PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MODELO GERENCIADOR DE DESCOBERTA DE SERVIÇOS PERVASIVOS CIENTE DE CONTEXTO

VALDERI REIS QUIETINHO LEITHARDT

Dissertação apresentada como requisito para obtenção do grau de Mestre em Ciência da Computação, pelo Programa de Pós-Graduação da Faculdade de Informática da Pontifícia Universidade Católica do Rio Grande do Sul, PPGCC.

Orientador: Fabiano Passuelo Hessel

Porto Alegre
2008

Dados Internacionais de Catalogação na Publicação (CIP)

L533m Leithardt, Valderi Reis Quietinho
Modelo gerenciador de descoberta de serviços pervasivos ciente de contexto / Valderi Reis Quietinho Leithardt. – Porto Alegre, 2008.
84 f.

Diss. (Mestrado) – Fac. de Informática, PUCRS.
Orientador: Prof. Dr.Fabiano Passuelo Hessel.

1. Informática. 2. Arquitetura de Computador.
3. Computação Pervasiva. 4. Sistemas Distribuídos.
5. Protocolos. I. Hessel, Fabiano Passuelo. II. Título.

CDD 004.21

Ficha Catalográfica elaborada pelo
Setor de Tratamento da Informação da BC-PUCRS

Pontifícia Universidade Católica do Rio Grande do Sul
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

TERMO DE APRESENTAÇÃO DE DISSERTAÇÃO DE MESTRADO

Dissertação intitulada **"Modelo Gerenciador de Descoberta de Serviços Pervasivos Ciente de Contexto"**, apresentada por Valderi Reis Quietinho Leithardt, como parte dos requisitos para obtenção do grau de Mestre em Ciência da Computação, Sistemas Embarcados e Sistemas Digitais, aprovada em 30/12/08 pela Comissão Examinadora:

Prof. Dr. Fabiano Passuelo Hessel – Orientador — PPGCC/PUCRS

Prof. Dr. Eduardo Augusto Bezerra — PPGCC/PUCRS

Prof. Dr. Jorge Luis Victória Barbosa — UNISINOS

Homologada em 01/12/2009, conforme Ata No. 21/09 pela Comissão Coordenadora.

Prof. Dr. Fernando Gehm Moraes
Coordenador.

PUCRS
Campus Central
Av. Ipiranga, 6681 – P32 – sala 507 – CEP: 90619-900
Fone: (51) 3320-3611 – Fax (51) 3320-3621
E-mail: ppgcc@pucrs.br
www.pucrs.br/facin/pos

Primeiramente a DEUS, também agradeço a minha esposa e principalmente as minhas filhas pelo apoio, compreensão de minhas ausências no decorrer deste trabalho.

AGRADECIMENTOS

Esta seção é dedicada a homenagear a todos que de uma forma ou de outra contribuíram para a elaboração deste trabalho, foram vários colegas que estiveram ao meu lado dando apoio, auxiliando e fazendo com que o caminho a ser percorrido ficasse cada vez mais próximo.

A Deus por me proporcionar saúde e perseverança, a tornar mais um grande sonho realidade. A minha esposa, Edilaine, pelo apoio, dedicação e carinho nos momentos difíceis. A compreensão de minhas ausências com minhas duas filhas, Jheine e Nathaly nesse período de dedicação a esse trabalho.

Ao meu orientador, Professor Dr. Fabiano Passuelo Hessel, que me auxiliou no desenvolvimento deste trabalho, apoiou em vários momentos de decisões não somente do trabalho, mas em várias outras questões “baita parceiro”. Pela paciência na correção e esclarecimento de dúvidas, Aos outros professores do PPGCC, que sempre me deram apoio e incentivo, aos meus colegas de trabalho do GSE e GAPH, especialmente a Luís Henrique Leal Ries, de fundamental importância no decorrer deste trabalho.

Enfim, agradeço também a mim mesmo pela dedicação, pelas várias noites de sono inspiradoras, pela busca constante do aprimoramento, e antecipo-me em desculpas aos que também contribuíram e não foram citados diretamente, visto que foram vários os colegas que me auxiliaram.

MODELO GERENCIADOR DE SERVIÇOS
PARA PLATAFORMAS PERVASIVAS SENSÍVEIS AO CONTEXTO

RESUMO

A computação pervasiva possibilita uma série de aplicações na qual o usuário não precisa se preocupar com o sistema computacional e passa a se focar na sua tarefa fim, ou até mesmo deixar de se preocupar com tarefas secundárias e se concentrar em uma tarefa principal. Esta mudança é mais um passo em direção ao conceito de Computação Ubíqua (Ubiquitous/Pervasive Computing) introduzido por Mark Weiser [WEI91]. Weiser propõe, assim, uma integração contínua entre ambiente e tecnologia na tarefa de auxiliar os usuários nas suas mais variadas atividades cotidianas. Entretanto, existe uma série de requisitos a serem analisados e verificados na elaboração de projetos de computação pervasiva. O desenvolvimento da computação pervasiva tornará os dispositivos eletrônicos repletos de funcionalidades computacionais que serão capazes de reconhecer e responder às necessidades contextuais e individuais de cada pessoa. Por exemplo, em uma sala de aula, os alunos se adaptariam ao contexto do professor, (disciplina e do conteúdo), assim que o mesmo entrar na sala. Para essa situação tornar-se realidade, uma característica importante desse tipo de aplicação é a capacidade de reconhecer o contexto do usuário de forma mais transparente possível, tornando os dispositivos eletrônicos do ambientes ainda menos perceptíveis para o usuário. Este ambiente computacional pervasivo consiste de uma grande variedade de nodos de diversos tipos: móveis e fixos, aplicações e serviços interconectados. Neste caso, computação pervasiva também é sinônimo de conectividade pervasiva, e reflete uma computação altamente dinâmica e distribuída. Hoje, a concentração das pesquisas está em disponibilizar tecnologias chave, para suportar este novo ambiente computacional. Dentro dessas necessidades, este trabalho apresenta um modelo gerenciador de descoberta de serviços pervasivos (MgDsP) para utilização em aplicações móveis, sensíveis ao contexto.

Palavras-chave: Computação pervasiva, ciência de contexto, descoberta de serviços, protocolos.

