UNIVERSIDADE DO VALE DO RIO DOS SINOS
CIÊNCIAS EXATAS E TECNOLÓGICAS
PROGRAMA INTERDISCIPLINAR DE PÓS-GRADUAÇÃO
EM COMPUTAÇÃO APLICADA

Dario Fernandes Franz

Exploração do Ambiente de Computação Móvel MHolo no Desenvolvimento de Aplicações

São Leopoldo
2006

Dario Fernandes Franz

Exploração do Ambiente de Computação Móvel MHolo no Desenvolvimento de Aplicações

Dissertação submetida à avaliação como requisito parcial para a obtenção do grau de Mestre em Computação Aplicada

Orientador: Prof. Dr. Gerson Geraldo H. Cavalheiro

São Leopoldo
2006

CIP — CATALOGAÇÃO NA PUBLICAÇÃO

Franz, Dario Fernandes

Exploração do Ambiente de Computação Móvel MHolo no Desenvolvimento de Aplicações / por Dario Fernandes Franz. — São Leopoldo: Ciências Exatas e Tecnologicas da UNISINOS, 2006.

70 f.: il.

Dissertação (mestrado) — Universidade do Vale do Rio dos Sinos. Ciências Exatas e Tecnológicas, Programa Interdisciplinar de Pós-Graduação em Computação Aplicada, São Leopoldo, BR–RS, 2006. Orientador: Cavalheiro, Gerson Geraldo H.

1. Computação Móvel. 2. Aplicações. 3. Linguagens de Programação. I. Cavalheiro, Gerson Geraldo H. II. Título.

UNIVERSIDADE DO VALE DO RIO DOS SINOS
Reitor: Dr. Marcelo Fernandes de Aquino
Diretora da Unidade de Pesquisa e Pós-Graduação: Profª. Drª. Ione Bentz
Coordenador do PIPCA: Prof. Dr. Arthur Tórgo Gómez

"Aos que me podem ouvir eu digo: 'Não desespereis!'
A desgraça que tem caído sobre nós não é mais do que o produto da cobiça em agonia,
da amargura dos homens que temem o avanço humano..."
Charles Chaplin

Agradecimentos

O ambiente de pesquisa disponibiliza uma elevada troca de idéias e as interações entre alunos e professores proporcionam um aprendizado inigualável. Desta forma, agradeço a todos que direta e indiretamente contribuíram com este aprendizado: professores, colegas e amigos.

Com especial atenção, deixo meu agradecimento ao amigo, professor e orientador Dr. Gerson Geraldo H. Cavalheiro, que em vários momentos possuía a palavra para desenrolar o que existia e gerava motivação para concluir esta caminhada.

Agradeço também a três amigos e colegas, Fernando, Marcelo e Daniel, por: estudos em cima da hora, sacanagem quando tudo parecia perdido e o companheirismo acima de tudo.

Não posso deixar de agradecer a toda minha família, ao meu cunhado Cesar e minha irmã Roberta, por terem me acolhido na hora em que eu mais precisei; aos meus pais, por terem me escutado quando eu precisava; e a minha avó, por sempre ter me ajudado com os estudos. Sem minha família eu não chegaria aonde cheguei.

Agradeço também à UNISINOS pelo apoio institucional ao trabalho, e à HP pelo apoio financeiro ao projeto.

Por fim, agradeço a inspiração do Prof. Dr. Jorge Barbosa, com a proposta do Projeto MHolo.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

CF	Compact Framework
CSBC	Congresso da Sociedade Brasileira de Computação
HML	Holo Modeling Language
Holo	Holoparadigma
HS	History Server
HVM	Holo Virtual Machine
MHolo	Mobile Holo
PDA	Personal Digital Assistant
SEMISH	Seminário Integrado de Software e Hardware
URL	Uniform Resource Locator
VM	Virtual Machine
VS	Visual Studio

Resumo

O avanço do poder computacional de dispositivos móveis e a popularização destes equipamentos, tem alavancado a demanda por uma nova classe de aplicações: *aplicações móveis*. Com isso, surgem novos modelos de programação que buscam explorar um novo cenário computacional. Com o objetivo de explorar ao máximo a característica de mobilidade dos dispositivos recém mencionados, temos como resultado, a materialização de modelos de programação em estruturas computacionais, implementadas por ferramentas de desenvolvimento e suporte à execução de software. Esta dissertação apresenta um estudo sobre estas ferramentas de suporte a execução de aplicações móveis, tendo como foco a exploração do ambiente de computação móvel MHolo. Para isso, foi construída uma aplicação móvel real, denominada *"Acompanhamento de um Evento Científico"*, que abrange mobilidade, distribuição e consciência ao contexto. O modelo para esta aplicação foi baseado em um evento científico genérico e, como aspectos de trabalho, este modelo foi mapeado e implementado para um evento científico real.

O modelo da aplicação foi concebido respeitando as abstrações para programação providas pelo Holoparadigma. Sua implementação se deu com o *.NET Compact Framework*, sendo avaliada durante o SEMISH 2005; esta dissertação documenta o modelo concebido e a respectiva implementação. Uma segunda versão do modelo também foi implementada empregando o suporte de execução provido pelas ferramentas desenvolvidas no projeto MHolo. Os resultados atingidos apontam a viabilidade das abstrações de programação providas pelo Holoparadigma para a computação móvel.

Palavras-chave: Computação Móvel, Aplicações, Linguagens de Programação.

TITLE: "Exploring the MHolo Mobile Computing Environment for Application Development"

Abstract

The growth of computational power in mobile devices and the popularization of these devices, has introduced the need of a new class of applications: *mobile applications*. As a consequence, the number of programming models which try to explore a new computational scenery. Aiming to explore the mobility of such devices at their maximum, we have as result, the materialization of programming models in computational structures, implemented by software development and execution support tools. This work presents a study about those tools that support the execution of mobile applications, focusing the exploration of MHolo mobile computing environment. For that purpose, a real mobile application was built, called *"Scientific Meeting Assistant"*, which encloses mobility, distribution and context awareness. The model for this application was based on a generic scientific meeting, and as work aspects, this model has been mapped and implemented in a real scientific symposium.

The application model was conceived respecting the programming abstractions provided by the Holoparadigm. Its implementation was accomplished with the *.NET Compact Framework*, and it was evaluated during SEMISH 2005; this work presents the proposed model and its implementation. A second version of the model was also implemented, applying the execution support provided by the tools developed in the MHolo project. The results prove the viability of the programming abstractions provided by the Holoparadigm for mobile computing.

Keywords: Mobile Computing, ApplicationsF, Programming Languages.

Capítulo 1

Introdução

1.1 Contextualização

O poder computacional de dispositivos móveis torna-se cada vez maior com o avanço tecnológico. Tal avanço, associado a popularização dos dispositivos, tem alavancado a demanda por uma nova classe de aplicações: *aplicações móveis*. O sucesso destes equipamentos depende diretamente de suas aplicações, explorado por novos modelos de computação para o novo cenário computacional que está sendo exposto. Estes novos modelos procuram explorar ao máximo a capacidade de mobilidade dos dispositivos móveis com vistas a favorecer novas estruturas para construção de aplicações típicas em ambiente de computação móvel, aqui referenciadas como aplicações móveis. Como resultado, estes modelos de computação estão sendo materializados como recursos de programação, implementados por ferramentas de desenvolvimento de software e de suporte à execução. A dificuldade que se apresenta é como explorar estes novos recursos para desenvolvimento de aplicativos.

O desenvolvimento de aplicações para dispositivos móveis (PDA, laptops, etc) está voltado a um cenário que abrange mobilidade, distribuição e consciência ao contexto em que se encontram. Dentre as abstrações possíveis para construir uma aplicação, as duas principais são [1]: *mobilidade física*, refletindo a relação com o deslocamento de peças de software entre nós de uma arquitetura distribuída; e *mobilidade lógica*, refletindo a relação com o deslocamento de peças de software sem considerações sobre a plataforma de execução. O principal objetivo da integração destas abstrações é a transparência para o usuário final sobre a mobilidade. Com o propósito de abordar estas abstrações para mobilidade, está em desenvolvimento o *Projeto MHolo* (*Mobile Holo*), o qual objetiva promover suporte a mobilidade física e lógica, bem como de desenvolvimento de aplicações para a computação móvel na *Hololinguagem* [2].

O MHolo se propõe a discutir o emprego destes recursos, em termos de facilidades de modelagem e descrição dos fenômenos reais associados à mobilidade sob a forma de programas computacionais, explorando o desenvolvimento de sistemas voltados para à

computação móvel. O ambiente MHolo se compromete com os seguintes elementos: (i) linguagem Holo, uma linguagem de programação concorrente que permite a especificação de aplicações através dos conceitos do *Holoparadigma* [3] (de forma resumida, Holo), (ii) *HML* (*Holo Modeling Language*), uma linguagem de modelagem que fornece programação visual em Holo, (iii) *HoloCase*, uma ferramenta case que suporta a especificação de aplicações (através da HML) e geração automática de código em Holo, (iv) *HoloEnv*, um ambiente de desenvolvimento integrado (IDE) que permite edição, compilação e execução de programas Holo e (v) *HoloVM* [4], uma máquina virtual que suporta a execução de programas nativos em Holo.

O atual estágio do projeto envolve o desenvolvimento de aplicações móveis com utilidade prática real, tirando proveito da maturidade atingida com o desenvolvimento de Holo. O trabalho proposto aborda a modelagem e desenvolvimento de aplicações utilizando os novos recursos de programação propostos pelo MHolo, permitindo avaliar a infra-estrutura fornecida pela HoloVM e das demais ferramentas disponíveis. Particular atenção é dada à exploração dos recursos de Holo para descrição das necessidades de mobilidade das aplicações e de seu suporte operacional.

1.2 Definição do problema

A expansão do interesse em computação móvel tem motivado o surgimento de uma nova classe de aplicações. Novos ambientes têm sido projetados para oferecer suporte à construção destas aplicações. A questão que se coloca é como explorar estes novos recursos de programação na solução efetiva de programas reais. São caracterizados os recursos oferecidos por alguns destes ambientes para suportar a mobilidade das aplicações e como estes podem ser explorados na prática. O presente trabalho se propõe a discutir o emprego destes recursos em termos de facilidades de modelagem e descrição dos fenômenos reais associados à mobilidade sob a forma de programas computacionais. O foco é dado à exploração de ferramentas computacionais oferecendo maiores facilidades para descrição da mobilidade.

A introdução de mobilidade às aplicações implica no tratamento de um novo grau de abstração no desenvolvimento de código. As propostas de ambientes e/ou linguagens existentes oferecem soluções em diferentes níveis. Algumas abordagens focam aspectos específicos como adaptabilidade ao meio [5, 6], outras utilizam, de alguma forma, uma máquina virtual para suportar a mobilidade [3, 7]. Também identificam-se abordagens oferecendo ambientes de desenvolvimento que disponibilizam suporte apenas ao dispositivo móvel [8], um suporte de mais baixo nível de programação. Assim como existe uma diversidade de abordagens para soluções, existe também diversas implementação destes ambientes.

Considerando explorar o ambiente MHolo em dispositivos móveis, segue a

necessidade de identificar e explorar as abstrações necessárias ao desenvolvimento de aplicações móveis construíndo efetivamente a aplicação.

1.3 Objetivos

Este trabalho tem como objetivo principal explorar o ambiente de computação móvel MHolo, no desenvolvimento de aplicações. Procurando explorar este ambiente e seus componentes, foi desenvolvido uma aplicação móvel distribuída, que abrange mobilidade, distribuição e consciência ao contexto. Como aspectos de trabalho, consideram-se questões ligadas ao campus universitário, tendo como objetivo desenvolver uma aplicação voltada a permitir o acompanhamento de um evento científico por diferentes classes de atores (congressistas, coordenadores de sessão, coordenador geral, e participantes). Os principais aspectos desta aplicação são: (i) o controle e (ii) a troca de informações entre os participantes deste evento. A aplicação pode ser caracterizada como um "*Acompanhamento de um evento científico*". Esta aplicação explora uma infra-estrutura de rede sem fio e dispositivos computacionais móveis, tendo por finalidade avaliar os recursos de programação MHolo. Os objetivos específicos do trabalho, são:

- *Proposição do modelo e caracterização da aplicação*: dedica-se a modelagem de uma aplicação. Sendo o desenvolvimento do modelo de uma aplicação real, com o suporte ao controle e acompanhamento de um evento científico, bem como a modelagem dos entes e suas relações;

- *Desenvolvimento de uma aplicação*: implementar a aplicação proposta, considerando o modelo do Holoparadigma construído. Neste contexto, a construção da interface com os atores propostos, aliado a definição e inserção de informações sobre localização na aplicação e implementação da aplicação com a análise de resultados;

- *Exploração da nova implementação da HoloVM e documentação dos resultados*: implementar uma aplicação utilizando a HoloVM disponível para os dispositívos móveis. Para tal, a documentação do modelo construído e a generalização deste para outras aplicações de características semelhantes;

- *Avaliação prática*: a realização do CSBC 2005 na Universidade do Vale do Rio dos Sinos permitirá uma avaliação prática do software desenvolvido. A utilização da aplicação pelos atores permitirá observar a aplicabilidade do software realizado.

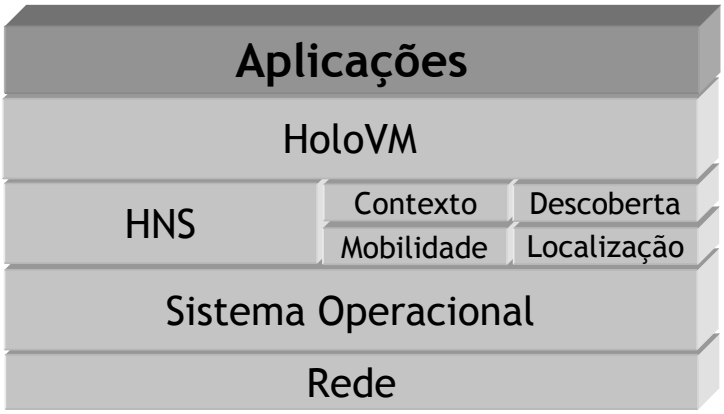


FIGURA 1.1 – Ambiente de programação MHolo

1.4 Metodologia

O presente trabalho foi desenvolvido como atividade conjunta do projeto MHolo. As etapas de trabalho refletem a construção de uma aplicação real em ambiente de computação móvel. No contexto do projeto MHolo, o desenvolvimento destas etapas visam avaliar o potencial do modelo de programação proposto pelo Holoparadigma para construção de aplicações móveis. A Figura 1.1 apresenta um esquema do ambiente de programação proposto por MHolo, com destaque a camada de aplicação, camada na qual o presente trabalho foi desenvolvido.

A primeira etapa desenvolvida no trabalho buscou caracterizar a computação móvel e identificar os recursos de programação empregados por alguns dos ambientes de desenvolvimento de aplicações móveis encontrados na literatura. A seqüência se deu pelo estudo das abstrações de programação providas pelo Holoparadigma e da análise do estado de implementação de seu suporte de execução.

Durante a segunda etapa foi identificada a aplicação real sobre a qual se deu o desenvolvimento da implementação: uma ferramenta para auxílio ao controle de um evento científico. Para esta aplicação foi gerado um modelo genérico caracterizando as necessidades de mobilidade dos elementos envolvidos. Deste modelo foi concebida sua estrutura considerando as abstrações do Holoparadigma e realizada sua implementação considerando um evento real: SEMISH 2005.

A terceira etapa marcou a avaliação da implementação realizada. Nesta etapa, o software desenvolvido foi utilizado durante a realização do SEMISH 2005. Considerações sobre o uso deste software e das abstrações providas pelo Holoparadigma foram reportadas ao restante da equipe do projeto MHolo com vistas à agregar valor ao ambiente de execução de MHolo em desenvolvimento.

A quarta e última etapa desenvolveu uma nova versão da aplicação sobre a infra-estrutura provida pelo ambiente MHolo

1.5 Organização da proposta

O restante desta dissertação está organizado como segue. O Capítulo 2 descreve alguns termos utilizados na dissertação e exemplifica aplicações em computação móvel. A Seção 2.5 ilustra alguns ambientes de computação móvel, seus projetos e conceitos, identificando suas propostas de suporte à mobilidade. O Capítulo 3 é dedicado ao Holoparadigma, sendo descritos seus principais conceitos e caracterizando estratégias utilizadas para sua implementação através de sua plataforma de execução a HoloVM. Também, neste capítulo, algumas aplicações, Seção 3.4, são modeladas e desenvolvidas. O objetivo é explorar os recursos de programação propostos pelo paradigma através da construção de aplicações móveis, abordando desde suas especificações até sua implementação em Holo. O Capítulo 4 descreve a aplicação e o modelo proposto. O desenvolvimento encontra-se no Capítulo 5. A dissertação é encerrada com os resultados obtidos e considerações finais sobre o trabalho, apresentados no Capítulo 6.

Capítulo 2

Computação Móvel

Este capítulo tem o objetivo de descrever alguns termos utilizados e, sobretudo, exemplificar o principal foco desta dissertação, aplicações em computação móvel. Na seqüência, apresentamos alguns ambientes de desenvolvimento de aplicações móveis encontrados na literatura.

2.1 Distribuição e Mobilidade

Os sistemas distribuídos tradicionais são constituídos por uma coleção de nodos fixos, com possibilidade de conexão permanente à rede, e apresentam uma configuração cujas variações são poucas no tempo e/ou espaço. Tais sistemas são implantados de forma a atingir requisitos como: escalabilidade, tolerância a falhas, heterogeneidade e compartilhamento de recursos.

Já os sistemas distribuídos com suporte à mobilidade exploram, em grande parte, as estruturas já existentes nos sistemas distribuídos tradicionais. Efetivamente, mudando a construção das soluções para os problemas, em geral advindos da mobilidade física dos usuários e/ou dispositivos, como adaptação ao contexto e o modo de acesso aos dados. Entretanto, também são potencializados os fatores geradores de problemas. A mobilidade física, dentre outros aspectos, introduz um comportamento não-determinístico como a conexão e desconexões às redes existentes e, conseqüentemente, presença ou não de nós.

Considerando a evolução dos sistemas distribuídos no sentido da computação móvel, aliado a evolução dos dispositivos portáteis, e, em decorrência a integração de seus conceitos, requisitos e definições não estarem estabelecidos, o Projeto MHolo considera o surgimento de um cenário computacional, a *Computação Móvel*.

TABELA 2.1 – Tipos de aplicações móveis

Aplicações conscientes da rede (*network-aware*)	Adaptam-se aos recursos disponibilizados pela infra-estrutura de rede. Seu objetivo é a continuidade do serviço, apesar de perdas em componentes do resultado final.
Aplicações conscientes dos recursos (*resource-aware*)	As aplicações buscam identificar e empregar os recursos disponíveis.
Aplicações conscientes da localização (*location-aware*)	Estas aplicações empregam a informação de localização do usuário como referência para refinar a execução de ações, bem como para fornecer informações sobre o que está próximo ao usuário.
Aplicações conscientes do contexto (*context-aware*)	O contexto caracteriza informações sobre o estado dos dispositivos envolvidos no processamento, sobre os recursos de rede, sobre a localização do usuário, dentre outros.

