UAV	User taps UAV	String	Requests information about UAV	0
	2	UAV	User taps Image Menu Option	String	Requests Image from UAV	0
	3	UAV	User taps Video Menu Option	String	Requests video stream from UAV	0
RECEIVE	Receive ID	From	When	Data Type	Action	API ID
	1	UAV	UAV detects an event	String	Receives Alert from UAV	0
	2	UAV	UAV sends requested data	String	Receives information data from UAV	0
	3	UAV	UAV sends media	MediaStream	Receives media from UAV and opens Media Frame	0
Connections						
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action	

Figura 79 - Elemento de Interface VANT


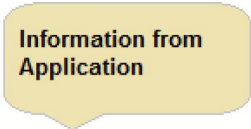


Interface Element: Sink								
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description				
11	Sink	1	0	Usage	Operation			
Requirements				Shows the position of the Sink		Double tap opens information window		
Parts								
Part ID	Part Name	Part Type	Part Description		Part Usage		API ID	
1	Sink	Image Button	Shows the position of the Sink		Double tap opens information window		0	
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
1	2				Opens information window	Gesture_DoubleTap	-	
Part ID	Part Name	Part Type	Part Description		Part Usage		API ID	
2	Information Window	Frame	Shows information about the Sink		Tap to close information window		0	
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
2	1				Closes the information window	Gesture_Tap	-	
States								
State ID	State Name	Visible Parts	Connection IDs	State Description				
1	Sink	1		Only the sink				
Draft								
								
State ID	State Name	Visible Parts	Connection IDs	State Description				
2	Sink Information	1,2		Shows information about the Sink				
Draft								
								
State ID	State Name	Visible Parts	Connection IDs	State Description				
1000	Disabled			Hides the interface element				
Draft								
API								
Listeners								
SEND	Send ID	To	When	Data Type	Action	API ID		
RECEIVE	Receive ID	From	When	Data Type	Action	API ID		
Connections								
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action			

Figura 80 - Elemento de Interface Mesa de Comando e Controle

Interface Element: Operations									
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description					
12	Operations	1000	0	Usage			Operation		
Requirements				Used as container of operations interface elements					
Parts									
Part ID	Part Name	Part Type	Part Description			Part Usage	API ID		
1	Legend	Image	Shows the legend of interface elements contained in operations layer				0		
Draft									
									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
Part ID	Part Name	Part Type	Part Description			Part Usage	API ID		
2	Sentinel	Image	Shows the position of sentinel point				0		
Draft									
									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
Part ID	Part Name	Part Type	Part Description			Part Usage	API ID		
3	Patrol Route	Image	Shows the points of patrol route				0		
Draft									
									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
Part ID	Part Name	Part Type	Part Description			Part Usage	API ID		
4	Command Center	Image	Shows the position of the command center				0		
Draft									
									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
Part ID	Part Name	Part Type	Part Description			Part Usage	API ID		
5	Reaction Force	Image	Shows the position of reaction forces				0		
Draft									
									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
Part ID	Part Name	Part Type	Part Description			Part Usage	API ID		
6	Entrance Control	Image	Shows the position of the entrance control				0		
Draft									
									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	

States						
State ID	State Name	Visible Parts	Connection IDs	State Description		
1	Enabled	1,2,3,4,5,6		The layer is visible		
				Draft		
State ID	State Name	Visible Parts	Connection IDs	State Description		
1000	Disabled			Operations Layer is Disabled		
				Draft		
API						
Listeners						
SEND	Send ID	To	When	Data Type	Action	API ID
RECEIVE	Receive ID	From	When	Data Type	Action	API ID
Connections						
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action	

Figura 81 - Elemento de Interface da Camada de Operações

Anexo C

QUESTIONÁRIO DE EXPERIMENTAÇÃO TTUI-SM

Experimentação TTUI-SM

Questionário

Nome: _____ Idade: _____

Instituição: _____ Período: _____

1- Qual seu grau de conhecimento sobre as interfaces tangíveis *tabletop* ANTES do experimento?