MODEL MANAGER SERVICES FOR PLATFORMS PERVASIVE SENSITIVE TO CONTEXT

ABSTRACT

The pervasive computing allows a range of applications in which the user does not need to worry about the computer system and begins to focus on their task order, or even stop worrying about secondary tasks and focus on a primary task. This change is another step toward the concept of Ubiquitous Computing (ubiquitous / Pervasive Computing) introduced by Mark Weiser [WEI91]. Weiser therefore proposes a continuous integration between environment and technology in the task of assisting users in their most varied daily activities. However, there are a number of requirements to be analyzed and verified in the development of projects in pervasive computing. The development of pervasive computing devices become filled with electronic computing features that will be able to recognize and respond to contextual and individual needs of each person. For example, in a classroom, students have adapted to the context of the teacher, (discipline and content), so that it enters the room. For this to become reality, an important feature of this type of application is the ability to recognize the context of the user a more transparent way possible, making the electronic devices of the environment even less visible to the user. This pervasive computing environment consists of a variety of nodes of various types: fixed and mobile, interconnected applications and services. In this case, pervasive computing is also synonymous with pervasive connectivity, and reflects a highly dynamic and distributed computing. Today, the focus of research is to provide key technologies, to support this new computing environment. Within these needs, this paper presents a model manager discovered service pervasive (MgDsP) for use in mobile applications, sensitive to the context.

Keywords: Pervasive Computing, context of science, discovery services, protocols.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

AMA	Agente Móvel da Aplicação
API	Application Programming Interface
CORBA	Common Object Request Broker Architecture
CTI	Centro de Tratamento Intensivo
DDR-RAM	Double Data Rating - Random Access Memory
DHCP	Dynamic Host Configuration Protocol
DSP	Digital Signal Processing
DVD	Digital Vídeo Disc
E/S	Entrada/Saída
GIOP	General Inter ORB Protocol
GPIO	General Purpose Input/Output
GTSH	Gator Tech Smart House
HD	Hard Disk
HNS	Holo Name System
HW	Hardware
HW/SW	Hardware/Software
I/O	Input/output
IP	Internet Protocol
NFS	Network File System
OMAP	Open Multimedia Application Platform
ORB	Object Request Broker
PC	Personal Computer
PDA	Personal Digital Assistant
QoS	Quality of Software
RBA	Room-Bridge Adapter
RFID	Radio Frequency Identification
ROM	Read Only Memory
SDP	Service Discovery Protocol
SO	Sistema Operacional
SOC	System on a Chip
SW	Software
TCP/IP	Transmission Control Protoco l /Internet Protocol

TFTP	Trivial File Transfer Protocol
TV	Televisão
UART	Universal asynchronous receiver/transmitter
UPnP	Universal Plug and Play
USB	Universal Serial Bus
XML	Extensible Markup Language
YUM	Yellowdog Updater Modified

SUMÁRIO

1 INTRODUÇÃO

Na visão de Mark Weiser [WEY91], “Computação Ubíqua, os recursos de computação seriam onipresentes na vida diária e seriam conectados com a finalidade de fornecer a informação ou serviços que os usuários requerem em qualquer lugar e em qualquer tempo”. Nesta visão, se apresenta um ambiente impregnado com dispositivos de computação e comunicação interagindo com o homem de forma tranqüila (Calm Technology), na qual o usuário não percebe que esta interagindo com a máquina, o ambiente interage com ele de forma autônoma, interativa e relevante [WEY91].

A última década trouxe grandes avanços nas tecnologias de computação e de comunicação móvel. Existe hoje uma tendência de mudança no paradigma de computação, tradicionalmente estático, relativamente previsível e baseado em estações de trabalho, para um novo paradigma, altamente dinâmico, com constantes mudanças de ambientes e dispositivos causadas pela mobilidade do usuário. Caracterizado pelo uso de dispositivos móveis multifuncionais, como celulares, *smartphones* e *PDA’s* (*Personal Digital Assistants*) [AND04].

Esta mudança é mais um passo em direção ao conceito de Computação Ubíqua (Ubiquitous / Pervasive Computing) introduzido por Mark Weiser. Weiser, que vislumbrou novos sistemas e ambientes acrescidos de recursos computacionais capazes de prover serviços e informações quando e onde sejam desejados pelos usuários (“*everywhere, everytime computing*”). Weiser propõe, assim, uma integração contínua entre ambiente e tecnologia na tarefa de auxiliar os usuários nas suas mais variadas atividades cotidianas. Entretanto existe uma série de requisitos. A literatura não apresenta uma única definição para o termo computação ubíqua. Diferentes autores [SAT01], [YAM04], apresentam diferentes definições para este termo. Neste trabalho o termo computação ubíqua será utilizado com a junção da computação pervasiva e da computação móvel. A justificativa de se realizar uma diferenciação desses termos é que um dispositivo que está inserido em um ambiente, não necessariamente é móvel [LIM07]. Devido a isso, quando for utilizado o termo *computação ubíqua*, considera-se o alto grau de dispositivos inseridos da computação pervasiva juntamente com o alto grau de mobilidade da computação móvel, como pode ser visualizado na figura 1.

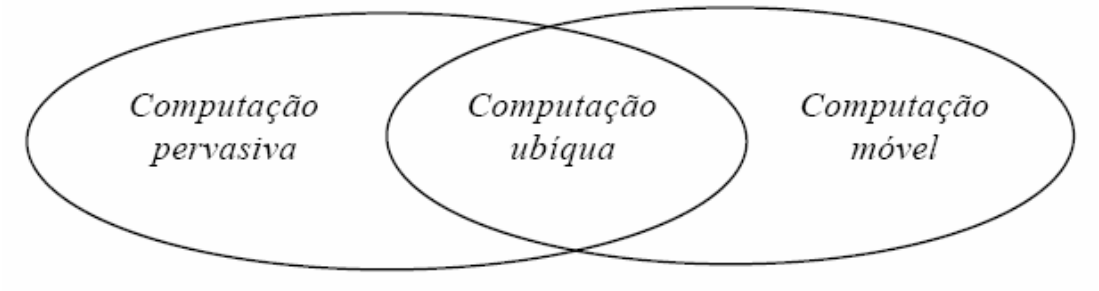

Figura 01: Relação entre computação pervasiva, ubíqua e móvel
(Figura extraída de [LYY02]).

A Computação Pervasiva que é fundamentada em "uma pessoa para muitos computadores" é possível de ser concretizada dado:

- O avanço da tecnologia das redes de comunicação, computação móvel, dispositivos embarcados, sensores, etiquetas RFID, etc.
- O incremento de eletro-eletrônicos digitais de uso diário;
- O avanço nas plataformas de computação pervasiva / ubíqua como *smart spaces*, middlewares, agentes, ontologias, etc.

O objetivo da computação pervasiva é criar Ambientes Inteligentes (salas de aula, residências, escritórios, hospitais, automóveis, etc.). Geralmente utilizam dispositivos em rede inseridos no ambiente que fornece conexão discreta e serviços diversos todo o tempo, melhorando assim o cotidiano do homem e a qualidade de vida sem conhecimento explícito sobre as comunicações e as tecnologias de computação.