2.2 Computação Móvel e Aplicações

O conceito de Computação Móvel tem sido proposto para suportar diferentes tipos de aplicações, dentre elas: gerenciamento de equipamentos de grande porte; integração com o ambiente/meio em que se encontra; controle de estoque; acesso a informações distribuídas; e gerenciamento de redes. Computação Móvel tem sido considerado um conceito que pode ser explorado para fornecer, entre outros, os seguintes benefícios: melhor uso de recursos de comunicação (em termos de distribuição e mobilidade); suporte flexível a operações desconectadas; flexibilidade no gerenciamento de equipamentos externos; e suporte adequado para interações com usuários.

Embora exista a proposta de utilizar a Computação Móvel para vários tipos de aplicações, o escopo de aplicabilidade deste conceito será determinado dependendo do desenvolvimento de soluções apropriadas para um conjunto de questões. Parte destas questões está relacionada à elevada procura e desenvolvimento de aplicações móveis. Usuários ou organizações utilizando computação móvel para executar aplicações em um ambiente distribuído devem ter controle sobre a sua execução. Dois dos aspectos relacionados ao desenvolvimento de aplicações móveis, que serão considerados no contexto deste trabalho, são: suporte a mobilidade física e lógica e a geração de implementações de aplicações baseadas nos conceitos de computação móvel em MHolo. Outros aspectos são segurança, suporte à confiabilidade de execução das aplicações e gerenciamento dos as aplicações, estando estes em desenvolvimentos no Projeto MHolo.

Na Tabela 2.1 são resumidos os principais tipos de aplicações móveis identificados na literatura. Este estudo foi centrado nos ambientes de computação móvel (Capítulo 2.5), o qual buscou identificar ambientes de programação móvel e suas formas de adaptações e aplicações.

Além disso, existem diversos requisitos não-funcionais e independentes de aplicação sendo perseguidos em sistemas de computação móvel, sendo eles:

- *tipo de nodo.* Nos sistemas distribuídos móveis, os nodos têm a propriedade de portabilidade e mobilidade, diferente dos nodos fixos dos sistemas distribuídos tradicionais;

- *tipo de conexão de rede.* Usualmente, os nodos fixos estão permanentemente conectados à rede de comunicação. Por outro lado, nodos móveis operam em modo desconectado na maior parte do tempo. Isto se deve a dois fatores: (i) para economia de energia, que provém da bateria que em geral tem poucas horas de duração. Desta forma, diz-se que a conexão no ambiente móvel é intermitente e que a desconexão é planejada; (ii) o meio sem fio é altamente propenso a interferências ambientais que causam a interrupção abrupta da conexão;

- *tipo de contexto de execução.* O contexto, que inclui recursos internos e externos à aplicação, como tamanho da tela, memória disponível, largura de banda, localização, pode influenciar seu comportamento. Em sistemas tradicionais, este ambiente é mais ou menos estático: banda é alta e contínua, localização dos nodos é conhecida, nodos podem ser adicionais ou removidos mas isto acontece com pouca freqüência. Em oposição, o ambiente móvel é extremamente dinâmico.

Existem várias abstrações para a forma de adaptação em aplicações móveis. Muitos fatores são relevantes, dentre estes, o projetista da aplicação depende diretamente do cenário computacional abordado. Contudo, o suporte à programação pode oferecer abordagens específicas para alguns domínios de aplicações, sistemas tais como: Odyssey [5], one.world [7] e MHolo [3], ou buscando atingir uma forma genérica, tais como: Context Toolkit [6].

2.3 Mobilidade Física e Lógica

De forma genérica, um cenário que envolve computação móvel é aquele onde alguns dos nodos envolvidos na computação, são móveis[9]. Em um ambiente como este, é possível identificar dois perfis para usuários, separando estes em grupos bem definidos. No primeiro, os usuários são basicamente nômades, ou seja, suas conexões de rede ocorrem em locais e momentos totalmente arbitrários. No segundo grupo se encontram os usuários que estão sempre conectados, utilizando-se para isto de redes sem fio. Desta forma é possível identificar algumas propriedades da computação móvel, são elas: mobilidade, portabilidade e conectividade[10].

A natureza essencialmente dinâmica de hardware e software, em ambientes de computação móvel, cria a necessidade de soluções para questões como a identificação física de nodos quanto a localização de componentes de software. Esta questão levanta a

necessidade de mecanismos que conheçam a localização de componentes móveis de modo a permitir a interação com os mesmos. Assim, o ambiente se propõe a oferecer suporte a computação móvel, deve levar em conta estas questões, disponibilizando serviços que permitam as aplicações não perderem consistência quando um nodo migrar na rede, ou mesmo quando um componente de software migrar para outro nodo para continuar sua execução.

2.4 Adaptação

A idéia de sistemas que se adaptam ao seu ambiente não é nova. Na verdade, mobilidade implica em adaptabilidade[11], ou seja, sistemas devem ter consciência do contexto aonde estão inseridos e tirar proveito desta informação estruturando-se de modo distribuído e reconfigurando-se dinamicamente. Os desafios colocados pela mobilidade[10] ainda são foco de pesquisa e modelos, arquiteturas e tecnologias se encontram em desenvolvimento.

Em resposta a estes desafios, a comunidade científica vem desenvolvendo trabalhos que têm por objetivo resolver estas questões[12]. Desta forma surgiram algumas soluções que contemplam a gerência de aplicações com suporte à mobilidade lógica e/ou física (software e/ou hardware). Analisando estas soluções, alguma características em comum podem ser observadas[13]: (i) atuam gerenciando as larguras de banda utilizadas na comunicação; (ii) se concentram em um domínio específico de aplicação; (iii) normalmente implementam apenas uma estratégia de adaptação: alteração no formato dos dados, replicação de dados ou migração de tarefas.

2.5 Ambientes de Computação Móvel

Os ambientes de aplicações móveis, em geral, fornecem uma adaptação de forma específica a um domínio de aplicação. Como conseqüência, a opção por um determinado domínio reflete decisões de projeto aos recursos de mobilidade. Nas subseções seguintes são apresentados alguns destes ambientes de computação móvel.

2.5.1 Odyssey

Odyssey [14, 15, 5], desenvolvido em colaboração com a Universidade de Michigan, oferece suporte à execução de aplicações sensíveis à variação do desempenho do meio de comunicação. O Odyssey é uma plataforma que prevê acesso a dados móveis, construída para explorar adaptações em *consciente de rede*. Este ambiente propõe um modelo de adaptação através de um conceito de *fidelidade de dados*, sendo sua adaptação alcançada pela troca do modo como a rede está sendo utilizada.

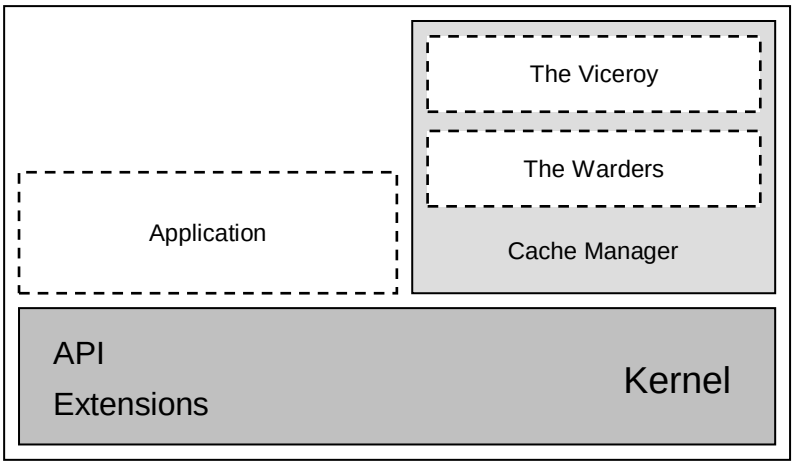


FIGURA 2.1 – Arquitetura do ambiente Odyssey [5]

Odyssey prevê seu uso em aplicações com um comportamento adaptativo. Esta adaptação é definida como uma troca entre qualidade de dados e consumo de recursos, abordando aspectos dirigidos ao acesso de dados. O projeto possui uma arquitetura baseada em cliente servidor. O componente cliente pode ser representado por uma adaptação de uma aplicação já existente no mercado, como um navegador web ou uma aplicação de vídeo. O componente servidor pode ser representado por um repositório contendo várias instâncias de informação, as quais encontram-se replicadas com resoluções diferentes de dados.

A arquitetura do cliente possui dois mecanismos de controle para uma aplicação, apresentados na Figura 2.1 e descritos a seguir:

- *Viceroy*: Responsável pelo monitoramento e gerenciamento do uso de recursos. É responsável por notificar a disponibilidade do recursos aos clientes, e controlar o uso de recursos das diversas aplicações;

- *Wardens*: Possui métodos de acesso e funcionalidade específicas. Por exemplo, implementadas no servidor para o atendimento das necessidades de cada aplicação, gerenciamento da comunicação com o servidor e manipulação da fidelidade dos dados em uma transmissão de vídeo.

Um elevado custo computacional é associado ao desenvolvimento de novas aplicações e novas adaptações para o Odyssey [5]. Este custo é destinado a criação de *Warden* e suas políticas de adaptação, únicas para cada aplicação gerada. O Odyssey, não possui recursos para ativação de mobilidade de peças de software. Seus recursos limitam-se a permitir uma adaptação através de sensores, voltada a troca de qualidade de dados e consumo de recursos. Esse recurso, na prática, pode ser explorado pela mobilidade de nós

computacionais em uma rede atrevés da manipulação explicita dos sensores construídos para cada aplicação. Como trabalhos futuros deste projeto [5], existe a necessidade de suporte a novos tipos de dados, facilitando sua portabilidade, aliado a novos dispositivos de controle para o conceito de fidelidade de dados.

2.5.2 Context Toolkit

O Context Toolkit [6, 16] é um projeto desenvolvido pelo grupo *Future Computing Environments da Georgia Institute of Technology (GeorgiaTech)*. A principal idéia deste projeto é criar um conjunto de ferramentas que facilitem o desenvolvimento de aplicações móveis sensíveis ao contexto.

O projeto Context Toolkit [6, 17] considera que os pontos mais importantes para a sensibilidade ao contexto estão associados a coleta e a atualização de informações contextuais. O objetivo é de facilitar a interação entre o usuário e o sistema, oferecendo características de um ambiente de execução de aplicações móveis sensíveis ao contexto dispondo de um conjunto de serviços para desenvolvimento de aplicações.

A arquitetura do Context Toolkit é visualizada na Figura 2.2, nas quais identifica-se três abstrações principais:

- *Widget*: utilizando o mesmo conceito dos *Widgets* de interfaces gráficas, foram criados os *Widgets* do Context Toolkit. Em uma GUI, o *Widget* faz a mediação entre o usuário e a aplicação. No Context Toolkit, o *Widget* faz a mediação entre o usuário e o ambiente. Esse componente é responsável por encapsular informações de uma forma específica ao contexto. Além disso, os *Widgets* fornecem uma interface comum para as aplicações que utilizem um mesmo contexto, independente dos sensores utilizados para captar a informação;

- *Aggregator*: Os *Aggregators*, também chamados de Servidores de Contexto, são responsáveis por coletar as informações de contexto que se referem a uma entidade do mundo real, como uma pessoa ou um lugar. Um *Aggregator* recebe dados de todos os *Widgets* que forneçam informações relevantes a uma determinada aplicação, e concentrando-as em um mesmo lugar;

- *Interpreters*: Os Interpretadores de Contexto são responsáveis por transformar uma forma de contexto em outra. Esta transformação pode ser simples, como retornar uma informação elementar, como um e-mail ou o nome de um usuário, ou mais complexa, como determinar se uma reunião está ocorrendo a partir da quantidade de pessoas em uma sala, do nível de ruído e da direção em que cada uma estiver olhando.

O Toolkit possui alguns serviços que surgem naturalmente a partir da arquitetura empregada. Os sensores são encapsulados em um conjunto de métodos comuns para

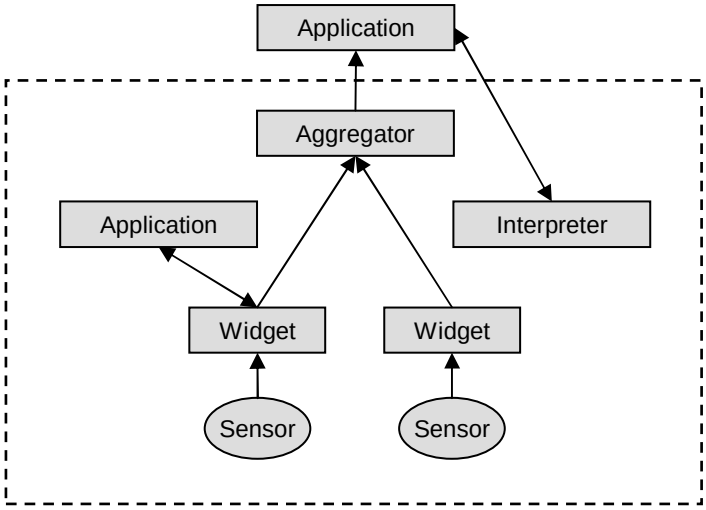


FIGURA 2.2 – Arquitetura do ambiente Context Toolkit [17]

acessá-los, independentemente do tipo de sensor. Os dados de um componente podem ser acessados via rede, de forma transparente pelas aplicações. Os dados de contexto podem ser compartilhados, já que cada *Widget* ou *Aggregator* pode ser utilizado por várias aplicações diferentes simultaneamente.

A atual implementação do Toolkit possui um serviço de Descoberta de Recursos. Uma aplicação que utilize tal serviço deve especificar ao servidor as informações de contexto que ela deseja receber (como local, nome do usuário, hora, etc.) e eventuais parâmetros (como "local = Bloco A"). A aplicação pode inscrever-se no Serviço de Descoberta para receber informações de mudanças no ambiente. A aplicação é informada se ocorre algum problema com ContextToolkit o *Widget* que ela está utilizando, ou caso surja outro componente que possa produzir as informações já especificadas. Além disso, os programadores podem utilizar o Serviço de Descoberta para identificar quais informações de contexto podem ser identificadas no ambiente. A mobilidade é suportada através de conexões e desconexões das peças de software a rede. Os mecanismos facilitadores são: o serviço de descoberta e os sensores. O foco das aplicações no Context Toolkit são aplicações conscientes de contexto. Grande parte do trabalho fica a cargo do programador que conta com o *framework* para orientá-lo.

2.5.3 one.world

O one.world [7, 18] é uma plataforma desenvolvida pela Universidade de Washington. Destina-se a *pervasive computing*, ou seja, à execução de aplicações em pequenos dispositivos móveis cooperando entre si na execução de tarefas. Este ambiente oferece um modelo de programação a ser seguido na construção destas aplicações, além de alguns serviços básicos comuns, com ênfase na mobilidade de código.

O desenvolvimento do one.world se direcionou por três princípios: expor mudanças, composição dinâmica e separação de dados e funcionalidade. Estes princípios são descritos a seguir [12, 19] :

- *Expor mudanças*: Ao invés de esconder a distribuição, a plataforma expõe a localização das aplicações e serviços, além de comunicar mudanças e falhas a estas, deixando que sejam tratadas conforme necessitem. Parte do mecanismo inclui facilitar o tratamento das mudanças e falhas pelas aplicações;

- *Composição dinâmica*: Ligações entre as aplicações podem ser feitas ou quebradas em tempo de execução, assim como permitir monitoramento ou a extensão das funções de uma aplicação;

- *Separação de dados e funcionalidade*: As aplicações devem manter seus dados distintos do código que opera sobre eles, permitindo um compartilhamento mais fácil entre aplicações diferentes. Um mecanismo para agrupar os dados e o código pode existir, mas o acesso a eles deve ser independente.

A Figura 2.3 mostra um exemplo de disposição de ambientes, aplicações e *tuplas*. Cada dispositivo executa uma cópia da plataforma one.world. As aplicações guardam seus dados em *tuplas* e seu código em componentes. Estas por sua vez ficam dispostas em ambientes, organizadas hierarquicamente. Cada aplicação tem necessariamente um ambiente onde fica seu componente principal, e pode criar ambientes filhos, componentes e *tuplas*.

Os ambientes são a unidade para os serviços de *checkpointing* e migração da plataforma. Quando estes serviços são solicitados é realizado o *checkpoint* ou migração de todo um ambiente e seus ambientes filhos. O processo para ambos é o mesmo. Primeiramente espera-se que todos os processamentos atuais nos ambientes afetados terminem para então o seu estado ser descrito de forma serializada; então o ambiente serializado ou é gravado como um *checkpoint* ou é enviado para outro dispositivo. Na recuperação o inverso acontece, o ambiente é desserializado e os componentes informados do que ocorreu para que retomem sua execução.

Todos os dados das aplicações one.world são armazenados na forma de *tuplas*. *Tuplas* podem armazenar números, *strings*, *arrays* e outras *tuplas*. Cada ambiente possui o seu espaço de *tuplas* distinto, não sendo implementado um espaço compartilhado entre ambientes no mesmo dispositivo ou dispositivos diferentes. Toda a comunicação, local ou remota, entre os componentes one.world é feita através de *eventos*. Estes eventos também são *tuplas* e indicam a origem do evento.

2.5.4 MHolo

O Holoparadigma [3, 20] (Holo) propõe um modelo multiparadigma orientado ao desenvolvimento de software distribuído. Este modelo possui como unidade de modelagem

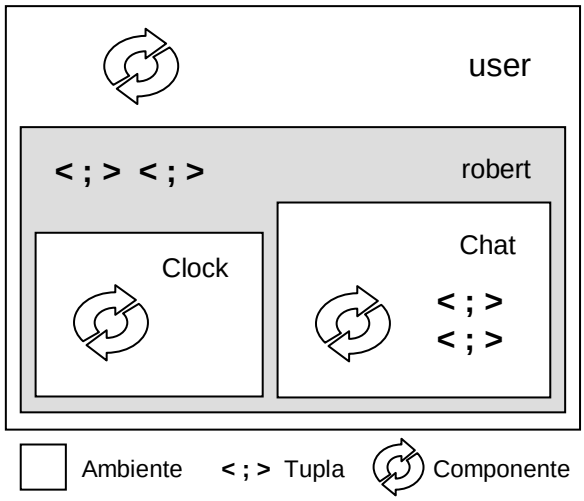


FIGURA 2.3 – Modelo de uma aplicação em one.world

o *ente* e como unidade de informação o *símbolo*. Um ente elementar é organizado em três partes: *interface*, *comportamento* e *história*. Um ente composto possui a mesma organização, no entanto, suporta a existência de outros entes na sua composição (entes componentes). Cada ente possui uma história. A história fica encapsulada no ente e, no caso dos entes compostos, é compartilhada pelos entes componentes. Sendo assim, podem existir vários níveis de encapsulamento da história. A composição varia de acordo com a mobilidade dos entes em tempo de execução.

No escopo de sistemas distribuídos, um ente pode assumir dois estados de distribuição: *centralizado* ou *distribuído*. No estado centralizado um ente encontra-se fisicamente sediado em apenas uma máquina, já no estado distribuído um ente pode criar uma forma de agregar entes executando em máquinas diferentes.