- Nenhum conhecimento
- Conhecimento Mínimo
- Conhecimento Básico
- Conhecimento Avançado

2- O modelo TTUI-SM abrange as necessidades (em termos de especificação) das interfaces tangíveis *tabletop*.

- Concordo Totalmente
- Concordo
- Indiferente
- Discordo
- Discordo Totalmente

3- A divisão do modelo TTUI-SM em seções facilita a compreensão do mesmo.

- Concordo Totalmente
- Concordo
- Indiferente
- Discordo
- Discordo Totalmente

4- O Diagrama de Elemento de Interface hierárquico e abstrato da primeira fase do modelo TTUI-SM é de fácil compreensão.

- Concordo Totalmente
- Concordo
- Indiferente
- Discordo
- Discordo Totalmente

5- O layout da Ficha de Elemento de Interface simplifica a organização dos dados especificados e é compreensível.

- Concordo Totalmente
- Concordo
- Indiferente
- Discordo
- Discordo Totalmente

6- Ao desenvolver uma aplicação do domínio das interfaces tangíveis *tabletop*, as Fichas de Elemento de Interface podem facilitar a codificação ao serem utilizadas como documentação.

- Concordo Totalmente
- Concordo
- Indiferente
- Discordo
- Discordo Totalmente

7- A ferramenta de especificação do TTUI-SM facilita e agiliza a especificação dos elementos de interface.

- Concordo Totalmente
- Concordo
- Indiferente
- Discordo
- Discordo Totalmente

8- É fácil utilizar a ferramenta de especificação do TTUI-SM.

- Concordo Totalmente
- Concordo
- Indiferente
- Discordo
- Discordo Totalmente

9- A geração de arquivo de definições XML da ferramenta do TTUI-SM é um recurso importante e facilita na codificação da aplicação.

- Concordo Totalmente
- Concordo
- Indiferente

- () Discordo
- () Discordo Totalmente

10- De forma geral e com o uso da ferramenta, o modelo TTUI-SM é de fácil utilização.

- () Concordo Totalmente
- () Concordo
- () Indiferente
- () Discordo
- () Discordo Totalmente

11- Qual seção do modelo TTUI-SM apresenta maior dificuldade para ser especificado?

- () Descrição Geral do Elemento de Interface
- () *Parts* e Interações do Elemento de Interface
- () *States* do Elemento de Interface
- () API do Elemento de Interface
- () *Listeners* do Elemento de Interface
- () *Connections* do Elemento de Interface

12- Qual seção do modelo TTUI-SM é mais simples e intuitivo de ser especificado?

- () Descrição Geral do Elemento de Interface
- () *Parts* e Interações do Elemento de Interface
- () *States* do Elemento de Interface
- () API do Elemento de Interface
- () *Listeners* do Elemento de Interface
- () *Connections* do Elemento de Interface

13- Em sua opinião, quais os pontos NEGATIVOS do modelo TTUI-SM e sua ferramenta de especificação?

14- Em sua opinião, quais os pontos POSITIVOS do modelo TTUI-SM e sua ferramenta de especificação?

15- Quais são suas sugestões de melhoria para o modelo?

16- Quais experiências foram obtidas através do experimento?

Obrigado pela participação!

Anexo D

FICHAS DO ESTUDO DE CASO VOO INTERATIVO



Interface Element: Media Frame								
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description				
4	Media Frame	2	0	Usage	Operation			
Requirements								
Parts								
Part ID	Part Name	Part Type	Part Description			Part Usage	API ID	
1	Media Frame	Frame	Shows images and videos from UAV			Tap and hold for a certain time to close it	0	
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
1	0				Closes the media frame	Gesture_TapAndHold	-	
States								
State ID	State Name	Visible Parts	Connection IDs		State Description			
1	Media Frame Visible	1			The media frame is exhibited			
Draft								
								
State ID	State Name	Visible Parts	Connection IDs		State Description			
2	Invisible				The media frame is hidden			
Draft								
API								
Listeners								
SEND	Send ID	To		When	Data Type	Action	API ID	
RECEIVE	Receive ID	From		When	Data Type	Action	API ID	
Connections								
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action			