A computação pervasiva (*Pervasive computing*), visa fornecer uma computação aonde, quando, o que e como se deseja, através da virtualização de informações, serviços e aplicações. Neste ambiente computacional há possibilidade de existir uma grande variedade de nodos de diversos tipos, móveis ou fixos, aplicações e serviços interconectados. O grau de utilização da arquitetura inserida indica, de maneira geral, o grau de inteligência dos computadores em um ambiente pervasivo, para detectar, explorar e construir dinamicamente modelos de seus ambientes. Na tabela 1, observa-se o grau de utilização de arquitetura embarcada e mobilidade, comparada a computação pervasiva, móvel e ubíqua.

.

Tabela 01: Dimensões comparativas (tabela extraída de [LYY02])

	Computação Pervasiva	Computação Móvel	Computação Ubíqua
Mobilidade	Baixa	Alta	Alta
Grau de "embarcamento"	Alto	Baixo	Alta

Neste caso, computação pervasiva também é sinônimo de conectividade pervasiva, e reflete uma computação altamente dinâmica e distribuída. Hoje, a concentração das pesquisas está em disponibilizar tecnologias - chave, para suportar este novo ambiente computacional. Dentro dessas concentrações, este trabalho apresenta um modelo gerenciador de descoberta de serviços pervasivos (MgDsP) para utilização em aplicações móveis, sensíveis ao contexto, voltado para utilização em ambientes diversos, com a finalidade de adaptação em plataformas e arquiteturas de uso pervasivo.

1.1 Motivação

1.1.1 Cenário 1 – Ambientes Hospitalares

As novas tecnologias de computação móvel e o crescente emprego de dispositivos portáteis têm tornado a computação cada vez mais presente na realização de diversas atividades humanas, favorecendo o surgimento de um novo paradigma computacional: a Computação Ubíqua. Nesse cenário, destacam-se as aplicações móveis, sensíveis ao contexto, que aprimoram a interação com os seus usuários ao se beneficiarem do uso de informações contextuais utilizadas na descoberta de serviços, dentre as quais, destacam-se aplicações hospitalares [CFP06]. A figura 2 apresenta o funcionamento de uma aplicação pervasiva sendo utilizado em um ambiente hospitalar.

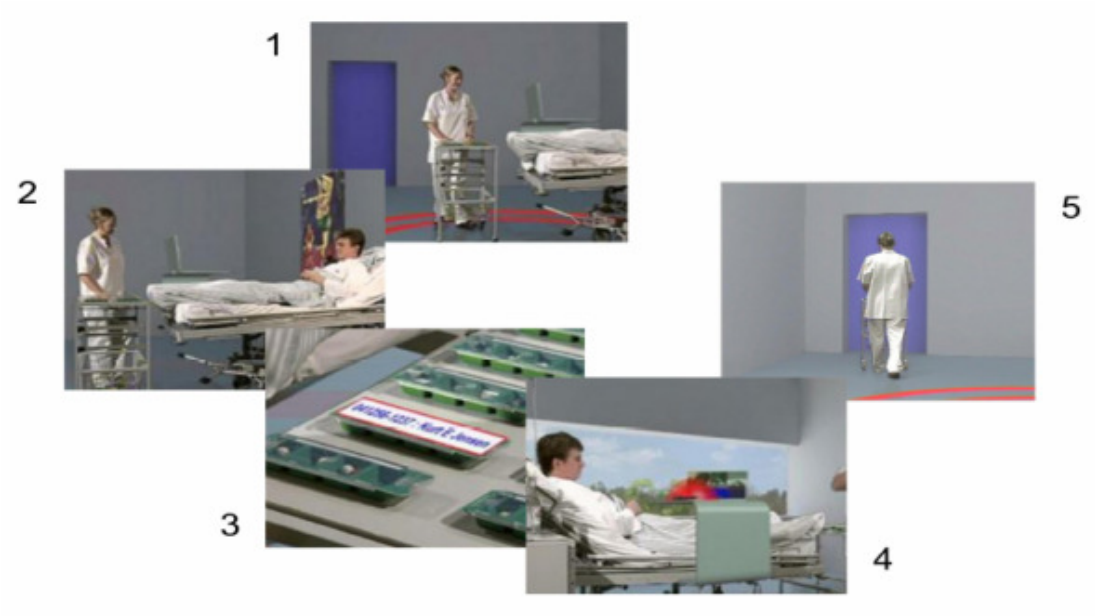

Figura 02: Aplicações hospitalares em 5 visões ilustradas
(Figura extraída de [BAR08]).

Cena 1: Apresenta a enfermeira sendo detectada por sensores, bem como a localização do paciente. A detecção é realizada por sensores localizados no carrinho utilizado pela enfermeira e a cama utilizada pelo paciente. Os sensores são detectados pelo sistema pervasivo, através de dispositivos localizados no ambiente local "leito do paciente" nesta ocasião (*context awareness*). Essas informações são armazenadas no gerenciador do sistema pervasivo, algumas informações como a localização do paciente já estavam previamente armazenadas e disponibilizadas para uso da enfermeira.

Cena 2: Na cena 2 a cama possui um monitor que pode ser utilizado pelo paciente para entretenimento, possuindo outras finalidades, tais como acesso aos dados e informações do paciente. O usuário que for utilizar o leito tem sua identificação previamente cadastrada ao sistema para posteriormente ser detectada e gerenciada pelo sistema. A enfermeira que está atendendo o paciente também possui prévio cadastro ao sistema pervasivo. As informações são relacionadas posteriormente a informações relativas ao paciente.

Cena 3: O tipo de medicamento é identificado através de *containers* com dispositivos de comunicação que são detectados por sensores. Ao se aproximar do paciente, é identificado o medicamento mostrando no *display* seu nome, proporcionando assim, uma maior segurança.

Cena 4: Quando os medicamentos que estão dentro dos *containers* são colocados em cada cama, é realizada a identificação do tipo de medicamento utilizado pelo paciente e baseado nessas informações do paciente e do medicamento o sistema prescreve a quantidade de medicação a ser consumida.

Cena 5: A última cena mostra a enfermeira saindo da zona de detecção local do leito do paciente. O sistema gerencia o monitor do paciente e faz com que volte a opção de televisão ou outra função de entretenimento solicitado pelo paciente.

1.1.2 Cenário 2 – Ambientes Inteligentes

Ambiente inteligente é um novo paradigma que suporta arquiteturas pervasivas, introduzindo interação entre homem, máquina e seu ambiente, um ambiente é denominado Ambiente Inteligente, quando este não interfere e onde as diversas tecnologias complementam uma a outra, envolvendo o usuário ao sistema pervasivo sem a necessidade de manipulação. A capacidade desse ambiente ter disponibilidade de serviços e características que são requeridas e previsíveis a esse ambientes torna – se possível quando:

- Reconhecer o usuário e as circunstâncias (*context user*) e operar conseqüentemente sem necessidade de configuração;
- Ter um conhecimento previsível baseado em conhecimento do ambiente (*context awareness*);
- Em tempo real, dependendo de a aplicação produzir novos serviços em áreas como entretenimento, segurança, trabalhos domésticos, hospitalares, provendo acesso a informações, comunicação, computação;
- Permitir acessar aos muitos serviços e características do ambiente inteligente, independente da localização dos usuários e tipo de dispositivos, fornecendo assim, suporte a descoberta de serviços existentes.