O Projeto MHolo está em desenvolvimento, com previsão de suporte às plataformas Windows, Linux e Pocket PC. O suporte para suas aplicações móveis é através de uma Máquina Virtual (VM). A HoloVM, atualmente disponível para a plataforma Linux, provê o suporte para aplicações móveis conscientes ao contexto, tendo como principal objetivo a transparência para o usuário final sobre a mobilidade.

2.5.5 .NET *Compact Framework*

O .NET *Compact Framework* (CF) [8] é uma plataforma de programação para dispositivos móveis, criada pela Microsoft. O .NET CF é uma versão reduzida do .NET *Framework*, criado para possibilitar o desenvolvimento de aplicações para *smart devices* (celulares, PDAs, etc) da mesma forma que no ambiente Windows *desktop*. Por ser um subconjunto do .NET *Framework*, o .NET CF herdou muitas de suas características e formas de execução.

Em 2002, a Microsoft disponibilizou o Visual Studio (VS) .NET, oferecendo recursos necessários para desenvolver aplicações para dispositivos móveis, unificando o ambiente de desenvolvimento em uma única ferramenta, sendo estes recursos como:

- *Templates*: Configurações pré-definidas para cada tipo de projeto, seja para Pocket PC ou Windows CE. Desta forma, o *Framework* já tem a informação para que tipo de dispositivo será compilada a aplicação;
- *Controles Específicos*: Utilizados em conjunto com Pocket PC ou Windows CE, esses controles são baseados nos controles Windows *Form*, limitando os recursos dependendo do dispositivo utilizado;
- *Emuladores*: Recursos oferecidos para o desenvolvedor testar a aplicação sem a necessidade do dispositivo. Os emuladores oferecem todos os recursos oferecidos pelos dispositivos;
- *Debugging*: Utilizada tanto no dispositivo quanto no emulador, facilitando assim para o desenvolvedor executar e testar a aplicação.

2.6 Considerações Gerais

A partir do estudo realizado neste capítulo, a Tabela 2.2 sumariza algumas das características dos ambientes relacionados. O .NET Compact Framework foi desenvolvido para prover suporte as funcionalidades de dispositivos móveis portáteis. Este ambiente não possuiu rotinas específicas de suporte às aplicações móveis, com isso suas informações foram omitidas da tabela. Destaca-se que Odyssey não oferece uma estrutura para desenvolvimento de aplicações propriamente dito, permitindo apenas níveis de adaptação à qualidade dos serviços (de rede) oferecidos. Já Context Toolkit vai além, permitindo que aplicações *conectem-se* a sensores de forma a tomar consciência do contexto em que se encontram. O one.world permite uma maior flexibilidade no desenvolvimento de software, por permitir que partes de uma aplicação migrem entre nodos, embora o usuário/programador deva ter consciência da distribuição. Por sua vez, Holoparadigma busca a integração da mobilidade física e lógica com o propósito da transparência para o usuário final sobre a mobilidade, provendo suporte a aplicações móveis.

TABELA 2.2 – Comparação dos ambientes de computação móvel

	Odyssey	Context Toolkit	one.world	MHolo
Arquitetura	*framework*	*middleware*	*middleware*	*virtual machine*
Contexto	forma genérica (sensores)	estratégia voltada para adaptação com aplicações e sensores de localização	estratégia colaborativa entre a aplicação e o *middleware* nas decisões de adaptação.	liberdade para o programador expor sua forma de contexto
Aplicações	acesso a dados, qualidade dos dados (*application-aware*)	aplicações interativas (*context-aware*)	localização, expondo a outras aplicações e serviços (*context-aware*)	localização, conciência do contexto (*context-aware*)
Adaptação	troca de qualidade de dados e consumo de recursos	presença de pessoas ou objetos em um lugar e reagem a esta presença	ações de novos eventos gerados	ações geradas
Sensibilidade da Aplicação	através de recursos	através de recursos	através de mobilidade e eventos	através de mobilidade
Mobilidade	adaptação aos serviços de rede a cada conexão	conexões aos sensores e descoberta de serviços	mobilidade de peças de software	mobilidade física e lógica, obtendo transparência para o usuário final sobre a mobilidade

Capítulo 3

Holoparadigma

Este capítulo tem por objetivo apresentar os principais aspectos do Holoparadigma (de forma abreviada, Holo) e, descrever aplicações utilizando o seu suporte de execução. Holo é um modelo multiparadigma que possui abstrações para a programação distribuída. Através destas abstrações, o modelo estimula a exploração automática da distribuição (distribuição implícita). Holo explora um mecanismo de coordenação baseado em *blackboards*. Este modelo possui como unidade de modelagem o *ente* e como unidade de informação o *símbolo*. Um maior detalhamento de Holo é apresentado em [3] onde podem ser encontrados maiores informações sobre o Holoparadigma.

3.1 Tipos de Entes

O Holoparadigma estabelece como uma unidade elementar de execução o ente. Estes entes podem ser classificados segundo sua organização e sua funcionalidade. A *classificação organizacional* distingue os entes, de acordo com a sua estrutura, em dois tipos:

- *Ente elementar*: ente sem níveis de composição;
- *Ente composto*: ente formado pela composição de outros entes. Não existe limite para níveis de composição.

Um *ente elementar* (Figura 3.1a) é organizado em três partes: interface, comportamento e história. A *interface* de um ente descreve suas possíveis relações com os demais entes. O *comportamento* contém *ações* que implementam a funcionalidade de um ente. Holo não estabelece os tipos de ações a serem utilizadas, no entanto, estabelece que existem dois tipos básicos de comportamento:

- *Imperativo*: o comportamento imperativo é composto de ações imperativas que descrevem os caminhos para solução de um problema (enfoque no controle, ou

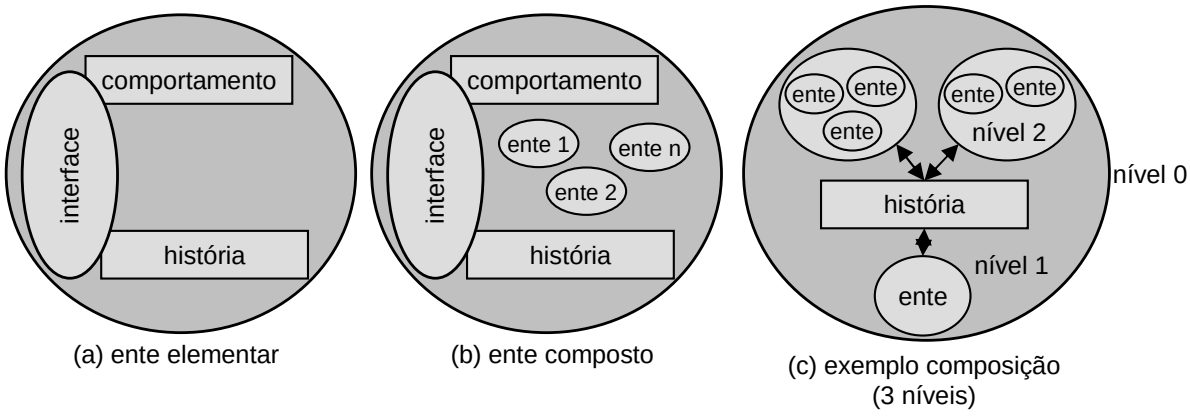


FIGURA 3.1 – Tipos de entes

seja, como deve ser realizada a ação). Uma ação imperativa possui uma natureza determinista. O paradigma imperativo é uma alternativa para descrição do comportamento imperativo;

- *Lógico*: o comportamento lógico é composto de ações lógicas que expressam um problema de forma declarativa (enfoque na lógica, ou seja, o que deve ser realizado). Uma ação lógica possui uma natureza não-determinista. O paradigma em lógica é uma alternativa para descrição do comportamento lógico.

A *história* é um espaço de armazenamento compartilhado no interior de um ente. O símbolo é o átomo de informação no Holoparadigma. Holo propõe a utilização do processamento simbólico como principal instrumento para o tratamento de informações. Esta característica é herdada do paradigma em lógica. Neste sentido, a variável lógica e a unificação são consideradas a base do tratamento de símbolos. Holo estabelece que a história deve ser direcionada para armazenamento e gerenciamento de símbolos. Portanto, o paradigma em lógica torna-se uma alternativa adequada para sua implementação.

Um *ente composto* (Figura 3.1b) possui a mesma organização de um ente elementar, no entanto, suporta a existência de outros entes na sua composição (*entes componentes*). Cada ente possui uma história. A história fica encapsulada no ente e, no caso dos entes compostos, é compartilhada pelos entes componentes. Os entes componentes participam do desenvolvimento da história compartilhada e sofrem os reflexos das mudanças históricas. Sendo assim, podem existir vários níveis de encapsulamento da história. Os entes acessam somente a história em um nível. A composição varia de acordo com a mobilidade dos entes em tempo de execução. A Figura 3.1c mostra dois níveis de história encapsulada em um ente composto organizado em três níveis. Os comportamentos e interfaces foram omitidos para simplificar a figura.

Um ente elementar assemelha-se a um *objeto* do paradigma orientado a objetos. Do ponto de vista estrutural, a principal diferença consiste na história, a qual atua como

uma forma alternativa de comunicação e sincronização. Um ente composto assemelha-se a um *grupo*. Neste caso, a história atua como um espaço compartilhado vinculado ao grupo. Lea Doug[21] salienta que os conceitos da orientação a objetos, tais como objetos e classes, tornam-se limitados quando o tratamento de composição envolve aspectos dinâmicos (mudam durante a execução). Lea também propõe a utilização de grupos para solução desta limitação. Os entes compostos aliados à mobilidade permitem a composição dinâmica no Holoparadigma. Além disso, a utilização de uma mesma unidade (ente) para manipulação de elementos e grupos, simplifica o modelo e sintetiza conceitos já existentes na ciência da computação.

A *classificação funcional* distingue os entes de acordo com suas funções:

- *Ente estático*: definição estática de um ente. Esta definição estabelece um padrão estático que pode ser utilizado para criação de outros entes através da clonagem (instanciação). Um ente estático é especificado no nível de modelagem e programação. Modelos e programas são compostos de descrições de entes (entes estáticos), as quais estabelecem interfaces, comportamentos e histórias;

- *Ente dinâmico*: ente em execução. Um programa em execução é composto de entes dinâmicos clonados a partir de entes estaticos. Estes entes executam ações e interagem de acordo com seus comportamentos e histórias.

A única distinção entre entes estáticos e dinâmicos consiste na sua função. Os entes estáticos são utilizados como matrizes estáticas para criação de outros entes. Além disso, estabelecem um estado inicial para execução de programas. Por sua vez, os dinâmicos representam o estado corrente de uma execução.

3.2 Distribuição e Mobilidade

O Holoparadigma busca a *distribuição implícita* através da Holosemântica. Uma discussão neste sentido está apresentada em [22]. Neste escopo, um ente assume dois *estados de distribuição*:

- *Centralizado*: um ente está centralizado quando se localiza em apenas um nodo de um sistema distribuído. Entes elementares estão sempre centralizados. Um ente composto está centralizado se todos os seus entes componentes estão localizados no mesmo nodo;

- *Distribuído*: um ente está distribuído quando se localiza em mais de um nodo de um sistema distribuído. Entes elementares não podem estar distribuídos. Um ente composto está distribuído se um ou mais entes componentes estão distribuídos.

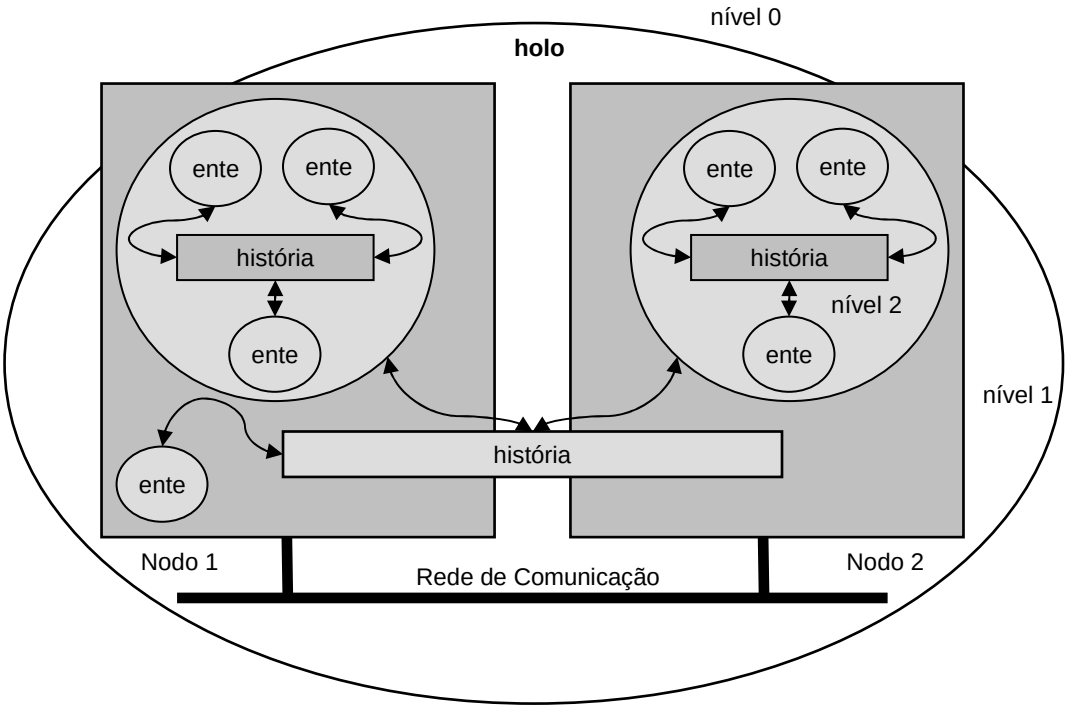


FIGURA 3.2 – Ente distribuído

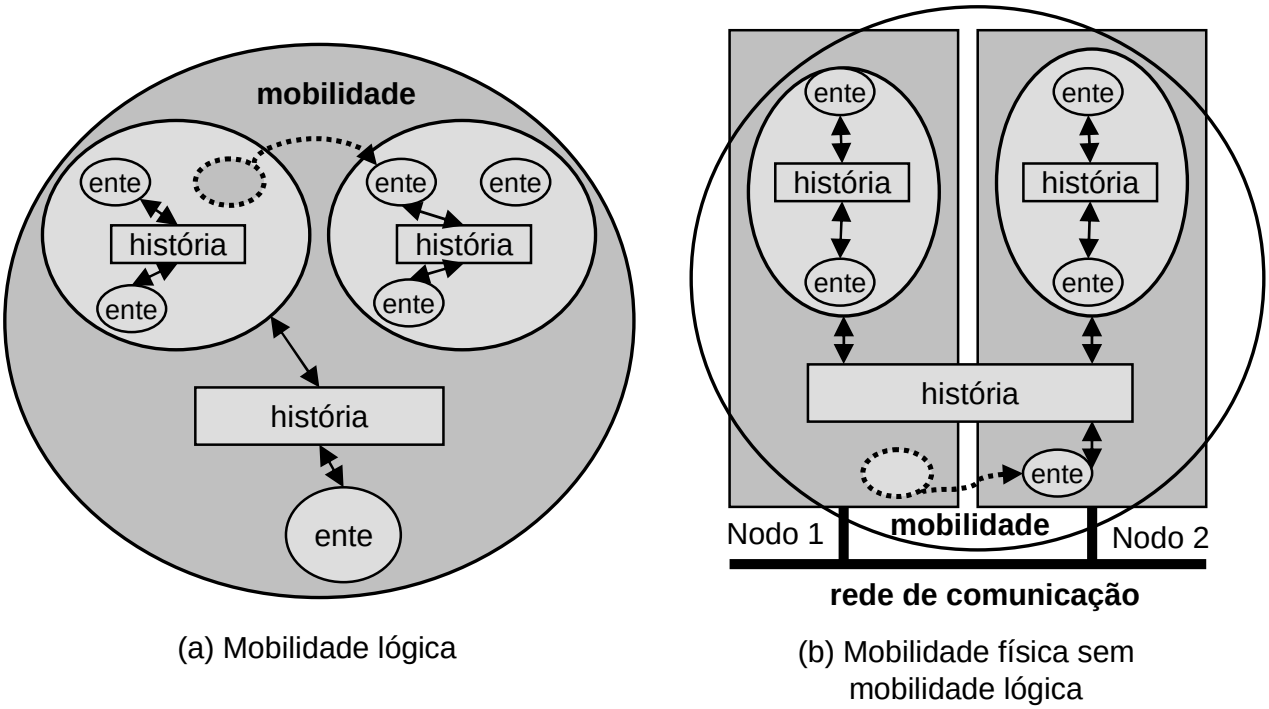
A Figura 3.2a exemplifica uma possível distribuição para o ente apresentado na Figura 3.1c. O ente encontra-se distribuído em dois nodos da arquitetura distribuída. A história de um ente distribuído é denominada *história distribuída*. A distribuição da história pode ser baseada em técnicas de *memória compartilhada distribuída* ou *espaços distribuídos*.

A mobilidade é a capacidade que permite o deslocamento de um ente. No Holoparadigma existem dois tipos de mobilidade:

- *Mobilidade lógica*: a mobilidade lógica relaciona-se com o deslocamento em nível de modelagem, ou seja, sem considerações sobre a plataforma de execução. Neste contexto, um ente se move quando cruza uma ou mais fronteiras de entes;
- *Mobilidade física*: a mobilidade física relaciona-se com o deslocamento entre nós de uma arquitetura distribuída. Neste contexto, um ente se move quando se desloca de um nó para outro.

A Figura 3.3 exemplifica uma possível mobilidade lógica no ente apresentado na Figura 3.1c. Conforme exemplificado, após o deslocamento, o *ente móvel* não possui mais acesso à história no *ente origem*. No entanto, passa a ter acesso à história no *ente destino*. Neste caso, somente ocorrerá mobilidade física se os entes origem e destino estiverem alocados em diferentes nós de uma arquitetura distribuída.

As mobilidades lógica e física são independentes. A ocorrência de um tipo de deslocamento não implica a ocorrência do outro. Merece atenção o caso da mobilidade



(a) Mobilidade lógica

(b) Mobilidade física sem mobilidade lógica

FIGURA 3.3 – Tipos de mobilidade

física não implicar obrigatoriamente a mobilidade lógica. Considere-se o ente distribuído mostrado na Figura 3.2. Se o ente elementar localizado no nível um, realizar um deslocamento físico conforme mostrado na Figura 3.3b, não haverá mobilidade lógica apesar de ter ocorrido mobilidade física. Neste caso, o ente movido continua com a mesma visão da história.