Figura 82 - Ficha do Elemento de Interface Media Frame



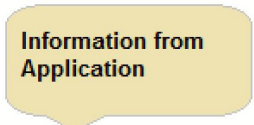

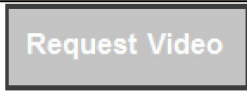
Interface Element: Map								
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description				
1	Map	1	0	Usage	Operation			
Requirements				Used as base map to the missions	Zoom In, Zoom Out, Pan			
Parts								
Part ID	Part Name	Part Type	Part Description		Part Usage	API ID		
1	Map	Image	Map Image of the Mission		Map Image	0		
Draft								
								
Interactions								
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation
1	0				PAN the map	Gesture_Drag_OneFinger	-	
2	0				Zoom In Map	Gesture_PinchOut	-	
3	0				Zoom out the map	Gesture_PinchIn	-	
States								
State ID	State Name	Visible Parts	Connection IDs		State Description			
1	Map on	1			Map visible			
Draft								
API								
Listeners								
SEND	Send ID	To	When		Data Type	Action	API ID	
RECEIVE	Receive ID	From	When		Data Type	Action	API ID	
Connections								
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action			
1	2	Passive	0	0	Connects the map with the interface element			
2	3	Passive	0	0	Connects the map to the Waypoint			

Figura 83 - Ficha do Elemento de Interface Map

Interface Element: UAV									
Interface Element ID	Interface Element Name	Initial State	Owner ID	Element Description					
2	UAV	1	0	Usage	Operation				
Requirements				Represents the UAV			Tap to show informations about the UAV, Double tap to open UAV Menu		
Parts									
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
1	UAV	Image	Representation of the UAV			Tap to open information window		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
1	2				Open information window	Gesture_Tap	-		
5	3				Opens UAV Menu	Gesture_DoubleTap	-		
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
2	Information Window	Frame	Shows informations about the UAV			Tap to close window		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
2	1				Close the information window	Gesture_Tap	-		
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
3	UAV Menu Image	Menu Option	Option to request a picture from UAV			Tap to request a picture frmo UAV		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
3	1				Sends an image request to UAV	Gesture_Tap	-		
Part ID	Part Name	Part Type	Part Description			Part Usage		API ID	
4	UAV Menu Video	Menu Option	Sends a video request to UAV			Tap to request video from UAV		0	
Draft									
									
Interactions									
Interaction ID	State ID	Send ID	API ID	Connection IDs	Interaction Action	Interaction Type	Object ID	Orientation	
4	1				Sends a video request to UAV	Gesture_Tap	-		
States									
State ID	State Name	Visible Parts		Connection IDs	State Description				
1	UAV	1			Just the UAV is visible				
Draft									



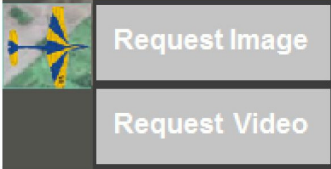
						
State ID	State Name	Visible Parts	Connection IDs	State Description		
2	UAV Info	1,2		The information window is visible with UAV		
Draft						
						
State ID	State Name	Visible Parts	Connection IDs	State Description		
3	UAV Menu	1,3,4		Shows the UAV Menu		
Draft						
						
API						
Listeners						
SEND	Send ID	To	When	Data Type	Action	API ID
	1	UAV	UAV information window is opened	String	Requests UAV data	0
	2	UAV	User taps UAV Menu Image	String	Requests a picture to UAV	0
	3	UAV	User taps UAV Menu Video	String	Sends a video request to UAV	0
RECEIVE	Receive ID	From	When	Data Type	Action	API ID
	1	Real UAV	The Real UAV sends information data	String	Receives UAV information data	0
Connections						
Connection ID	Interface Element ID	Connection Type	Interface Element State ID	API ID	Action	
1	4	Active	1	0	Opens the Media Frame	

Figura 84 - Ficha do Elemento de Interface UAV