Baseado nessas características para utilização de sistemas pervasivos surgem projetos como o OSGi [LIL05], que trabalha na área de pesquisa e desenvolvimento em computação móvel utilizada para automóveis inteligentes [LIL05]. A figura 3 apresenta a arquitetura completa de automóveis inteligentes, onde se observa um automóvel interligado ao sistema OSGi, entre diferentes tipos de dispositivos e formas de comunicação, comunicando-se a um servidor. As informações disponibilizadas pelo veiculo são navegação, diagnósticos do funcionamento do próprio veiculo, dentre outras informações. Destaca-se, um caso ilusório de um acontecimento relacionando o veiculo a um acidente, o sistema se encarregaria de acionar outro sistema, exemplo citado no cenário anterior. Este, por sua vez, receberia todos os dados relativos ao paciente, inclusive sua localização para um possível deslocamento de ambulâncias.

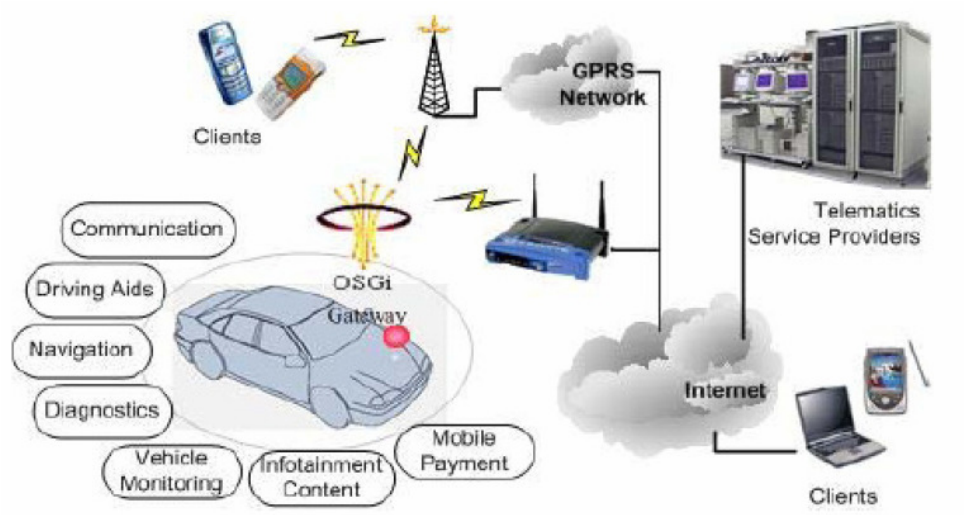

Figura 03: Arquitetura de Automóveis Inteligentes [LIL05]

1.2 Desafios

A utilização de um serviço de descoberta de serviços pervasivos proporciona comodidade e uma maior quantidade de aplicações a serem utilizadas, mas para que essas aplicações tenham desempenho e estejam disponíveis sempre que houver necessidade, há uma série de requisitos a serem verificados em seu contexto, por exemplo: segurança, privacidade, disponibilidade, velocidade, entre outros.

Uma aplicação pervasiva utiliza os contextos individuais e apropriados para cada usuário ou situações em que esses usuários se encontrem. Em um ambiente hospitalar isso pode ser utilizado da seguinte forma: Um paciente pode ter um contexto relacionado a um tipo de medicamento usado para um tipo de doença específica, tendo um contexto definido de apenas alguns profissionais habilitados a manipular essas informações, necessitando, assim, implementações mais seguras e confiáveis. Tornando assim, a manipulação da descoberta de serviços mais complexa. Podemos utilizar como exemplo de aumento dessa complexidade a quantidade de dispositivos que essas aplicações requerem e necessitam manipular.

Os cenários mostrados nas figuras 2 e 3 oferecem uma pequena amostra do potencial de aplicações ubíquas. Na medida em que o hospital, o meio de transporte e o trabalho do usuário tornam-se os cenários de aplicações ubíquas, oferecendo serviços como segurança, comodidade, informação, entretenimento, um universo de dispositivos diferentes deverá existir para suportar estes serviços. Entre outros exemplos que podem ser aplicados nos ambientes mostrados nas figuras citadas anteriormente, também incluem: controladores, sensores e atuadores. A existência de

múltiplos dispositivos seja para o acesso a informação, ao entretenimento, inseridos em utensílios domésticos, inseridos em ambientes inteligentes, constitui-se em um dos desafios da computação ubíqua. Uma tentativa de classificar esses dispositivos é ilustrada na tabela 2, extraída de [HAN01], [BUR02].

Tabela 02: Classificação dos dispositivos ubíquos (Tabela extraída de [HAN01]).

Controles Inteligentes	**Utensílios Inteligentes**
Controles de processo de manufatura	Quiosques
Controles residenciais	Terminais de Ponto de Venda
Termostatos	Centrais de telecomunicações
Etiquetas inteligentes	Eletro domésticos da linha branca (geladeiras,
Controles de bombas de ar, água, gás etc.	maquinas de lavar roupa etc.)
Cartões inteligentes	Terminais de Caixa eletrônico
Controles em sistemas automotivos	Maquinas de venda automática
	Instrumentos de monitoramento medico
Dispositivos de Acesso a Informação	**Sistemas de Entretenimento**
Telefones Celulares	TV
Telefones de tela	Caixas digitais – *set-top-box*
Agentes Pessoais Digitais - PDAs	Consoles de jogos
Computadores de mão	Câmeras digitais
Computadores portáteis	Brinquedos inteligentes
Pagers	Reprodutores de musica MP3

1.3 Objetivos

O principal aspecto a ser considerado na utilização de um serviço de descoberta para sistemas ubíquos é a capacidade de oferecer suporte a mobilidade de usuários e dispositivos, permitindo a adaptação dinâmica às condições variáveis dos ambientes que o integram. Está adaptação deve ser transparente para o usuário provendo, sempre que possível mecanismo para interação espontânea entre as aplicações do usuário e serviços, disponibilizando a troca de informações em quantidade de dispositivos e formas de comunicação heterogêneas. O objetivo deste trabalho é apresentar um modelo gerenciador de descoberta de serviços pervasivos (MgDsP) para utilização de aplicações móveis sensíveis ao contexto.