3.3 Suporte de Execução

Atualmente o Holoparadigma faz parte do projeto de pesquisa *Mobile Holo* (MHolo), o suporte a execução é fornecido por três sistemas: a *Holo Virtual Machine* (HoloVM) [4] , o *Holo Naming System* (HNS) [23] e o *History Server* (HS) [24, 25]. A HoloVM é uma máquina virtual criada para executar os programas em Holo. Como é natural de uma máquina virtual, ela cria uma camada de abstração entre o programa e o hardware sobre o qual este será executado. Isto permite que programas Holo sejam executados em qualquer plataforma, desde que exista uma versão da HoloVM disponível para a mesma. Esta possui um conjunto de instruções que foram criadas especificamente para dar suporte as funcionalidades propostas no Holoparadigma [3]. A HoloVM, por si, não é capaz de executar aplicações distribuídas. Para possibilitar que várias HoloVMs, dispostas em máquinas diferentes, sejam capazes de interagir e executar uma mesma aplicação foi criado o HNS. Este serviço permite a implementação de sistemas de computação móvel

e distribuída, onde tipicamente os programas executam de forma distribuída em diversos dispositivos fixos ou móveis, operando em conjunto para prover serviços e funcionalidades para os usuários. O HNS controla a execução distribuída de entes, conhecendo sua localização e fornecendo informações necessárias para as HoloVMs, de forma a permitir a comunicação entre entes.

Na HoloVM existe uma estrutura que é chamada de *HoloTree*. Esta estrutura mantém o controle de todos os entes em execução, bem como a suas respectivas composições. A HoloTree existente nas HoloVMs de cada dispositivo é apenas uma visão parcial do cenário, no qual apenas o servidor vai conhecer a estrutura completa da árvore, já que todas as HoloVMs reportam ao servidor quando fazem alguma alteração em suas HoloTrees. Esta estrutura mantida pelo HNS é chamada de *Distributed HoloTree* (*DHoloTree*).

Para que o servidor se mantenha sempre atualizado, com relação a execução da aplicação, algumas instruções ao serem interpretadas pela HoloVM, geram mensagens para o HNS. Uma delas é a instrução *clone*, que é responsavel pela criação de novos entes na aplicação. Ao executar esta ação a HoloVM informa ao HNS sobre o novo ente e o contexto no qual ele se encontra. Após a sua criação, um ente fica livre para se mover, ou ser movido, entre os contextos da aplicação. A mobilidade é fornecida na linguagem através da instrução *move*. Esta instrução é responsável pela dinamicidade das aplicações Holo, e seu monitoramento é especialmente importante para que o HNS tenha sempre o estado mais atualizado de cada HoloTree.

Em relação a história, a máquina virtual Holo possui um modelo de distribuição, com o objetivo de dar suporte de execução à história distribuída. Este modelo foi criado para reduzir a complexidade de gerência a partir da centralização das informações de controle dos entes e de seus dados. Dessa forma, a abordagem é centralizada e não é a mais indicada em ambientes onde a tolerância a falhas é necessária, ou pelo menos desejada. Contudo, o intuito inicial do projeto consiste em fazer testes com relação a execução de aplicações distribuídas em Holo. Com este objetivo foi criado o *History Server* (HS).

O HS é basicamente um *espaço de tuplas* que pode ser compartilhado entre várias HoloVMs. Este servidor implementa ainda o suporte a algumas características que o Holoparadigma possui, sendo o respeito às restrições de acesso à história, a principal. Estas restrições são criadas pelo encapsulamento de entes, e consistem no fato de que um ente pode acessar apenas a sua própria história e a de seu ente pai. Ao executar uma aplicação utilizando este sistema, todo acesso a história é mapeado diretamente para o servidor, ou seja, quando um ente acessa a história de seu ente pai, ou a dele próprio, a HoloVM se encarrega de mapear este acesso para uma comunicação com um HS. Este então se encarrega de manter espaços de memória independentes para cada ente criado no ambiente de execução distribuído, permitindo assim que entes dispostos em máquinas diferentes sejam capazes de se comunicar.

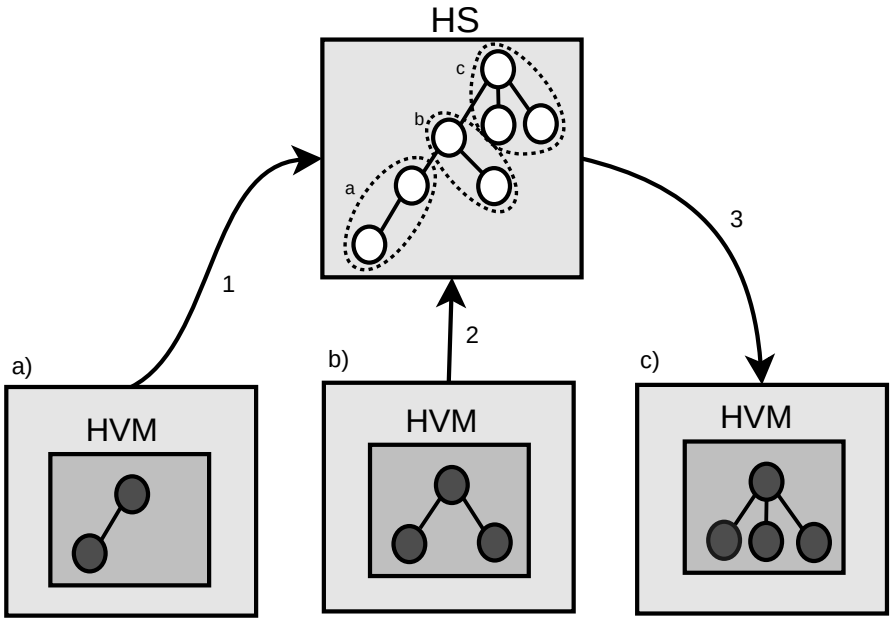


FIGURA 3.4 – Figura ilustrando funcionamento do HS

Na Figura 3.4 é mostrada a representação de um ambiente de execução utilizando o HS e as possíveis interações com uma ou mais HoloVMs. No *passo 1* a *HVM a* informa a criação de um ente para o servidor. Tendo feito isto, para que dois entes possam se comunicar basta que eles tenham acesso à história de um ente em comum. Os passos *2* e *3* exemplificam as *HVMs b* e *c*, interagindo com o servidor para acessar a história de um ente.

3.4 Aplicações

Nesta seção são descritas aplicações utilizando os recursos propostos por MHolo. Na Sub-seção 3.4.1 são caracterizadas duas aplicações sintéticas cuja estrutura pode ser reutilizada em aplicações reais. A Sub-seção 3.4.2 apresenta o desenvolvimento de uma aplicação real que está sendo realizado no contexto de *Aplicações CoolUnisinos* (um conjunto de aplicações que abordam questões ligadas ao Campus Universitário) sendo uma das linhas de pesquisa do Projeto MHolo.

3.4.1 Aplicações Sintéticas

Através de duas estruturas genéricas de programa, representadas, pela Torre de Hanói e pelo Problema de Mineração de Dados, são descritos os usos dos recursos de mobilidade lógica e física, respectivamente, em MHolo.

```
// ente principal
holo() {
  // início da execução da aplicação
  holo() {
     // criação dos entes envolvidos
     clone(disco(1),disco1),
     clone(disco(2),disco2),
     clone(disco(3),disco3),
     clone(pino,origem),
     clone(pino,ajuda),
     clone(pino,destino),
     // estado inicial
     move(disco3,origem),
     move(disco2,origem),
     move(disco1,origem),
     // gerência da mobilidade
     origem.pop(disco1),
     move(disco1,destino),
     origem.pop(disco2),
     move(disco2,ajuda),
     destino.pop(disco1),
     move(disco1,ajuda),
     origem.pop(disco3),
     move(disco3,destino),
     ajuda.pop(disco1),
     move(disco1,origem),
     ajuda.pop(disco2),
     move(disco2,destino),
     origem.pop(disco1),
     move(disco1,destino)
  }
}
// ente pino
pino(){
  pino(){
     whoami(Eu),
     writeln('Criado pino ',Eu)
  }
  pop(Ente) { // devolve disco
     move(Ente,out),
     writeln('Movendo ',Ente)
  }
}
// ente disco
disco() {
  disco(Numero) {
     whoami(Eu),
     writeln('Criado disco ',Eu)
  }
}
```

FIGURA 3.5 – Torre de hanói em MHolo

Torres de Hanói

Para exemplificar a mobilidade lógica, o código desenvolvido para resolver o problema das Torres de Hanói é apresentado na Figura 3.5. Nesta aplicação o problema é constituído de um conjunto de n discos de tamanhos diferentes e três pinos, os quais são representados por entes do tipo *disco* e entes do tipo *pino*, respectivamente. Os entes *disco*s, representados por $n = 3$ no exemplo, podem ser movidos para dentro de um ente pino, respeitando uma restrição: cada ente *disco* não pode ser movido para um ente *pino*, se no seu interior possuir um ente *disco* de menor tamanho. A configuração inicial consiste de todos os entes *disco*s (disco 1, disco 2 e disco 3) no ente pino origem. O objetivo é mover todos os entes *disco*s para o ente destino, podendo utilizar o ente ajuda e sempre obedecendo a esta restrição.

No exemplo apresentado, são utilizadas três instruções próprias à Holo: *clone*, para criação de entes; *move*, para acionar a movimentação de um ente para dentro e para fora de um contexto (outro ente), e *whoami*, para um ente ter acesso a sua própria identificação.

Esta aplicação utiliza-se da mobilidade lógica para sua resolução. Observe-se que o ente que move-se nesta aplicação é passivo nas decisões de mudança de contexto: a mobilidade é ativada por um processo de gerência de determinação de localidade. Cabe ao ente principal (identificado por *holo*) executar a lógica de determinação da localização dos discos. A idéia é alocar os discos sobre os pinos, permitindo que cada pino execute

operações sobre os discos recebidos enquanto não receber a solicitação de devolução dos discos (através da ação *pop*). Versões mais elaboradas desta estrutura básica de aplicação podem estender as funcionalidades dos *disco*s com um maior número de ações e considerar critérios de sincronização para devolução dos entes *disco*s para o contexto de gerência permitindo completar um conjunto de operações sobre cada *disco*.

Mineração de Dados

A aplicação mineração de dados é especificada por três entes: duas minas e um mineiro. O objetivo deste exemplo é ilustrar a mobilidade física realizada pelo ente mineiro – neste exemplo, as decisões de alteração de contexto de mineração são tomadas pelo ente transferido. Na história de cada mina encontra-se armazenado o dado a ser minerado bem como a identificação da mina. A mineração de dados propriamente dita é representada pelo *Cálculo de Fibonacci* do dado armazenado na história. A Figura 3.6 tem como objetivo apresentar as etapas em que o mineiro se move e minera suas minas. Essas etapas são comentadas a seguir:

- o programa é composto por quatro entes, o ente holo cria duas minas e um mineiro e distribui em duas máquinas (nodo 1 e nodo 2) e o nodo 3 possui a árvore de entes. Logo após, aguarda os resultados serem colocados na sua história (etapas 4 e 9);
- as minas (entes mina 1 e mina 2) são criadas e aguardam a mineração. A história das minas contém dois campos. O primeiro campo guarda um identificador da mina, e o segundo um número que indica qual o número de *Fibonacci* deverá ser calculado pelo mineiro;
- o mineiro se move para dentro da mina, realiza a mineração, sai da mina e insere o resultado na história de *holo*. Estes passos são executados para cada mina. A mineração consiste na busca de um valor para o cálculo de *Fibonacci*;
- o ente mineiro é responsável pelo controle de sua mobilidade (ação *move*);
- as etapas 2 e 7 do mineiro utilizam o acesso à história compartilhada ou do ente envolvente. Este acesso é realizado por uma ação à história, apesar do código ser o mesmo nas duas etapas, o *out(history)* é sensível ao contexto (mina) no qual o mineiro está em determinado momento;
- as etapas 3, 5 e 6 são execuções do comando *move*. Porém, a etapa 5 é por responsabilidade do sistema de execução, é uma mobilidade física. O ente mineiro, passa a executar no nodo 2.

Neste exemplo, ao contrário do anterior, a gerência da mobilidade é atribuída ao próprio ente que move-se entre os diferentes contextos da aplicação. O uso típico de

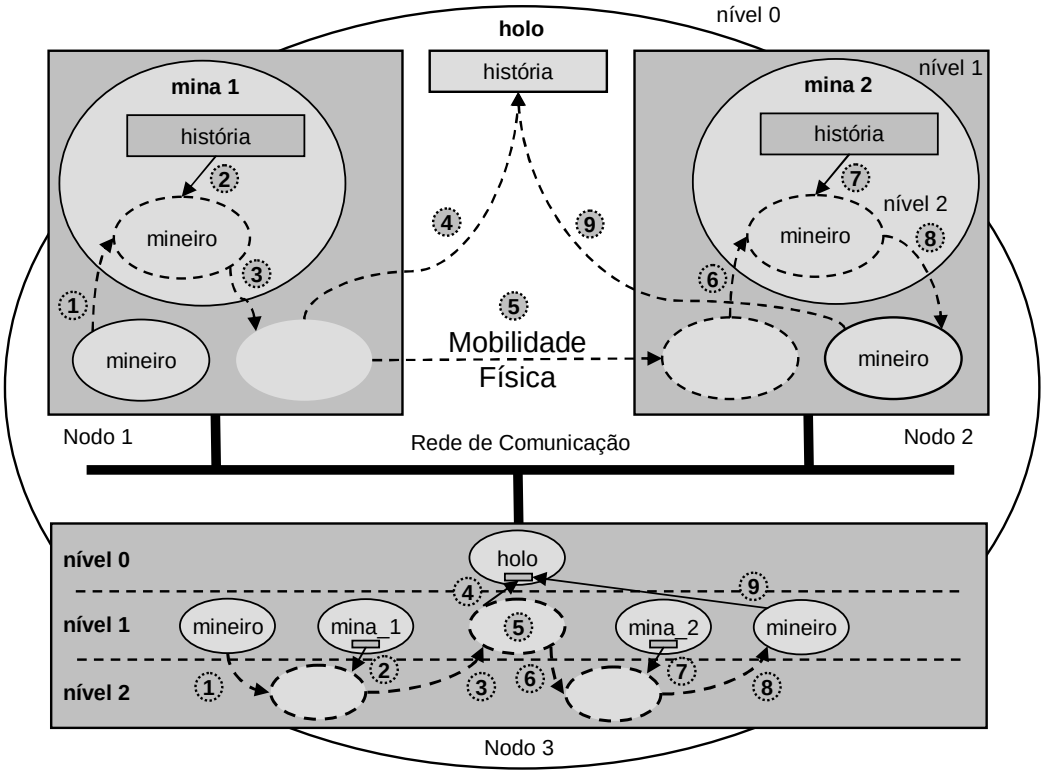


FIGURA 3.6 – Mineração de dados

tal estrutura de aplicação é viabilizar a busca de informação em diferentes contextos ou caracterizar um usuário de um dispositivo móvel.

3.4.2 Agenda Corporativa - CoolUnisinos

Para exemplificar a construção de uma aplicação real, foi desenvolvido como exemplo um *software* Agenda Corporativa, cujo o usuário possui uma visão diferente de acordo com sua localização. As passos a seguir são para exemplificar, descrever e modelar a aplicação.

Descrição da Aplicação

O modelo da aplicação está representado na Figura 3.7, onde sua composição em quatro níveis de entes é caracterizada. Nos níveis 0 e 1, considera-se um ente representando os compromissos de toda a Universidade do Vale do Rio dos Sinos e o Prédio 6B, respectivamente. O nível 2 possui entes representado os cursos da Universidade e sua agenda de eventos. Na figura são representados apenas os cursos da Engenharia da Computação (engcomp) e o Programa Interdisciplinar de Pós-Graduação em Computação Aplicada (pipca). No modelo contruído, os entes não móveis são representados por unisinos, prédio 6B, pipca e engcomp. Os entes naturalmente móveis são: aluno, professor

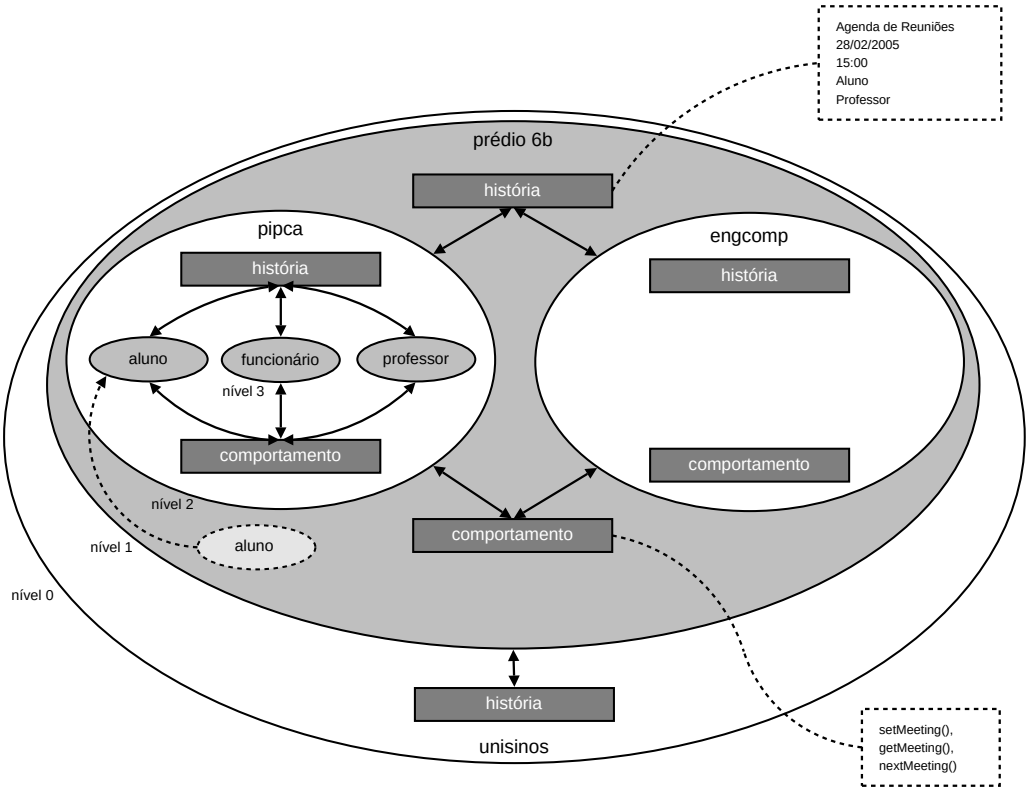


FIGURA 3.7 – Aplicação agenda corporativa

e funcionário. Em cada nível dos entes não móveis a história contém registros referentes a agenda do referido nível. O modelo mostra que de acordo com a localização em que o ente móvel se encontra, o mesmo tem acesso à história pertencente ao ente no qual está encapsulado e, assim, os compromissos naquele contexto.

O ente aluno ilustra a mobilidade entre dois níveis de entes. Quando o ente aluno se desloca para dentro do ente pipca, ele deixa de enxergar o contexto da agenda prédio 6B e passa a ter uma visão da história de sua nova localização (pipca). Através dos métodos descritos no comportamento do ente onde está inserido, tais como *getMeeting*, o aluno pode obter informações sobre seus compromissos, no caso, se informar de uma reunião que envolve ele e os professores no dia 28/02/2005 às 15:00. Desta forma o aluno pode compor sua própria agenda (na sua história) recuperando informações das agendas nos contextos visitados. Como observação deste processo, embora hierárquico, a história de um ente não absorve a história dos entes que o compõe. A não ser que o ente o faça de forma explícita.