2 REFERENCIAL TEÓRICO

A computação pervasiva é um novo paradigma que contempla um novo cenário computacional com aplicações e novas funcionalidades em ambientes inteligentes. Essas aplicações são distribuídas e utilizam recursos de uma grande variedade de dispositivos interligados em uma rede altamente heterogênea. Baseado nessas características esse capítulo tem como objetivo definir alguns conceitos relacionados à descoberta de recursos, serviços, protocolos e ciência de contexto, assim como apresentar algumas definições do trabalho. Os conceitos relacionados à descoberta de serviços são apresentados inicialmente. Em seguida, a definição de protocolos de descoberta de serviços é apresentada. No decorrer, serão apresentados conceitos e requisitos relativos à utilização de contexto em ambientes pervasivos. Por fim, esse capítulo descreve os principais requisitos necessários para o desenvolvimento de aplicações pervasivas com descoberta de recursos sensíveis ao contexto.

2.1 Descoberta de Serviços

Ambientes complexos, como o pervasivo, exigem mudanças nos paradigmas de projetos: da orientação a objetos para a programação adaptativa ao contexto (*Context-aware Computing*) [CHE02]. Software adaptativo ao contexto é aquele que usa as informações disponíveis sobre o ambiente para melhorar seu comportamento no tempo. A dificuldade de produzir tais softwares vem do fato de o projetista não poder prever todas as circunstâncias que a aplicação poderá ser usada, e tomar todas as decisões em tempo de projeto. Isto deixa o software com vida curta, devido às constantes atualizações de situações não previstas. Este ambiente é dinâmico, único, com certo grau de incerteza, não - determinístico e tem influência do contexto em que está inserido. Nestes sistemas, o usuário deve sempre obter o que deseja (manter a funcionalidade), saber o que está acontecendo (ter controle sobre o ambiente), saber que não está sozinho. Existem outros sistemas que cooperam, competem e se comunicam (fornecer consciência do contexto). Portanto, o paradigma móvel / pervasivo impõe novos requisitos de operação aos sistemas. Entre eles:

a) Inferir intenção do usuário;
b) Gerenciar pró - ativamente as atividades do usuário;
c) Mover estado de execução entre diversas plataformas e estados de contexto;
d) Armazenar dados e códigos disponíveis em qualquer lugar, em qualquer formato;

e) Prover meios adaptativos a diversos tipos de hw distintos.

Atender a todos esses requisitos exige uma reformulação nos atuais sistemas, e neste trabalho[1] houve estudos exaustivos sobre a discussão sobre questões derivadas da alínea (c, e). Tendo como principal objetivo o presente trabalho o desenvolvimento de um Modelo de Gerenciador de Serviços para Plataforma Pervasiva para utilização de aplicações móveis sensíveis ao contexto.

Em um ambiente de computação pervasiva, usuários e dispositivos formam associações para cooperações dinâmicas, de acordo com seu deslocamento no ambiente [CHE04]. Para isto, cada dispositivo interage com os integrantes de ambiente pervasivo através do uso e oferta de funcionalidades. Faz-se então necessário que estes componentes tenham capacidade de descobrir, utilizar e oferecer tais funcionalidades às demais entidades presentes no ambiente. Tais funcionalidades em um sistema pervasivo são vistas como serviços disponíveis e ofertados por cada dispositivo para potenciais clientes, que entram e saem do ambiente pervasivo. Esta característica impõe que dispositivos possam descobrir tais serviços em suas proximidades. Este problema pode ser solucionado através de um mecanismo específico de descoberta de serviços (*discovery service*). Este serviço funciona como uma espécie de serviço de diretório, onde ofertas de serviços são registradas para serem usadas por potenciais clientes. Dentre suas funções estão à oferta, descoberta, escolha e uso de serviços oferecidos pelos diversos componentes presentes em um dado instante. Alguns exemplos de implementações destes serviços são o Service Location Protocol, Jini e Universal Plug and Play [MUR01].

No caso de aplicações pervasivas, as implementações destes serviços devem levar em conta as características destes sistemas, como: (i) dados relativos às consultas feitas pelos potenciais clientes são determinados em função do contexto do usuário, que por sua vez é altamente dinâmico; (ii) via de regra, não há uma infra-estrutura pré-determinada para hospedar um serviço de descoberta de serviços, como um nodo com endereço fixo; (iii) serviços registrados no serviço de descoberta podem desaparecer espontaneamente; (iv) e por fim, os protocolos para descoberta de serviços devem ser sensíveis a limitações de consumo de bateria e recursos de rede, como largura de banda. Esse trabalho trata somente da descoberta do serviço no ambiente pervasivo, outras questões relativas às formas de comunicação, tratamento de serviços, transporte dos dados entre os dispositivos, não serão tratados nesse estudo.

[1] O objetivo não é reduzir a natural complexidade dos sistemas pervasivos adaptativos, e sim, desenvolver ferramentas que simplifiquem a sua usabilidade em uma quantidade maior de dispositivos.

2.2 Protocolos de Descoberta de Serviços

A rápida evolução e expansão de ambientes dotados de redes sem fio têm aumentado a necessidade por protocolos de descoberta de serviços (SDPs) mais sofisticados [MUR01]. Entretanto, desde que a comunidade de redes de computadores percebeu as necessidades relativas à descoberta de serviços, diversas empresas, consórcios e um grupo de trabalho da IETF (*Internet Engineering Task Force*) passaram a desenvolver pesquisas nesta área. Como resultado, uma grande variedade de protocolos de descoberta de serviços se encontra disponível atualmente, segundo [ZHU02]:

- Service Location Protocol (SLP);
- Jini;
- Salutation;
- Universal Plug and Play (UPnP);
- Bluetooth Service Discovery Protocol (SDP);
- Intentional Naming System (INS).

Protocolos de descoberta de serviços provêem mecanismos para descobrir dinamicamente os serviços disponíveis em uma rede, fornecendo as informações necessárias para procurar serviços específicos, escolher o serviço adequado (aquele com as características desejadas), e utilizar o serviço. Tipicamente, descoberta de serviços, envolve um cliente, um provedor de serviço e um servidor de diretórios ou busca. Além disso, o registro e a busca de serviços são componentes importantes dos protocolos SDPs mais comuns [AVA02]. Nas próximas seções serão discutidas brevemente as arquiteturas desses protocolos e interações entre as instâncias para a descoberta de serviços, registro de serviços e anúncio de serviços.

2.2.1 Service Location Protocol (SLP)

O SLP [AVA02] foi desenvolvido pelo grupo de trabalho da IETF e está disponível em sua versão 2. Seu objetivo é ser um padrão independente de fabricantes. Foi projetado para redes TCP/IP e é escalável para redes de grande porte. Sua arquitetura é constituída de três componentes principais: Agentes do Usuário (UA), que executam a descoberta do serviço em nome do cliente (aplicação ou usuário); Agentes de Serviço (SA), que anunciam a localização e características dos serviços, em nome dos próprios serviços; e Agentes de Diretório (DA), que coletam endereços de

serviços e informações recebidas dos SAs em suas bases de dados e respondem às requisições de serviços dos Uas.