A Figura 3.7 também representa o modelo e a execução do comando *move* sobre o ente móvel. Esta ação ocorre a partir do deslocamento de um dispositivo móvel sobre o qual executa o ente móvel. Quando o ente móvel se desloca para outro contexto, ocorre a mobilidade lógica. A possibilidade de integrar a aplicação a um recurso de localização (a ser futuramente integrado com o sistema) irá disponibilizar que o usuário se mova, e seu ente troque de contexto de forma automática, supondo que seja integrado ao sistema um

```
franz@holo:~/Projetos/MHolo/Agenda$ hvm agenda
HoloVM 2.0
 Criando o Unisinos !! Criando o Prédio 6B !!
 Criando o Prédio da engcomp !! Criando o Prédio da pipca !!
   Criando o Ente Móvel (Aluno) !!
   IPAQ:UNISINOS:PREDIO6B:  Estou dentro do Predio6B !
   IPAQ:UNISINOS:PREDIO6B:   (1) Entrar na engcomp
                             (2) Entrar no Pipca
                             (3) Ler Compromissos - Prédio6B
                             (4) Escrever Compromisso - Predio6B
                             (5) Apagar Compromisso - Predio6B
                             (6) Listar minha Agenda
                             (0) Sair da Aplicacao Agenda : 1
   IPAQ:UNISINOS:PREDIO6b:ENGCOMP: Estou dentro da engcomp !
   IPAQ:UNISINOS:PREDIO6b:ENGCOMP: (1) Ler Compromissos - engcomp
                                   (2) Escrever Compromisso - engcomp
                                   (3) Apagar Compromisso - engcomp
                                   (4) Listar minha Agenda
                                   (0) Voltar o engcomp : 0
                  Saindo da engcomp !
```

FIGURA 3.8 – Amostra da aplicação em execução

dispositivo que forneça a localização dos elementos móveis.

Modelo e Contexto da Aplicação

A definição do modelo e do contexto da aplicação está diretamente ligada com os aspectos previstos na aplicação. O programador é responsável por estas definições. Considerando a aplicação Agenda Corporativa, toda a troca de contexto está baseada na mobilidade do ente móvel. Esse usuário possui como opções uma mobilidade entre os entes considerados pela aplicação como entes no contexto do câmpus Universitário, representados pelos prédios e suas organizações internas.

A estratégia de adaptação (diferentes visões de compromissos) é realizada através do comando *move*. Para a execução deste comando sobre a estrutura do modelo foi criado um menu de opções. O menu é a forma atual de migração de um ente sobre a árvore de entes da aplicação, a aplicação em execução pode ser vista na Figura 3.8. Essas ações ocorrem com a interação direta do usuário com a aplicação. Desta forma, o usuário final pode ser visto como parte integrante da lógica de um ente móvel.

Suporte de Execução

Na Figura 3.9 é exemplificada a execução da aplicação agenda corporativa em um ambiente distribuído. Nela o ente aluno se utiliza da instrução *move* para sair do ente *pipca* e entrar no *engcomp*. Com esta operação, o ente *aluno* passa a ter acesso ao contexto de

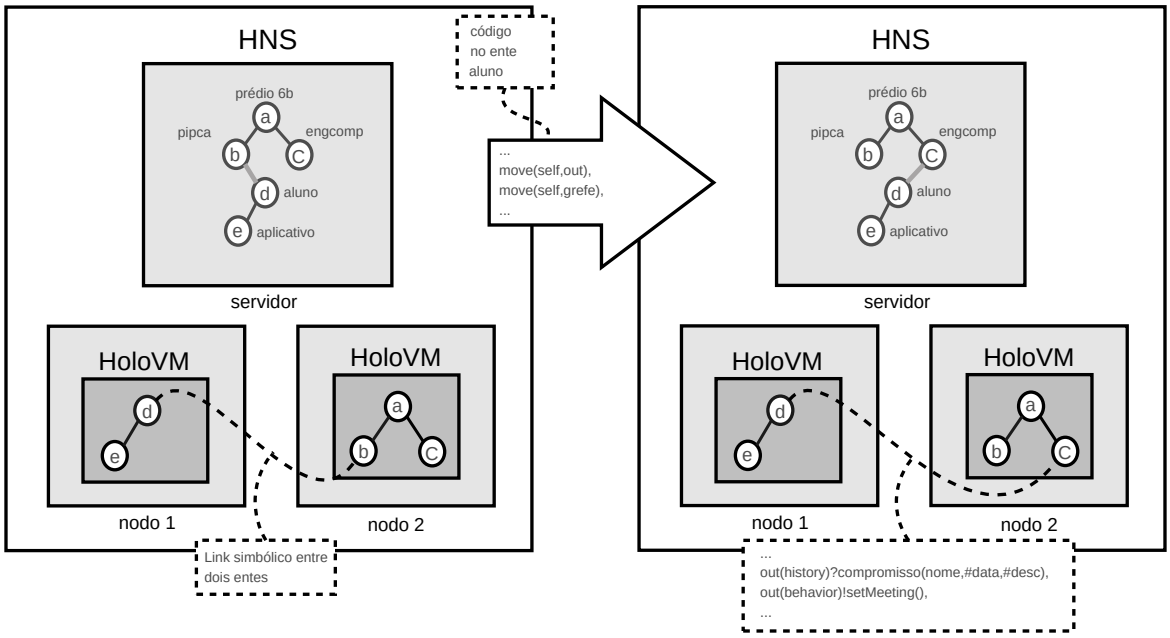


FIGURA 3.9 – Exemplo de funcionamento do HNS

engcomp, podendo assim acessar a história e executar ações de seu ente pai. Esta é uma mobilidade lógica, pois o ente troca de contexto, mas continua executando na mesma HoloVM. A mobilidade física ainda não é implementada no ambiente, porêm uma vez inserida esta característica, regras de escalomento poderão agir sobre uma aplicação para garantir a melhor utilização do ambiente no qual ela se encontra.

Sempre que um ente precisa se comunicar com outro que esteja executando em uma plataforma diferente, o ambiente se encarrega de fornecer os meios para que as HoloVMs se comuniquem. Neste processo o programador não precisa explicitar de nenhuma forma que o acesso é remoto, ou a localização da HoloVM remota.

3.5 Considerações

O presente capítulo apresentou os conceitos propostos pelo Holoparadigma e algumas das aplicações já construídas com a atual etapa de desenvolvimento da HoloVM. O estágio de aperfeiçoamento e desenvolvimento da HoloVM não possibilita a criação de uma aplicação real de grande porte. Esta aplicação necessita de algumas funcionalidades ainda não suportadas pela HoloVM, por exemplo, a consistência de dados, suporte a mobilidade dentre outras. A partir dos estudos realizados no Capítulo 2.5, a necessidade de implementação de uma aplicação real para provar os conceitos propostos pelo Holoparadigma, torna-se indispensável.

Capítulo 4

Modelo Proposto

Procurando explorar o ambiente de programação MHolo e seus componentes, foi desenvolvida uma aplicação móvel distribuída. Como aspectos do trabalho, consideram-se questões ligadas ao campus universitário, envolvendo mobilidade, distribuição e consciência ao contexto. Esta aplicação tem como objetivo desenvolver e permitir o acompanhamento e controle de um evento científico por diferentes classes de atores (*chair* da conferência, *chair* de sessão, apresentadores e participantes). Esta aplicação explora uma infra-estrutura de rede sem fio e dispositivos computacionais móveis, tendo sido desenvolvida para avaliar os recursos de programação Holo. A partir destes aspectos, este capítulo apresenta a descrição da aplicação proposta, com sua metodologia de desenvolvimento e uma aplicação relacionada. Na Seção 4.1 a aplicação é caracterizada e na Seção 4.2 é apresentado um estudo de caso. Por fim, na Seção 4.3 é apresentada uma avaliação sobre o uso do modelo MHolo com o andamento desta dissertação.

4.1 Aplicação Proposta

A acessibilidade a dispositivos móveis, o avanço de aplicações para estes dispositivos, aliados a disponibilidade da infra-estrutura fornecida pela universidade, torna viável a modelagem e desenvolvimento de uma aplicação móvel explorando formas de adaptação ao contexto. Para fins de trabalho, se desenvolveu uma aplicação que busca o auxílio ao controle de eventos científicos. Esta seção descreve a aplicação e a construção do modelo, baseados em um evento científico genérico. O objetivo é apresentar o modelo construído para esta aplicação, verificando as possibilidades oferecida pelo Holoparadigma.

4.1.1 Descrição da Aplicação

A aplicação desenvolvida seleciona um evento cientifico genérico (congresso, conferencias, simpósio e etc) que busca obedecer a um conjunto de requisitos. Estes requisitos são baseados na organização de um determinado evento, que é composto por

TABELA 4.1 – Componetes do modelo

Componentes	Objetivos
Atores	
participante	assistir as sessões, fazer anotações
chair de sessão	controlar e manipular a sessão ao qual foi alocado
chair da conferência	controlar e manipular todo o evento científico
apresentador	manipular algumas funcionalidades da sessão ao qual foi alocado
Sessões	
sessões técnicas	disponibilizar ações para as funcionalidades disponíveis na sessão em questão
palestras	disponibilizar ações para as funcionalidades disponíveis no local alocado para a palestra
pôsteres	disponibilizar espaços para serviços e novidades
Ações	
mobilidade	mobilidade da classe dos atores
funcionalidades	funcionalidades disponíveis pela classe dos atores e sessões

atores, sessões e ações. Enquanto os dois primeiros refletem as entidades do modelo, as ações refletem a forma como estas entidades evoluem. Os atores podem ser divididos em: *chair* da conferência, *chair* de sessão, apresentadores e participantes. O cenário é composto por um evento científico, no qual podem ocorrer sessões técnicas em paralelo. As ações são comportamentos específicos identificados para cada ator ou sessão. Estes componentes do modelo são apresentados na Tabela 4.1. Chama-se a atenção que os objetivos dos componentes não são esgotados na descrição apresentada.

No cenário concebido, a mobilidade é observada através dos atores. Os atores possuem suas ações e objetivos específicos, interagindo com as sessões e suas funcionalidades. O objetivo de um participante é assistir ao maior número de sessões técnicas, fazendo anotações e interagindo com outros participantes. Já o objetivo de um *chair* de sessão é focado na sessão ao qual ele foi alocado, iniciando/terminando apresentações e manipulando a agenda da sessão. O *chair* da conferência possui como objetivo principal a coordenação geral do evento, manipulando a agenda do evento e gerando avisos por parte do evento. O objetivo do apresentador é manipular algumas funcionalidades oferecidas na sessão ao qual foi alocado, trocando informações com o *chair* da sessão e utilizando os equipamentos disponíveis.

Note que neste cenário, atores possuem, além de suas funcionalidade específicas, funcionalidades agregadas de outros atores. Todo apresentador agrega as funcionalidades de um ator participante. Um *chair* de sessão também possui as características do ator participante, obtendo as funcionalidades específicas deste componente. Já o *chair* da conferência recebe as funcionalidades de um ator *chair* de sessão, agregando assim todas as ações de um ator *chair* de sessão e as ações de um ator participante.

Todo ator é identificado ao entrar no contexto do evento. A partir deste momento, o ator é reconhecido como um dos componentes atores. Caso este ator seja identificado como um participante, ele recebe informações sobre o evento, assim escolhendo uma sessão para participar. Ao entrar nesta sessão, algumas funcionalidades e adaptações podem ocorrer. Dentre as novas funcionalidades disponíveis por esta sessão estão: (i) questionar o apresentador; (ii) fazer algumas anotações pessoais; e (iii) se comunicar com outros participantes do evento através de mensagens privadas. Já como forma de adaptação, o participante irá fazer parte de estatísticas promovidas pelo evento e disponibilizadas nas sessões.

Já se o ator for identificado como apresentador, ele possuirá todas as funcionalidades e informações atribuídas a um ator participante, além de suas ações específicas: (i) informações detalhadas sobre sessão que será sua apresentação; (i) horário e local da sessão; (iii) informações sobre o tempo de apresentação; (iv) informação de quem será o coordenador desta sessão; (v) informações sobre disponibilidades de equipamentos na sessão alocada; e (vi) quais funcionalidades a sessão em questão disponibiliza para o apresentador.

Caso o ator seja identificado como *chair* de sessão, o mesmo receberá todas as funcionalidades e informações disponibilizadas a um ator participante, além de suas ações específicas: (i) informações detalhadas sobre a sessão que foi alocado; (ii) horário e local da sessão; (iii) informações detalhadas sobre os apresentadores, tal como a descrição dos mesmos; (iv) controle sobre as estatísticas da sessão; (v) anotações sobre a sessão, manipulando a agenda e outros dados da sessão; (vi) troca de mensagens com o *chair* da conferência; e (vii) atualizar o evento principal com os dados obtidos nesta sessão.

Por último, o ator pode ser identificado como *chair* da conferência. Neste caso o ator possui todas as funcionalidades do ator *chair* de sessão e do ator participante. Tais ações são acrescidas de algumas ações específicas ao *chair* da conferência, como: ao entrar no evento, (i) obtém informações sobre o andamento da conferência; (ii) podendo manipular, enquanto o evento ocorre, tais informações; (iii) chamando ações específicas da conferência; e ao entrar em uma sessão, (iv) recebe e manipula os dados desta sessão; (v) executando ações específicas da sessão; e (vi) publicando e atualizando a agenda da conferência.

Todas as funcionalidades descritas para cada classe de ator da conferência buscam uma idéia central, tendo como foco o auxílio para o acompanhamento e/ou para o controle de um evento científico, explorando os conceitos de mobilidade propostos pelo Holoparadigma em aplicações móveis. Na Tabela 4.2 é apresentado um resumo das principais ações atribuídas a cada ator, bem como os dados que são manipulados por estas ações. Com isso, este modelo permite o controle e a administração das atividades do evento enquanto o mesmo ocorre, com foco prioritariamente nas atividades de gerência.

TABELA 4.2 – Componentes do modelo, ações e dados

Atores	Ações	Dados
participante	busca artigos; faz anotações pessoais; recebe notícias do evento; compõe sua agenda	manipula dados locais de seu próprio repositório; agenda pessoal; artigos baixados; fichario com anotações
chair de sessão	inicia/termina as apresentações; obtém currículos dos apresentadores; envia/recebe mensagens do *chair* da conferência	manipula dados da sessão ao qual foi alocado; agenda da sessão; currículo dos apresentadores; estatísticas da sessão
chair da conferência	inicia/termina a conferência; coordena os chairs de sessão	manipula dados da conferência; agenda e informações da conferência; informações sobre as sessões; horários do evento
apresentador	utiliza os equipamentos disponíveis na sessão; envia o currículo para o *chair* de sessão; recebe/envia mensagens para o *chair* de sessão	acessa os dados específicos da sessão ao qual foi alocado; estatísticas do evento; equipamentos disponíveis

4.1.2 Construção do Modelo

O desenvolvimento, modelagem e especificações são descritos a partir do evento científico genérico. A composição do modelo é baseada nos conceitos do paradigma Holo. Os atores e sessões são modelados como entes. O comportamento é destinado às funcionalidades e ações de um *ente*. Cada ente possui sua própria história, seu comportamento e sua interface. A história é vista como uma forma de controle e armazenamento de dados. A mobilidade é exercida através de um *move* lógico respeitando a hierarquia de entes.

Na Figura 4.1 é apresentada a estrutura construída de forma a representar o evento, o modelo geral proposto. Nela é encontrado o *ente holo*, sendo este o ente principal de todas as aplicações desenvolvidas em Holo. O qual igualmente representa a raiz da hierarquia de entes. A partir deste ente é apresentado o evento científico genérico, *ente evento*. A estrutura do modelo apresentada na Figura 4.1 ilustra um momento do transcurso do evento onde 2 sessões estão ocorrendo em paralelo: *ente sessão 1* e *ente sessão 2*. Como cada elemento deste evento é modelado como um *ente*, estes entes possuem todas suas características básicas, sua *história*, sua *interface* e seu *comportamento*. Na seqüência cada um destes entes é destacado.

A Figura 4.2 projeta um ente participante identificado no modelo principal, ente *participante 1*. Tal projeção mostra as características básicas deste ente, sendo sua história composta por algumas estruturas de dados, contendo informações sobre o ente, informações sobre o evento, bem como dados sobre o próprio participante. A Tabela 4.3

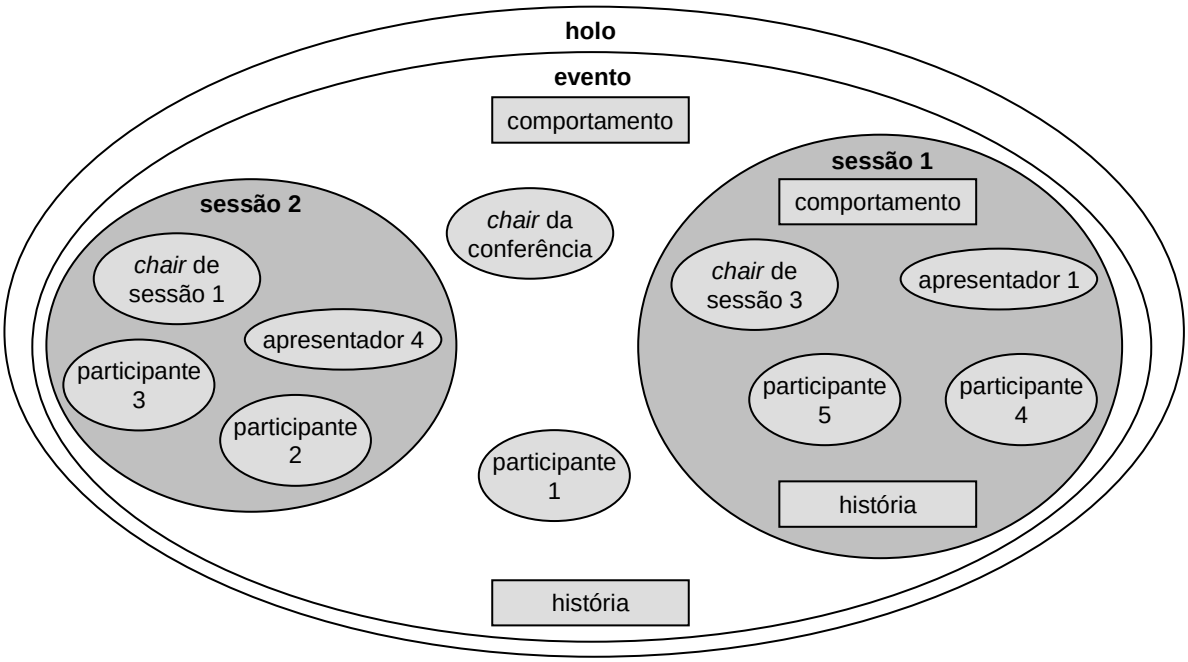


FIGURA 4.1 – Modelo principal

TABELA 4.3 – Representação da história do participante

Origem	Tipo	Dados
Local	Identificação	IdUsuário, Nome, Universidade, ÁreasInteresse
Local	Anotação	IdUsuário, IdAnotação, Anotação
Local	Mensagens	IdMensagem, IdUsuárioOrigem, IdUsuárioDestino, Data, Hora, Mensagem
Evento	Agenda Evento	IdEvento, IdSessão, NomeSessão, Data, HoraInício, HoraFim, Descricao
Evento	Mensagens	IdMensagem, IdEventoOrgivem, Data, Hora, Mensagem

resume as informações manipuladas por este ente.

Na tabela, a manipulação dos dados é dada quanto a origem do mesmo, sendo classificada como: *Local*, os dados são manipulados pelo próprio ente; e o *Evento*, os dados são manipulados pelo ente *chair* da conferência. Já o tipo dos dados pode ser dividido em: *Identificação*, dados com a funcionalidade de identificar um usuário, sessão e/ou evento; *Mensagens*, é o armazenamento de dados utilizados na troca de mensagens entre os usuários do evento e/ou mensagens geradas pelo próprio evento; e a *Agenda Evento*, são dados de descrição do próprio evento e suas sessões e/ou palestras. Além disso, a representação dos próprios dados armazenados.

O comportamento do ente reflete as suas ações e funcionalidades, podendo, este comportamento, acessar a história no próprio ente em operação de leitura e escrita. Os métodos de acesso ao mundo externo são selecionados através da sua interface. A interface

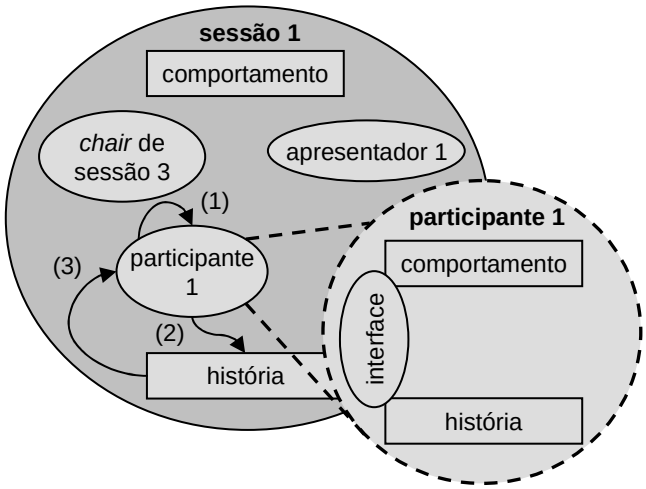


FIGURA 4.2 – Modelo do participante

possibilita o acesso de outros entes a algumas ações disposta na interface deste ente. Podendo ser utilizada como uma forma de adaptação e/ou comunicação imposta por entes externos.

Dentre as funcionalidades encontradas no comportamento de um ente participante, destacam-se:

- *Anotações pessoais*, (1) na Figura 4.2: o participante do evento tem a possibilidade de fazer anotações em sua própria história.

- *Questionar o apresentador*, (2) na Figura 4.2: a possibilidade de interagir com a sessão que esta ocorrendo no momento, fazendo perguntas ao apresentador. Tais perguntas são inseridas na história da sessão, podendo ser buscadas por qualquer participante da mesma.

Participar das estatísticas do evento, (3) na Figura 4.2: o evento possui algumas estatísticas como: número de participantes de uma sessão e grau de satisfação com a sessão ou apresentação atual.

O modelo do participante e suas ações são visualizados na Figura 4.3. Um ator participante ao se identificar passa a ser um ente participante, representado como *participante 1*. Com sua entrada no evento científico, o participante pode cadastrar/alterar seus dados. As informações e a agenda sobre o evento são obtidas, e com isso, o participante pode escolher uma sessão para assistir, no caso, a *sessão 1*. Ao estar no contexto da sessão 1, as funcionalidades oferecidas pela sessão tornam-se disponíveis ao participante. A possibilidade de questionar o apresentador fazendo perguntas na história da sessão e publicar as informações, estão entre as ações previstas. O participante ainda possui as possibilidades de comunicação com outros participantes do evento através de

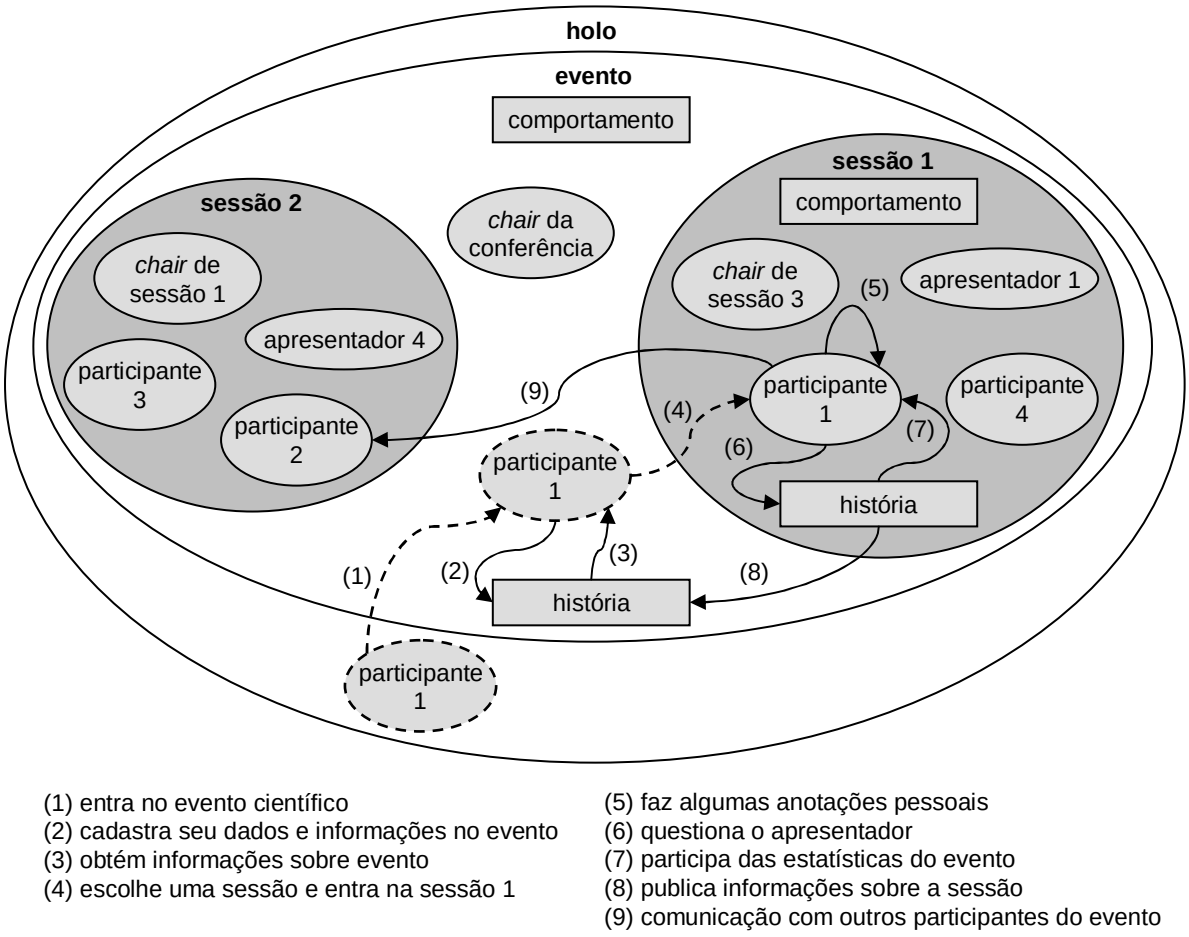


FIGURA 4.3 – Modelo do participante com suas funcionalidades

mensagens pessoais e a possibilidade de anotações na história pessoal possibilitando a troca de informações pelos participantes ao término do evento.

O modelo de um *chair* de sessão é baseado diretamente no modelo do ator participante, agregando funcionalidades específicas de controle de sessão. Este *chair* de sessão possui em sua história, representada na Tabela 4.4, as informações sobre a agenda e os apresentadores alocados a ela. Adicionando informações a descrição da Tabela 4.3, o campo *Informação* é composto por dados pertencentes a identificação da sessão e dos apresentadores.

Com a definição desta história, o *chair* de sessão passa a ter o controle e administração da sessão com a manipulação baseada nestes dados. O ator chair de sessão ao entrar no evento é identificado como um ente, na Figura 4.4, ente *chair de sessão 3*. Algumas informações sobre suas sessões ou palestras são obtidas na história do evento. Dentre estas informações, o chair de sessão recebe sua agenda pessoal contendo qual sessão irá administrar. Com isso, se desloca para a sessão ao qual foi alocado, ente *sessão 1*. No contexto da sessão 1, o chair de sessão 3 recebe os dados sobre os apresentadores, com a

TABELA 4.4 – Representação da história do *chair* de sessão

Origem	Tipo	Dados
Local	Identificação	IdUsuário, Nome, Universidade, AreasInteresse, IdSessão
Pessoal	Anotação	IdUsuário, IdAnotação, Anotação
Sessão	Informação	IdSessão, NomeSessão, IdChairSessão, HoraInício, HoraFim, Local
Sessão	Anotação	IdPergunta, IdUsuário, IdSessão, Pergunta
Sessão	Anotação	IdPesquisa, IdSessão, DadosRequeridos
Sessão	Informação	IdApresentador, IdSessão, HoraInício, DadosApresentador

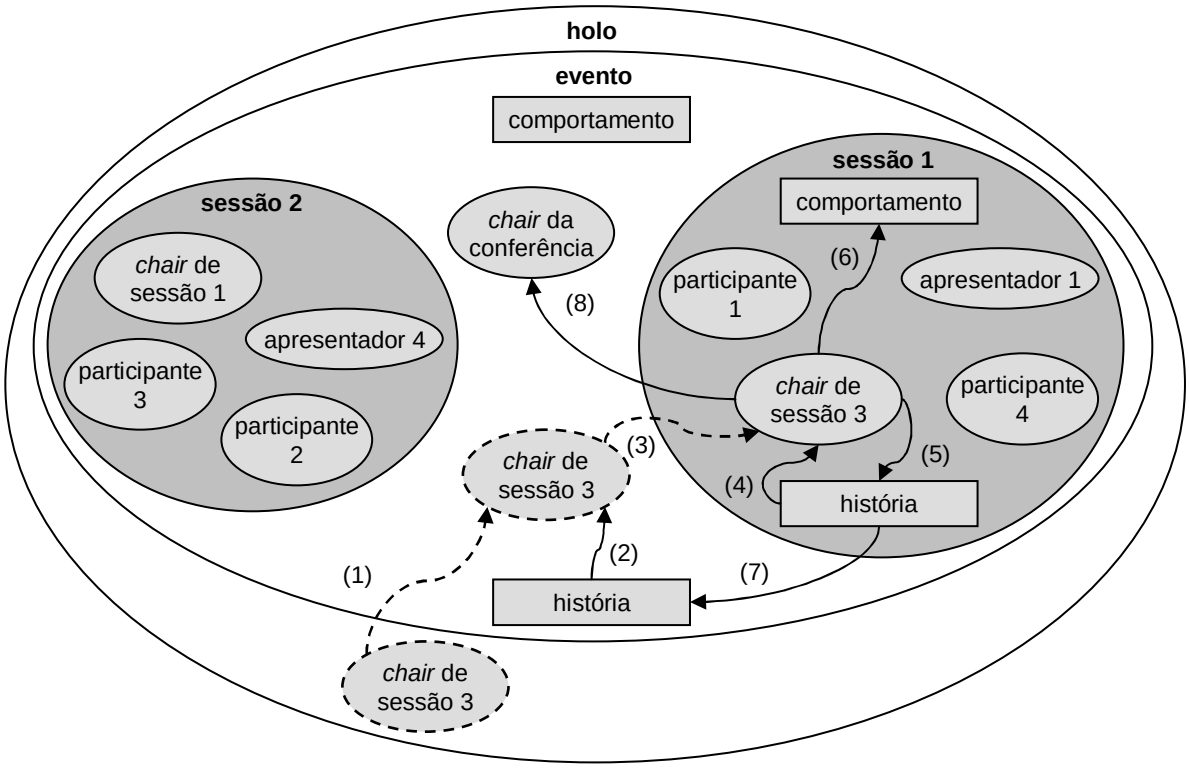
ordem dos mesmos e alguns currículos já cadastrados. A manipulação dos dados da sessão é através de algumas ações da própria sessão, outra ação pertencente as funcionalidades é a de estatística do evento, fazendo a contagem de participantes e publicando estas informações. O *chair* de sessão ainda tem a possibilidade de comunicação direta com o *chair* da conferência, para troca de informações e andamento do evento.

O *chair* da conferência é responsável pelo controle e manipulação da agenda do evento, exercendo ações específicas como: inserir apresentações, alocar salas e sessões, atualizar a agenda do evento, bem como enviar mensagens a todos da conferência. Um ator *chair* da conferência ao se identificar, entrar no evento, Figura 4.5, passando a ser reconhecido com um ente *chair da conferência*. No contexto do evento, o *chair* da conferência obtém informações sobre o andamento do evento, podendo manipular a agenda e os dados armazenados na história do evento. No contexto de uma sessão, o *chair* da conferência passa ter as mesmas funcionalidades de um *chair* de sessão, recebendo e/ou alterando dados sobre a sessão.

Com a instância de cada componente do modelo, as principais funcionalidades foram descritas. O modelo geral (Figura 4.1) é a base para todos os outros modelos. Nota-se que alguns atores são compostos por características de outros atores, como no caso de um ator chair da conferência. Suas características são constituídas por dois tipos de atores, o ator participante e o ator chair de sessão, bem como agregada de algumas características específicas. Destaca-se então que o modelo representado de alguns atores possuem só representado suas características específicas, não representados as funcionalidades agregadas de outros atores.

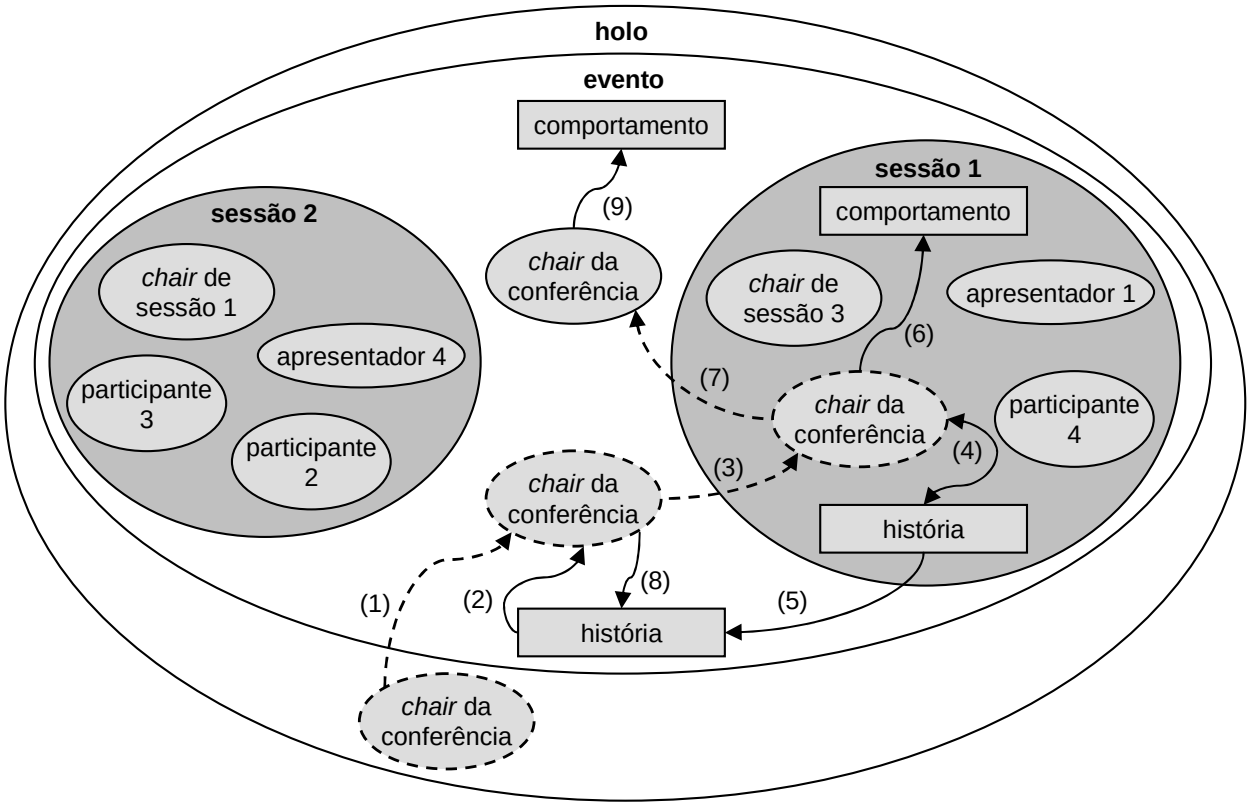
4.2 Estudo de caso

Em comparação, Dey [17] descreve um assistente de conferência. Esta aplicação possui três participantes; o cenário é composto por uma conferência técnica, caracterizada por um número elevado de apresentações ocorrendo em paralelo. O objetivo dos



(1) entra no evento científico
(2) obtém informações sobre suas sessões ou palestras
(3) entra na sua sessão, sessão 1
(4) recebe dados sobre os apresentadores
(5) faz algumas anotações na sessão
(6) invoca ações da própria sessão
(7) publica informações sobre a sessão
(8) comunicação direta com o *chair* da conferência

FIGURA 4.4 – Modelo do *chair* de sessão



(1) entra no evento científico
(2) obtém informações sobre o andamento da conferência
(3) entra em uma sessão
(4) recebe ou altera dados sobre a sessão
(5) publica informações sobre a sessão
(6) chama ações específicas da sessão
(7) sai de uma sessão
(8) manipula informações da conferência
(9) chama ações da conferência
(10) comunicação direta com o *chair* da conferência

FIGURA 4.5 – Modelo do *chair* da conferência

participantes é assistir a apresentações diferentes e compartilhar o que eles aprenderam ao término do dia. Cada participante recebe em seu primeiro dia um dispositivo móvel, neste existe um software de assistência à participação na conferência.

O software recebe informações sobre o interesse do usuário e, baseado nessas informações, recomenda uma apresentação que está para começar; informando ainda a direção para seguir até a apresentação. Ao entrar na sala da apresentação, o software disponibiliza informações sobre a sessão, informações sobre o apresentador, *URL*s disponibilizadas sobre o assunto e outros dados referentes a apresentação. Dentre outras funções oferecidas estão: possibilidade de anotações, marcas em apresentações, informações sobre outras sessões, aliado os avisos sobre mudanças na agenda da conferência, mensagens e localização entre outros participantes da conferência, bem como possibilidade de inserir informações pessoais.

4.3 Considerações sobre o Modelo Holo

Com a caracterização da aplicação proposta, bem como o desenvolvimento do seu modelo com suas funcionalidades, algumas considerações sobre a forma de modelagem com Holo podem ser feitas. O modelo desenvolvido pode ser considerado como um modelo de alto nível, comparado com modelos de outros paradigmas de programação.