2.2.2 Jini

A tecnologia Jini, segundo [ZHU02], é uma extensão da linguagem de programação Java e foi desenvolvida pela Sun Microsystems. A abordagem da questão de como dispositivos se conectam uns com os outros para formar uma rede ad-hoc simples é tratada através de uma comunidade Jini, e também como esses dispositivos provêem serviços para outros dispositivos nessa rede. Jini é constituído de uma arquitetura e um modelo de programação. O princípio da arquitetura Jini é similar a do SLP. Dispositivos e aplicações se registram com a rede Jini utilizando um processo chamado Descoberta e Adesão (*Discovery and Join*). Para se unir a uma rede Jini, um dispositivo ou aplicação se registra na Tabela de Consulta (Lookup Table) de um Serviço de Consulta (Lookup Service - LS), que é à base de dados para todos os serviços na rede (similar ao DA no SLP).

2.2.3 Salutation

Salutation é mais uma solução para descoberta de serviços, segundo [LIM07]. É um consórcio aberto de indústrias, chamado Consórcio Salutation sendo responsável pelo desenvolvimento de sua arquitetura e definição de um modelo abstrato com três componentes: Cliente, Servidor e Salutation Manager (SLM). O Salutation Manager gerencia toda a comunicação, e faz a ponte através de diferentes meios de comunicação. Serviços registram suas capacidades com um SLM, e clientes consultam o SLM quando necessitam de um serviço. Depois de descobrir um serviço desejado, clientes podem solicitar a utilização do serviço através do SLM. Salutation define seu protocolo baseado no SunRPC.

2.2.4 Universal Plug and Play (UPnP)

Universal Plug and Play (UPnP) [MUR01], foi desenvolvido por um consórcio empresarial fundado e liderado pela Microsoft. Pode-se dizer que ele estende a tecnologia Plug and Play da

Microsoft para o caso em que dispositivos podem ser acessados através de uma rede TCP/IP. UPnP é voltado para redes sem gerenciamento, ou seja, pequenas redes de escritório ou residências, onde ele habilita mecanismos peer-to-peer para auto-configuração de dispositivos, descoberta de serviços, e controle de serviços. UPnP é um protocolo de descoberta de serviços orientado a dispositivos. Todas as informações sobre os serviços e comunicação estão no formato XML, que é independente de plataforma e de linguagem de programação, o que aumenta bastante a interoperabilidade entre dispositivos. Na atual versão do UPnP (release 0.91) não há um serviço central de registro, como o DA no SLP ou o LS de Jini.

2.2.5 Bluetooth Service Discovery Protocol (SDP)

Bluetooth [BIL03] é uma recente tecnologia de transmissão sem fio para curtas distâncias. A pilha de protocolos Bluetooth contém o Service Discovery Protocol (SDP), que é usado para localizar serviços fornecidos por dispositivos disponíveis através do Bluetooth. SDP é baseado na plataforma Piano da Motorola e foi projetado para atender à natureza dinâmica das comunicações ad-hoc. O protocolo também trata a descoberta de serviços especificamente para esse ambiente, e suporta as seguintes requisições: busca por serviços pelo tipo de serviço, busca por atributos de serviços, e busca por serviços sem um prévio conhecimento das características do serviço. SDP não inclui funções para acessar serviços. Uma vez que os serviços tenham sido descobertos pelo SDP, eles devem ser selecionados, acessados e utilizados por mecanismos fora do escopo do SDP, utilizando outros protocolos de descoberta de serviços como SLP ou Salutation. SDP pode coexistir com outros protocolos de descoberta de serviço, mas não necessita deles.

2.2.6 Intentional Naming System (INS)

No Sistema de Nomes Intencionais (INS) [ADJ99]. Aplicações podem ser serviços, que provêem funcionalidades ou dados, ou clientes, que solicitam acesso a esses serviços. *Intentional Name Resolvers* (INRs) faz o roteamento das solicitações dos clientes até os serviços adequados. Qualquer dispositivo em uma rede ad-hoc pode agir como um INR, e uma rede de INRs cooperativos provêem um serviço de descoberta para um sistema completo. INRs formam uma rede sobreposta no nível da aplicação que permite a troca de descrições de serviços e construção de

caches locais baseados nestes anúncios de serviço. Cada serviço conecta um INR e divulga uma descrição baseada em pares de atributo-valor. Clientes se comunicam com INRs e solicitam serviços usando uma expressão de consulta. Uma vez que as descrições de serviço são disseminadas através de rede de INRs, um novo serviço se torna conhecido para outros INRs e através deles para os clientes.

2.3 Comparativo entre Protocolos para Descoberta de Serviços

Em sua tese [LIM07], propõe um protocolo com possíveis soluções para a descoberta de serviços utilizada em redes sem fio. Também cita a concentração em três cenários distintos: (*i*) as soluções projetadas para redes fixas adaptadas às redes sem fio, (*ii*) as soluções projetadas para as redes sem fio *ad hoc* de salto único e (*iii*) as soluções projetadas para as redes sem fio *ad hoc* de saltos múltiplos. E por fim apresenta uma tabela de classificação que é utilizada como base de comparação entre as principais iniciativas de prover mecanismos de descoberta de serviços para as redes sem fio. A classificação apresentada é utilizada como base de comparação entre as principais iniciativas de prover mecanismos de descoberta de serviços sem fio, onde são realizadas diversas comparações entre os protocolos apresentados anteriormente. De acordo com [LIM07], a tabela 3 mostra as principais iniciativas de prover mecanismos de descoberta de serviços para as redes sem fio. São analisados vários aspectos relativos ao uso de protocolos em computação, móvel, pervasiva e conforme estudado no capitulo anterior, utilizando a união entre mobilidade e pervasiva, surgindo assim à computação ubíqua. Utiliza comparações entre protocolos utilizados em redes fixas adaptadas a redes sem e soluções de redes sem fio ad hoc de saltos únicos e múltiplos. As características analisadas são arquitetura de descoberta, escopo, descrição de serviços, métodos de busca e armazenamento do serviço, anuncio, requisição, seleção e invocação de serviços. Também são analisadas as características de provisão de mobilidade e uma característica importante analisada nos protocolos estudados, a segurança, visto que essas aplicações exigem em sua maioria tratamentos adequadas principalmente nesse quesito. Entre todos os trabalhos estudados verificaram-se questões relativas à adaptação ao modelo proposto.

Tabela 03: Comparação entre protocolos (Extraída de [LIM07]).