A forma de adaptação é baseada em contextos. No modelo proposto, os principais contextos são as sessões e o evento. Os atores ao entrar no evento ou sessão, passam a fazer parte deste contexto. As funcionalidades deste contexto são disponibilizadas aos atores, assim fazendo uso das ações específicas de cada contexto, exercendo a adaptação.

O grande diferencial de ter um processo de modelagem com este é que o programador não necessita de um conhecimento do baixo nível de programação. O meio de comunicação dos dados, a forma como os dados serão acessados, segurança dos dados e as formas de mobilidade física dos equipamentos são de inteira responsabilidade da máquina virtual Holo. O modelo representa apenas a mobilidade lógica e pode ser visto como uma forma natural de mobilidade entre os elementos do modelo. Buscando uma modelagem compatível com o mundo real, as abstrações do modelo são através de ações e funcionalidades, sendo armazenadas como forma de comportamento do elemento modelado e, os dados através da história dos componentes deste modelo. Sendo o modelo e suas abstrações de alto nível, a apresentação do mesmo torna-se bastante descritiva.

Capítulo 5

Desenvolvimento

O desenvolvimento da aplicação proposta no Capítulo 4 foi realizado respeitando a especificação e o mapeamento de um evento selecionado, Seção 5.1. A composição do modelo da aplicação foi baseada nos conceitos apresentados pelo paradigma Holo, cujo suporte de execução é realizado através da *Holo Virtual Machine*. Esta máquina virtual encontra-se em desenvolvimento para as plataformas Linux e Windows, bem como para PDAs. Como o presente trabalho esta sendo desenvolvido de forma simultânea ao projeto e a implementação da HoloVM[24, 25], o primeiro passo tomado foi de utilizar uma ferramenta de programação estável, com vistas a validar os recursos de programação propostos em Holo. Para este desenvolvimento foi utilizado o *.NET Compact Framework*, Seção 5.2, que disponibiliza acesso as funcionalidades de dispositivos portáteis e, oferece maior liberdade para o desenvolvimento de aplicações. Contudo, este ambiente não possui rotinas específicas de suporte às aplicações móveis, acarretando em um custo de desenvolvimento mais elevado face a implementação equivalente em Holo. O atual desenvolvimento do Projeto MHolo, disponibiliza uma versão da HoloVM para PDAs, assim surge a possibilidade de constuir a mesma aplicação utilizando sua própria plataforma. Dessa forma, o segundo passo foi o desenvolvimento da aplicação acompanhamento de um evento científico utilizando a HoloVM em um dispositivo móvel 5.3, respeitando as limitações do porte atual da máquina virtual.

5.1 Mapeamento do Modelo

Para o desenvolvimento de aplicações móveis baseadas em contexto, o projetista da aplicação necessita saber as formas de adaptação, assim tendo todo o controle da adaptação. Para suprir este requisito e colocar em prática o desenvolvimento do modelo proposto, foi selecionado o XXV Congresso da Sociedade Brasileira de Computação (CSBC 2005) [26], realizado na Universidade do Vale do Rio dos Sinos em de julho de 2005. O evento científico selecionado, dentro deste congresso, foi o XXXII Seminário

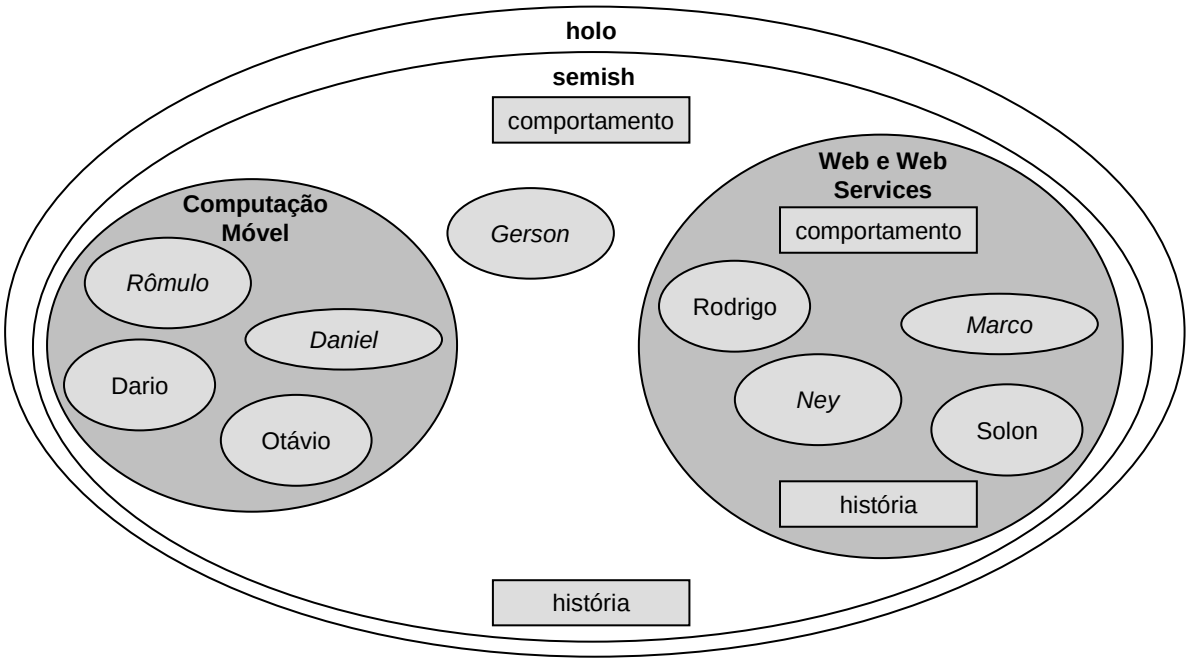


FIGURA 5.1 – Modelo mapeado

Integrado de Software e Hardware (SEMISH 2005) [27]. Este evento científico possui todas as características básicas necessária pelo modelo: a sua organização é baseada em atores, sessões e ações; a mobilidade é exercida pelos atores; existem sessões ocorrendo em paralelo; e as ações são funcionalidades dos atores e das sessões.

A Figura 5.1 ilustra a instanciação do modelo de aplicação proposto (Figura 4.1 na página 46). Nesta figura, o evendo científico é representado no modelo como o *ente semish*, seus atores e sessões também são apresentados como *entes* neste modelo. O Gerson é modelado como um ator *chair* da conferência, Romulo e Ney são caracterizados como atores *chairs* de sessão, Daniel e Marco são atores apresentadores das sessões técnicas Computação Móvel e Web e Web Services, respectivamente e os demais são atores participantes do evento.

Com o mapeamento do modelo, a próxima etapa do trabalho é a implementação. A implementação desenvolvida limita as funcionalidades do modelo às necessárias ao controle e administração do evento. Este controle é exercido pelos atores *chair* de sessão e *chair* da conferência, com funcionalidades de controle da sessão e controle geral da conferência, respectivamente.

5.2 *.NET Compact Framework*

A partir da seleção e análise do evento, bem como do mapeamento do modelo, o próximo passo do trabalho é o desenvolvimento da aplicação. Através destes requisitos, e

FIGURA 5.2 – Instância gráfica da aplicação

aliado a análise dos ambientes estudados no Capítulo 2.5, o uso da plataforma Microsoft *.NET Compact Framework* se tornou uma opção para o desenvolvimento da aplicação. A liberdade oferecida por esta plataforma possibilita desenvolver uma aplicação utilizando os conceitos propostos pelo Holoparadigma para aplicações móveis. Utilizando-se da compatibilidade do .NET CF com dispositivos móveis, a definição da interface e algumas funcionalidades podem ser observadas no decorrer desta sub-seção.

A interface da aplicação é ilustrada na Figura 5.2. O principal componente desta interface é o menu de opções oferecido ao usuário do *software*. A busca de informações sobre o evento e as funcionalidades existentes a este usuário estão disponíveis através deste menu. Pode-se observar ainda um campo para a identificação do usuário e outro para a autenticação do mesmo. A partir desta autenticação ao evento, o usuário do dispositivo móvel é identificado como um ator e dependendo do tipo de ator as ações no menu de opções serão disponibilizadas. O *software* prevê também a possibilidade de troca de mensagens entre usuários da aplicação, informações sobre a grade de apresentações do evento, prevendo alterações por parte do *chair* da conferência durante a realização do evento.

O menu é composto por cinco opções: Principal, SEMISH, Sessões, Avisos e Controle. Para um ator participante e um ator apresentador a opção Controle seria omitida. Após o usuário ter se autenticado no menu Principal, a opção SEMISH passa a ser a opção ativa, ilustrada na Figura 5.3. Nesta opção são apresentadas as informações sobre o evento: os chairs alocados para este evento, os membros de programa, as palestras que estão previstas, bem como as sessões técnicas.

A opção Sessão, Figura 5.4, apresenta todas as sessões técnicas previstas. Nesta opção o usuário da aplicação pode escolher qual sessão ele pretende saber mais

FIGURA 5.3 – Descrição da opção SEMISH

informações, assim abrindo uma descrição liberando acesso aos artigos desta sessão.

O controle tem como foco na manipulação do evento enquanto ele ocorre. O *chair* de sessão possui neste controle informações sobre a sessão a qual foi alocado, podendo visualizar os dados de sua sessão. A Figura 5.5 representa uma seqüência das funcionalidades do controle do *chair* de sessão. Neste caso, o usuário da aplicação é o Prof. Dr. Ney Lemke e seu controle possui as informações sobre a sessão *Web e Web Service*. Neste controle é disponibilizado a hora, local e data da sessão, bem como os artigos que serão apresentados. Cada artigo possui sua descrição detalhada, apresentando o nome dos autores e o currículo do autor principal. Outra funcionalidade oferecida para um *chair* de sessão é a possibilidade de informar ao participante quanto tempo falta para acabar sua apresentação, com 10 minutos e com 5 minutos.

5.3 *Holo Virtual Machine*

Com o decorrer do desenvolvimento do projeto, um protótipo da HoloVM para o dispositivo móvel foi disponibilizado. Com isso, esta parte do desenvolvimento da aplicação é baseada diretamente no ambiente de programação móvel MHolo. Com o porte da sua máquina virtual Holo em fase de testes, esta etapa do trabalho consiste em selecionar uma parte do modelo proposto e construir a aplicação, respeitando as limitações impostas pela atual implementação da HoloVM.

O principal operador de mobilidade é a instrução *move*. Através desta instrução, o suporte de execução exerce uma mobilidade lógica. Para isso, o programador necessita saber os diversos contextos da aplicação, tendo consciência da distribuição lógica na

FIGURA 5.4 – Sessões técnicas

FIGURA 5.5 – Controle do evento

construção da aplicação, mas não necessariamente tendo que manipular a mobilidade física da mesma. O comportamento de um *ente* após a mobilidade não muda, mas o resultado de suas ações depende do novo contexto em que este *ente* está inserido.

Baseado no operador *move*, as principais adaptações e funcionalidades podem ocorrer. O controle deste operador é através da interação do usuário. Atualmente, a máquina virtual Holo não possui um ambiente gráfico, disponibilizando assim interações através de comandos de texto, sendo caracterizado como um menu de opções, já mostrado na figura 3.8 na página 40.

Tais interações por parte do usuário variam de acordo com o contexto do mesmo. Um usuário ao se identificar na conferência, tem suas ações disponíveis através destes menus. Com o objetivo de controle e administração do SEMISH os *entes* selecionados para a implementação são: *chair* de sessão e *chair* da conferência. Suas funcionalidades, perante o modelo, estão ilustradas nas Figuras 4.4 e 4.5, nas páginas 50 e 51, respectivamente.

O usuário do dispositivo móvel ao executar a aplicação necessita se identificar ao sistema, através do seu número de matrícula no evento e sua senha. Caso seja reconhecido como *chair* de sessão, seu principal objetivo é o controle e manipulação da sessão. Dessa forma, ao entrar no evento, o *chair* de sessão recebe as informações da agenda de sua sessão, juntamente com os currículos dos apresentadores. Suas ações podem ser ilustradas através do seu menu de opções, apresentado na Figura 5.6. Tal figura ilustra a aplicação em execução, mostrando algumas ações desenvolvidas e o processo de identificação do usuário à conferência. Essas ações ocorrem a partir da interação direta do usuário com a aplicação. Desta forma, o usuário final pode ser visto como parte integrante da mobilidade lógica de um *ente* móvel.

Caso este usuário seja identificado como um *chair* da conferência, seu principal objetivo é a coordenação geral do evento, manipulando toda a agenda do evento. Dessa forma, ao entrar no evento, o *chair* da conferência recebe as informações da agenda do evento, podendo manipular tais informações e outras funcionalidades através de seu menu de opções, ilustrado na Figura 5.7.

5.4 Avaliação da Linguagem

O desenvolvimento de uma aplicação real, bem como de aplicações sintéticas, tais com aquelas descritas na seção 3.4, permite obter uma avaliação da Hololinguagem. Sendo esta uma linguagem de programação recente e seu suporte de execução encontra-se em construção. Alguns recursos básicos, para desenvolvimento de aplicações, não se encontram disponíveis em todas as plataformas sobre as quais vem sendo implantada. Por exemplo, o programador não dispõe, de manipulação de arquivos, ambiente gráfico e segurança quanto à falha, dentre outros aspéctos. A extensão para suprir estas necessidades na linguagem é uma etapa de trabalhos futuros do projeto. Nesta etapa,

```
c:/ HoloVM.exe evento
HoloVM 0.2 ($Revision: 1.44 $)
- acompanhamento de um evento científico -
Evento SEMISH aberto
Identificação:
 - inscrição := 122
 - senha := ney
 Voce foi reconhecido como Ney - chair de sessão
 faz parte da Instituicao UNISINOS
Controle:
 Sessão Técnica 2 : " Web e Web Service "
  - data : Terça-feira, 26 de julho de 2005
  - hora : 8:30 - 10:30
  - local: Auditório Central
 Artigos:
  - 1: Monitoring Bioinformatics Web services Requests and Responses
  - 2: Arquitetura para busca por Web Services com suporte para QoS
  - 3: Codes: Um Ambiente para Prototipação Musical Cooperativa Baseado na Web
  - 4: Um novo retrato da Web brasileira
Menu:
  1 := Visualizar o Controle
  2 := Alterar status da sessão técnica
  3 := Visualizar informações específicas do artigo
  4 := Alterar dados específicos de um artigo
  5 := Enviar uma mensagem para o chair da conferência
  Ação := 3
Visualizar informações específicas do artigo:
  Informe o número do artigo := 2
   - Nome := Arquitetura para busca por Web Services com suporte para QoS
   - Autores := Rafael Krüger Tavares, Rafael da Rosa Righi, Carlos Becker Westphall
   - CV do ator principal := "Rafael Krüger Tavares está cursando o segundo ano do curso de
mestrando em Ciências da Computação na Universidade Federal de Santa Catarina (UFSC).
Foi bolsista de IC atuando no Laboratório de Redes e Gerência da USFC
participando do projeto TAGERE e HOPE durante 2001-2003.
Atualmente é colaborador do projeto TAGARE."
Menu:
  1 := Visualizar o Controle
  2 := Alterar status da sessão técnica
  3 := Visualizar informações específicas do artigo
  4 := Alterar dados específicos de um artigo
  5 := Enviar uma mensagem para o chair da conferência
  Ação := __
```

FIGURA 5.6 – Amostra da aplicação em execução por parte do *chair* de sessão

```
c:/ HoloVM.exe evento
HoloVM 0.2 ($Revision: 1.44 $)
- acompanhamento de um evento científico -
Evento SEMISH aberto
Identificação:
 - inscrição := 124
 - senha := g
 Voce foi reconhecido como Gerson - chair da conferência
 faz parte da Instituicao UNISINOS
Controle:
 Sessões Abertas:
  - Sessão Técnica 1 - " Computação Móvel "
  - Sessão Técnica 2 - " Web e Web Service "
 Chairs de Sessões:
  - Ney : Sessão Técnica 2
  - Rômulo : Sessão Técnica 1
Menu:
  1 := Visualizar o Controle
  2 := Alterar status de uma sessão técnica
  3 := Alocar um chair de sessão para uma sessão técnica
  4 := Criar um chair de sessão
  5 := Visualizar dados de uma sessão técnica
  6 := Alterar dados de uma sessão técnica
  Ação := 2
Alterar status de uma sessão técnica:
  Informe o número da sessão técnica := 1
   1 := Aberta
   2 := Fechada
   opção := 2
   - A sessão técnica 2, "Web e Web Service", foi fechada.
Menu:
  1 := Visualizar o Controle
  2 := Alterar status de uma sessão técnica
  3 := Alocar um chair de sessão para uma sessão técnica
  4 := Criar um chair de sessão
  5 := Visualizar dados de uma sessão técnica
  6 := Alterar dados de uma sessão técnica
  Ação := 1
Controle:
 Sessões Abertas:
  - Sessão Técnica 1 - " Computação Móvel "
 Sessões Fechadas:
  - Sessão Técnica 2 - " Web e Web Service "
 Chairs de Sessões:
  - Ney : Sessão Técnica 2
  - Rômulo : Sessão Técnica 1
```

FIGURA 5.7 – Amostra da aplicação em execução por parte do *chair* da conferência

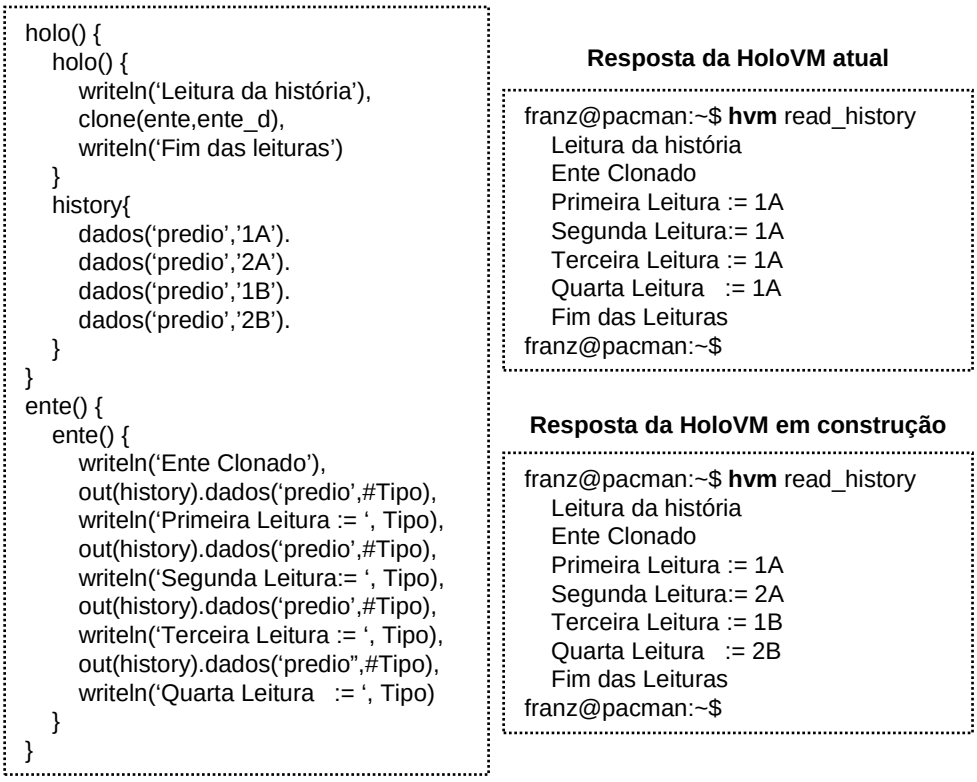


FIGURA 5.8 – Leitura da história de um ente

o conceito de bibliotécas está sendo inserido.