	Soluções para redes fixas adaptadas às redes sem fio				Soluções para redes sem fio *ad hoc* de salto único		Soluções para redes sem fio *ad hoc* de saltos múltiplos						
	Salutation	**SLPv2**	**Jini**	**UPnP SSDP**	**Bluetooth SDP**	**DEAPspace**	**GSD**	**Allia**	**Konark Gossip**	**ORION**	**[Varshavsky et al. 2005]**	**FTA**	**P2PDP**
Arquitetura de Descoberta	Diretório centralizado hierárquico ou plano	Diretório centralizado plano ou distribuído	Diretório centralizado	Diretório distribuído	Diretório distribuído	Diretório distribuído	Diretório distribuído	Diretório distribuído	Diretório distribuído	Diretório distribuído	Diretório distribuído	Diretório distribuído	Diretório distribuído
Escopo	Topologia da rede (PANs a WANs)	Topologia da rede e papéis de usuário (escopos)	Topologia da rede, papéis de usuário e contexto	Topologia da rede (PANs a WANs)	Topologia da rede	Topologia da rede	Topologia da rede	Topologia da rede	Topologia da rede	Topologia da rede	Topologia da rede	Topologia da rede	Topologia da rede e contexto
Descrição de Serviços	Pares atributo-valor baseados em *templates* e elementos pré-definidos (unidades funcionais)	Pares atributo-valor baseados em *templates* de atributos	Descrição de pares atributo-valor utilizando linguagem de programação	Descrição em XML baseada em *templates* UPnP	Pares atributo-valor baseados em *templates* e elementos pré-definidos	Pares atributo-valor	Ontologia	Independente	Descrição em XML, com *templates* pré-definidos	Pares atributo-valor	Independente	Não tratada (considera-se o uso de ontologias)	Descrição de pares atributo-valor utilizando linguagem de programação
Métodos de Busca (matching)	Baseado na comparação de vários atributos	Comparação de vários atributos utilizando predicados lógicos	Dependente da linguagem de programação utilizada	Baseado na comparação de um único atributo	Baseado na comparação de vários atributos do tipo UUID	Baseado na comparação de vários atributos	Semântico	Compatível com o mecanismo de descrição adotado	Semântico e baseado na comparação de diferentes atributos	Baseado na comparação de diferentes atributos	Módulo de comparação adaptável	Não Tratado	Dependente da linguagem de programação utilizada
Armazenagem da Informação de Serviço	Centralizada, mantida como *hard state*	Centralizada ou distribuída não-cooperativa, mantida como *soft state*	Centralizada, mantida como *soft state* (*leasing*)	Distribuída não cooperativa, mantida como *soft state*	Distribuída cooperativa, mantida como *hard state*	Distribuída não cooperativa, mantida como *soft state*	Distribuída cooperativa, mantida como *soft state*	Distribuída cooperativa, mantida como *soft state*	Distribuída não cooperativa, mantida como *soft state*	Distribuída cooperativa, mantida como *hard state*, (uso da política LRU	Distribuída cooperativa: mantida como *soft state*	Distribuída cooperativa: mantida como *soft state*	Distribuída cooperativa, mantida como *hard state*
Anúncio de Serviços	*Broadcast*	*Multicast*	*Multicast* e notificação por inscrição	*Multicast*	Não Tratado	*Broadcast*	*Broadcast* limitado ao diâmetro do anúncio	*Broadcast* limitado à aliança	Anúncios incrementais, através de *multicast*	Não Tratado	A difusão depende do protocolo de roteamento usado	*Broadcast* limitado pelo número de saltos	Não Tratado

Continuação da tabela comparativa entre protocolos (Extraída de [LIM07]).

	Soluções para redes fixas adaptadas às redes sem fio				Soluções para redes sem fio *ad hoc* de salto único		Soluções para redes sem fio *ad hoc* de saltos múltiplos						
	Salutation	**SLPv2**	**Jini**	**UPnP SSDP**	**Bluetooth SDP**	**DEAPspace**	**GSD**	**Allia**	**Konark Gossip**	**ORION**	**[Varshavsky et al. 2005]**	**FTA**	**P2PDP**
Requisição de Serviços	Requisições são enviadas ao *Salutation Manager* (SLM) via *unicast* ou através de *broadcast* quando se usa diretório distribuído	*User Agents* (UAs) enviam requisições em *unicast* para o *Directory Agent* (DA) ou através de *multicast* para os *Service Agents* (SAs)	Serviços, clientes e LUSs (*LookUp Services*) usam *multicast* ou fazem requisições aos LUSs via *unicast*	Requisições via *unicast* ou *multicast*	Requisições através de *broadcast*	Não Tratada	Encaminhada, em modo seletivo, com base no(s) grupo(s) do serviço	Encaminhada para as plataformas de agentes vizinhas, utilizando *broadcast* ou *multicast*	Encaminhada para o endereço de grupo do Konark (*multicast*)	Encaminhada por *multicast* ou *broadcast*	Disseminadas através de *broadcast* (ex. DSR e AODV) ou *unicast* (ex. CBRP)	Encaminhada à instância do serviço mais próximo ou com maior gradiente do campo	Encaminhada por *broadcast* local, limitado pelo número de saltos
Seleção de Serviços	Manual	Manual	Manual	Manual	Manual	Manual	Manual	Manual	Manual	Manual	Automática	Automática	Automática
Invocação de Serviços	Localização do serviço, mecanismo de comunicação	Localização do serviço (URLs)	Localização do serviço mecanismo de comunicação e operações	Localização do serviço, mecanismo de comunicação e operações	Localização do serviço	Não tratada	Não Tratada	Não Tratada	Localização do serviço, mecanismo de comunicação e operações específicas	Localização do serviço e mecanismo de comunicação	Não Tratada	Não Tratada	Localização do Serviço
Provisão de Suporte à Mobilidade	Reativa	Proativa e Reativa	Proativa e Reativa	Proativa e Reativa	Implícita	Proativa	Proativa	Proativa	Proativa e Reativa	Reativa	Proativa e Reativa	Proativa	Implícita
Segurança	Autenticação do usuário no RPC	Uso opcional de assinaturas digitais (autenticação)	Nível de segurança provido pela linguagem de programação (Java)	Extensões oferecem um *framework* para controle de acesso e autenticação de dispositivos	Estabelecida na fase de negociação da conexão; a comunicação pode ser criptografada no nível de enlace	Não Tratada	Não Tratada	Uso de políticas locais para definir restrições de segurança, como direito de acesso e verificação de credenciais	Não Tratada	Não Tratada	Não Tratada	Não Tratada	Não Tratada

2.4 Estudo sobre Protocolo Escolhido

O protocolo escolhido foi o Bluetooth Service Discovery Protocol (SDP), entre outros fatores o fato de maior relevância é que Bluetooth é uma tecnologia desenvolvida essencialmente para ser usada em sistemas embarcados onde a capacidade de memória, processamento, entre outros, são bastante limitados. Outro fator é ter sido utilizado em parte da validação na arquitetura desenvolvida por [RIE06]. Segundo o resumo descrito na seção 2.3, também pode ser facilmente adaptado a outras plataformas pervasivas de uso genérico, sendo um dos objetivos deste trabalho.