Com o desenvolvimento do suporte para a Hololinguagem ocorrendo em paralelo à construção das aplicações, algumas necessidades voltadas ao ambiente foram encontradas. A integração com as outras áreas do projeto possibilitou a proposta de novas abstrações, para suprir tais necessidades e integrá-las ao desenvolvimento da HoloVM.

Com o decorrer das aplicações, a troca de mensagem entre entes de diferentes níveis, possibilita uma interação direta entre dois entes e novas abstrações para uma aplicação, facilitando a comunicação. Tal necessidade, depois de constatada, passou a fazer parte do suporte da Hololinguagem.

Outra necessidade encontrada foi a forma de acesso aos dados da história. Tal funcionalidade, ainda não agregada a versão atual da HoloVM, pode ser ilustrada na Figura 5.8. Atualmente, uma chamada de leitura à história de um ente possui como resposta sempre a primeira ocorrência encontrada. Esta reestruturação prevê um controle das ocorrências, deslocando a próxima ocorrência desejada.

Levando em consideração a aplicação controle do evento científico, ao utilizar os dispositivos móveis, ocorreu uma elevada perda de conexão. Este fato acarretou no desempenho da aplicação, em decorrência do suporte ao *History Server* para aplicações distribuídas. Com este suporte, os dados da história são armazenados de forma centralizada. Dessa forma, ao ocorrer uma desconexão do dispositivo móvel, a aplicação

perdeu comunicação com os dados do evento científico, perdendo desempenho.

Em relação ao apoio ao desenvolvimento, o ambiente de programação da Hololinguagem apresenta diversas ferramentas já operacionais. O cenário atual é composto por um compilador operacional, suportanto aplicações móveis, mobilidade e consciência ao contexto. Para o desenvolvimento de aplicações, o MHolo é composto por um conjunto de elementos: (i) uma linguagem Holo, (ii) uma linguagem de modelagem que fornece programação visual em Holo, (iii) uma ferramenta case que suporta a especificação de aplicações através da HML e geração automática de código em Holo, (iv) um ambiente de desenvolvimento integrado (IDE) que permite edição, compilação e execução de programas Holo e (v) uma máquina virtual que suporta a execução de programas nativos em Holo.

Particular atenção neste trabalho é dada a exploração das facilidades de mobilidade neste paradigma. Como instrução da mobilidade, o operador *move* é um nível de abstração que permite movimentar entes em diversos contextos. O programador de uma aplicação necessita ter consciência da distribuição lógica na construção da aplicação, mas não necessariamente tendo que manipular a mobilidade física da mesma. Com isso, pode-se considerar abstrações de alto nível, tornando a apresentação do modelo bastante descritiva.

Capítulo 6

Conclusão

A avaliação da aplicação, que foi o objeto de estudo deste trabalho, foi realizada no Congresso da Sociedade Brasileira de Computação [26] com a organização e controle do Seminário Integrado de Software e Hardware (SEMISH) [27]. Este evento ocorreu na Universidade do Vale do Rio dos Sinos em julho de 2005. Este capítulo inicia sumarizando o resultado geral atingido, Seção 6.1. Em seguida, na Seção 6.2 são apresentandos os objetivos atingidos com a construção do modelo, as contribuições do trabalho são listadas na Seção 6.3. Por fim, na Seção 6.4, são apresentadas algumas considerações finais.

6.1 Resultado Geral

O foco desta dissertação é a exploração de ambientes de computação móvel para o desenvolvimento de aplicações. Neste sentido surge a necessidade de manipular, uma nova classe de abstrações para programação de sistemas, envolvendo mobilidade e sensibilidade ao contexto. Estas abstrações foram dominadas pelo estudo de ambientes na Seção 2.5 e pelo desenvolvimento de aplicações sintéticas na Seção 3.4, em particular, o projeto MHolo.

Os estudos realizados sobre o modelo de programação proposto pelo Holoparadigma e a atual implementação da HoloVM, buscam explorar o ambiente MHolo através da transparência da mobilidade, provendo suporte as aplicações móveis. Estes resultados atingidos indicam a viabilidade do modelo desenvolvido, sendo apresentado na Seção 4.1.2. Dessa forma, foi possível mostrar que novas abstrações de recursos de programação podem auxiliar na exploração de infra-estrutura de redes móveis.

6.2 Modelo Proposto

A concretização do modelo e o desenvolvimento da aplicação de controle de um evento científico proposta permitiu uma avaliação do ambiente de programação móvel

MHolo, que abrange mobilidade, distribuição e consciência ao contexto. Esta avaliação considerou tanto o modelo proposto por Holo, quanto de seus recursos de programação. Como aspecto de trabalho, considerou-se questões ligadas ao campus universitário, tendo como objetivo permitir o acompanhamento do evento científico SEMISH 2005, realizado na UNISINOS.

Os principais objetivos atingidos foram: a modelagem em Holo de uma aplicação real; a descrição do comportamento dos atores envolvidos; o mapeamento do evento no modelo em Holo; bem como a exploração da infra-estrutura de rede sem fio e dos dispositivos computacionais móveis. Com isso, ocorreu a possibilidade de avaliar os recursos de programação oferecidos pelo MHolo.

Com a previsão da continuidade do desenvolvimento da Hololinguagem e sua máquina virtual, novas implementações e abstrações ao suporte da linguagem são esperadas. O modelo desenvolvido busca atingir todas as necessidades de um evento científico e explorar o ambiente de programação MHolo. Futuramente, com as novas abstrações adicionadas a HoloVM, existe a possibilidade de implementação completa deste modelo.

6.3 Contribuição do Trabalho

A principal contribuição desta dissertação é dada na área computação móvel, mais especificamente no desenvolvimento de aplicações para este tipo de arquitetura. O objetivo geral é explorar a proposta de abstrações para mobilidade especificadas por Holo na construção de uma aplicação real. Este objetivo geral foi atingindo através dos seguintes resultados:

- *Ambientes de computação móvel*: o estudo realizado sobre os ambientes de computação móvel e suas abstrações para o desenvolvimento de aplicações;
- *Evento científico genérico*: o desenvolvimento do modelo de um evento científico genérico (congresso, conferencias, simpósio e etc) estabelecendo um conjunto de requisitos. Estes requisitos são baseados na organização de um determinado evento, sendo composto por: atores, sessões e ações;
- *Mapeamento do evento*: a partir da definição de um evento científico, SEMISH 2005. O mapeamento deste evento e a oportunidade de um canal de comunicação com a organização do evento científico, possibilitou acrescentar algumas características específicas, agregando informações ao modelo;
- *.NET Compact Framework*: construção de uma aplicação móvel, respeitando as especificações do Holoparadigma. A partir dos estudos realizados, o ambiente de programação móvel, o .NET CF, possibilitou a construção das abstrações previstas

por Holo, e também prove suporte para as funcionalidades dos dispositivos móveis portáteis;

- *SEMISH 2005*: a avaliação da aplicação ocorreu no CSBC 2005, mais especificamente no evento científico SEMISH. O uso da aplicação caracterizada como "*Acompanhamento de um evento científico*" facilitou a coordenação e controle do evento por parte dos organizadores. A funcionalidade mais explorada foi a troca de mensagens, possibilitando a comunicação constante e direta entre os coordenadores do evento;

- *Holo Virtual Machine*: ainda em desenvolvimento, e com a disponibilidade de um protótipo para dispositivos móveis. A HoloVM se tornou o foco para a construção da mesma aplicação, respeitando a sua atual implementação. Com isso, foi explorado o ambiente de programação móvel MHolo no desenvolvimento de uma aplicação móvel com a finalidade de controle e acompanhamento de um evento científico;

- *Contribuição da aplicação*: esta aplicação explorou uma infra-estrutura de rede sem fio e os recursos oferecidos pelo dispositivo computacional móvel, tendo por finalidade avaliar os recursos de programação do MHolo;

- *Contribuição com o Projeto MHolo*: esta dissertação foi desenvolvida como atividade conjunta do projeto MHolo. O reflexo desta atividade foi a construção de uma aplicação real, fazendo uso do ambiente de execução para computação móvel, o MHolo. O trabalho potencializa a área de aplicações neste projeto, reportando ao restante da equipe do MHolo algumas considerações sobre o uso deste software e das abstrações providas pelo Holoparadigma. Dessa forma, agregando valor ao ambiente de execução de MHolo em desenvolvimento.

- *Estado da arte*: contribuição ao estado da arte no desenvolvimento de aplicações móveis, explorando novas estruturas de programação e ambientes voltados a computação móvel;

- *Publicações de artigos*: espera-se submeter um artigo com o resultado final da dissertação para o *XXXIII Seminário Integrado de Software e Hardware* (SEMISH 2006), que ocorrerá em julho de 2006. Além disso, este trabalho[28] foi a apresentado no Fórum de Pós-Graduação da Escola Regional de Alto Desempenho (ERAD 2006), em janeiro de 2006.

6.4 Considerações Finais

O desenvolvimento de aplicações para dispositivos móveis (PDA, laptop, etc) está voltado a um cenário que abrange mobilidade, distribuição e consciência ao contexto em

que se encontram. Esta dissertação buscou explorar estes recursos para desenvolvimento de aplicativos, tendo como principais metas: (i) explorar computação móvel sobre uma estrutura computacional com recursos de programação limitados; (ii) aplicar os conceitos de mobilidade do Holoparadigma em uma aplicação real; (iii) colaborar diretamente com a coordenação do evento científico; (iv) desenvolver novas aplicações no contexto do Projeto MHolo, gerando novas possibilidades de aplicações sobre o desenvolvimento e sua execução e (v) colaborar com novas funcionalidades para a Hololinguagem.

Esta aplicação foi desenvolvida de forma simultânea ao projeto e a implementação da HoloVM. A opção tomada foi utilizar uma ferramenta de programação estável, com vista a validar os recursos de programação em Holo. Dessa forma, na primeira parte do desenvolvimento foi utilizado o *.NET Compact Framework*. Este ambiente disponibiliza acesso às funcionalidades de dispositivos portáteis e, oferece uma maior liberdade para o desenvolvimento de aplicações. Contudo, este ambiente não possui rotinas específicas de suporte a abstrações de alto nível para programação de aplicações móveis. Em um segundo momento, uma nova versão da aplicação foi construída diretamente no suporte de execução provido pelo ambiente MHolo, a HoloVM. Esta segunda versão, embora não refletindo toda a funcionalidade da aplicação construída em *.NET*, possibilitou explorar efetivamente os recursos disponibilizados na atual versão da máquina virtual Holo e o próprio modelo desenvido. Desta forma, o trabalho apresentou algumas características do ambiente MHolo, demonstrando uma forma de representação intuitiva e natural para o desenvolvimento de aplicações.

Os capítulos iniciais desta dissertação apresentam alguns termos utilizados em computação móvel (Capítulo 2) e um estudo quanto aos ambientes de computação móvel (Seção 2.5). Durante este estudo verificou-se que as aplicações utilizadas para demonstrar as funcionalidades destes ambientes propunham soluções a problemas de baixa complexidade. A aplicação de maior impacto estudada foi aquela propondo um serviço de acompanhamento de conferência. A descrição do paradigma adotado é apresentada no Capítulo 3. O desenvolvimento de algumas aplicações e a mobilidade adotada por este paradigma de programação são apresentadas na Seção 3.4. O evento científico é idealizado no Capítulo 4, com a descrição e o modelo proposto. Esta aplicação difere da do Dey [17] por permitir o controle e gerência do evento, e não seu acompanhamento por algum participante. O desenvolvimento é descrito no Capítulo 5, seguido dos resultados obtidos e algumas considerações finais, no Capítulo 6.

Como trabalho futuro, planeja-se a avaliação do modelo em outros eventos científicos. Será também desenvolvido novas aplicações e testes, buscando explorar as novas funcionalidades propostas ao ambiente MHolo.

Bibliografia

[1] Adenauer Corrêa Yamin, Iara Augustin, Jorge Luis Victória Barbosa, Luciano Cavalheiro da Silva, Rodrigo Araújo Real, Gerson Geraldo Homrich Cavalheiro, and Cláudio Fernando Resin Geyer. Towards merging context-aware, mobile and grid computing. *International Journal of High Performance Computing Applications*, 17:191–203, 2003.

[2] Jorge Barbosa and Cláudio Fernando Resin Geyer. Uma linguagem multiparadigma orientada ao desenvolvimento de software distribuído. In *V Simpósio Brasileiro de Linguagens de Programação (SBLP)*, volume 6, Curitiba, maio 2001.

[3] Jorge Barbosa. *Holoparadigma: um modelo multiparadigma orientado ao desenvolvimento de software distribuído.* Tese (doutorado em ciência da computação), Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2002. 213p.

[4] Alex Sandro Garzão and Jorge Barbosa. Uma máquina virtual com suporte à concorrência, mobilidade e blackboards. In *XXIX Conferência Latinoamericana de Informática (CLEI)*, volume 24, La Paz, 2003. Universidad Mayor de San Andrés. p. 65-69.

[5] B. D. Noble and M. Satyanarayanan. Experience with adaptive mobile aplications in odyssey. In *Mobile Networks and Applications*, volume 4, pages 245–254, Springer Netherlands, 1999.

[6] Daniel Salber, Anind K. Dey, and Gregory D. Abowd. The context toolkit: Aiding the development of contextaware applications. In *Human Factors in Computing Systems: CHI 99. Pittsburgh, PA: ACM Press.*, may 1999. pp. 434-441.

[7] Robert Grimm. *System support for pervasive applications.* Phd thesis, University of Washington, Washington, December 2002.

[8] Microsoft Corporation. *The .NET Compact Framework*, April 2002. Disponível em: http://msdn.microsoft.com/vstudio/device/compactfx.asp
Acesso em janeiro de 2006.

[9] Aline Baggio. System support for transparency and network-aware adaptation in mobile environments. In *SAC 98: Proceedings of the 1998 ACM symposium on Applied Computing*, pages 405–408, New York, NY, USA, 1998. ACM Press.

[10] G. Roman, A. Murphy, and G. Picco. A software engineering perspective on mobility. In *A. C. W. Finkelstein, editor, Future of Software Engineering. ACM Press*, 2000.

[11] R. H. Katz. Adaptation and mobility in wireless information systems. In *IEEE Personal Communications*, volume 1, pages 6–17, 1994.

[12] Robert Grimm, Janet Davis, Ben Hendrickson, Eric Lemar, Adam MacBeth, Steven Swanson, Tom Anderson, Brian Bershad, Gaetano Borriello, Steven Gribble, and David Wetherall. Programming for pervasive computing environments. In *In Proceedings of the 18th ACM Symposium on Operating Systems Principle*, volume Chateau Lake Louise, Banff, Canada., October 2001.

[13] Adenauer Corrêa Yamin. *Arquitetura para um ambiente de grade computacional direcionado às aplicações distribuídas, Móveis e Conscientes do Contexto da Computação Pervasiva.* Tese (doutorado em ciência da computação), Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2004. 194p.

[14] J. Inouye, S. Cen, C. Pu, and J. Walpole. System support for mobile multimedia applications. In *7th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 97) (St. Louis, Missouri, May 1997)*, pages 143–154, 1997.

[15] M. Satyanarayanan. Mobile Information Access project home page, 2005. Disponível em:
http://www-2.cs.cmu.edu/afs/cs/project/coda-www/ResearchWebPages/
Acesso em janeiro de 2006.

[16] A. Dey and G. Abowd. Context Toolkit project home page, 2005. Disponível em:
http://www.cc.gatech.edu/fce/contexttoolkit/
Acesso em janeiro de 2006.

[17] Anind K. Dey, Daniel Salber, and Gregory D. Abowd. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. In *Human-Computer Interaction (HCI) Journal*, volume 16, 2001. pp. 97-166.

[18] Robert Grimm. one.world project home page, 2005. Disponível em:
http://cs.nyu.edu/rgrimm/one.world/
Acesso em janeiro de 2006.

[19] Robert Grimm, Janet Davis, Ben Hendrickson, Eric Lemar, Adam MacBeth, Steven Swanson, Tom Anderson, Brian Bershad, Gaetano Borriello, Steven Gribble, and David Wetherall. System support for pervasive applications. In *8th Workshop on Hot Topics in Operating Systems (HotOS-VIII)*, volume Elmau, Germany., pages 147–151, 2001.

[20] Jorge Barbosa. Holoparadigma project home page, 2005. Disponível em:
http://www.inf.unisinos.br/ holo/
Acesso em janeiro de 2006.

[21] Doug Lea. Objects in groups. 1994.

[22] Jorge Barbosa and A. E. Schaeffer. Using mobility and blackboards to support a multiparadigm model oriented to distributed processing. In *XIII Simpósio Brasileiro de Arquitetura de Computadores e Processamento de Alto Desempenho (SBAC-PAD)*, volume 13, Pirenópolis, Brasil, 2001.

[23] Daniel Torres Bonatto. *HNS: Uma solução para suporte à execução distribuída considerando aspectos da pervasividade.* Dissertação (mestrado em computação aplicada), Programa Interdisciplinar de Pós Graduação em Computação Aplicada, Universidade do Vale do Rio dos Sinos, São Leopoldo - RS, 2005.

[24] Daniel Torres Bonatto, Jorge Barbosa, José Dirceu G. Ramos, and Gerson Geraldo H. Cavalheiro. Pholo: Uma arquitetura para a computação pervasiva utilizando o holoparadigma. In *VI Workshop em Sistemas de Alto Desempenho (WSCAD)*, Rio de Janeiro, 2005.

[25] Daniel Torres Bonatto, Jorge Barbosa, and Gerson Geraldo H. Cavalheiro. Pholo: um suporte à computação pervasiva para o holoparadigma. In *Escola Regional de Alto Desempenho (ERAD) - Fórum de Pós-Graduação*, Ijuí - RS, 2006. Comissão Regional de Alto Desempenho (CRAD).

[26] XXV Congresso da Sociedade Brasileira de Computação, 2005. Disponível em:
http://www.sbc.org.br/sbc2005/
Acesso em janeiro de 2006.

[27] XXXII Seminário Integrado de Software e Hardware, 2005. Disponível em:
http://www.unisinos.br/congresso/sbc2005/?sessao=semish
Acesso em janeiro de 2006.

[28] Dario Fernandes Franz, Jorge Barbosa, and Gerson Geraldo H. Cavalheiro. Exploração do ambiente de computação móvel mholo no desenvolvimento de aplicações. In *Escola Regional de Alto Desempenho (ERAD) - Fórum de Pós-Graduação*, Ijuí - RS, 2006. Comissão Regional de Alto Desempenho (CRAD).