2.4.1 Descrição do Protocolo em Camadas

Segundo [BIL03], o sistema Bluetooth foi especificado por meio de uma pilha de protocolos, formando uma pilha de cinco camadas, sendo elas:

- Bluetooth Radio – especifica detalhes da interface com o ar, incluindo freqüência, salteamento, esquema de modulação e força da transmissão;
- Baseband – trata sobre estabelecimento de conexão com uma piconet, endereçamento, formato do pacote, temporização e controle de energia;
- Link Manager Protocol (LMP) – estabelece a configuração do link entre dispositivo bluetooth e gerenciamento de links em andamento, incluindo aspectos de segurança (ex. autenticação e encriptação), controle e negociação do tamanho do pacote da banda base;
- Logical Link Control and Adaptation Protocol (L2CAP) – adapta os protocolos da camada superior à camada de banda base, fornecendo tanto serviços sem conexão quanto serviços orientados à conexão;
- Service Discovery Protocol (SDP) – manipula informações do dispositivo, serviços e consultas para características de serviço entre dois ou mais dispositivos Bluetooth.

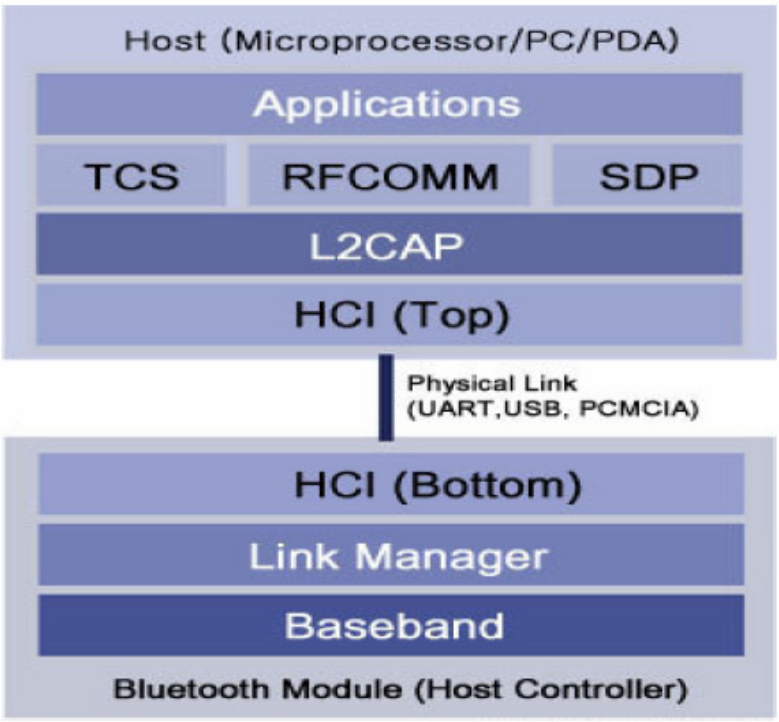

Figura 04: Pilha de Protocolos Bluetooth (Figura extraída de [BIL03])

Inquiry: O dispositivo realiza um processo conhecido como *inquiry*. Neste processo, ele "pergunta" quais pontos de acesso estão próximos. Todos os pontos de acesso respondem com seu endereço físico (cada módulo Bluetooth tem um endereço físico único no mundo, gravado no momento da fabricação) [BIL03]. O dispositivo móvel escolhe então um destes pontos de acesso.

Paging: A seguir, é iniciado um processo conhecido como *paging*. Neste processo, o dispositivo móvel procura sincronizar-se com o ponto de acesso, em termos de *clock offset* e fase do salto em freqüência, além de outras inicializações necessárias.

Estabelecimento de um link: Como se trata de uma aplicação que não demanda um fluxo contínuo de dados, é estabelecido um link assíncrono (ACL). A camada responsável por estabelecer tal link é a LMP (Link Manager Protocol).

Procura por Serviços: Depois de estabelecido um link físico, o dispositivo móvel procura descobrir por meio do SDP (Service Discovery Protocol), quais os serviços disponíveis no ponto de acesso. No caso, verificará se é possível acessar algum serviço a partir do ponto de acesso. Suponha-se que seja possível, pois caso contrário à aplicação não funcionaria.

L2CAP: Baseado na informação obtida pelo SDP, um canal L2CAP será criado para possibilitar a comunicação entre os dois dispositivos.

RFCOMM: Um canal RFCOMM é criado sobre o canal L2CAP. O canal RFCOMM transmite dados entre os dispositivos Bluetooth através de uma interface semelhante a um socket.

Segurança: Caso o ponto de acesso restrinja o acesso a um grupo específico de usuários, é realizado um processo de autenticação, onde o dispositivo móvel deve saber o PIN correto para acessar o serviço. Além disso, se os dispositivos estiverem operando no modo seguro, os dados serão encriptados. Entretanto, o quesito segurança não foi tratado nesse trabalho, disponibilizando assim, proposta para trabalhos futuros.

PPP: Para acessar um recurso de dispositivo, tendo como finalidade acessar um serviço de email, é usado o TCP/IP. No nível da aplicação do TCP/IP, tem-se o protocolo POP responsável pelo acesso à conta de e-mail. Este protocolo faz uso de uma conexão PPP. O PPP, geralmente é executado a partir de um link serial (um modem em uma conexão dial-up, por exemplo). No caso do sistema Bluetooth, o protocolo PPP rodará a partir do canal RFCOMM, que emula similarmente um link serial. A partir daí, os protocolos usuais da Internet pode ser executado normalmente e, neste caso, o e-mail pode ser acessado.

TCS: Telephony control protocol-binary (TCS BIN) é o protocolo orientado a bit que define o controle de chamada de sinalização para estabelecimento de chamadas de voz e dados entre dispositivo Bluetooth. Ainda, TCS BIN define procedimentos de gerenciamento de mobilidade para manipular grupos de dispositivo Bluetooth TCS, entretanto esse módulo da pilha de protocolos não será utilizado no trabalho.

HCI: A camada HCI fornece uma interface de Software básica para que sejam desenvolvidas camadas superiores. Métodos de transmissão/recepção (send/receive), são alguns dos métodos providos por esta camada. (Hardware Controller Interface) que oferece uma interface necessária para: transmissão/recepção de pacotes, envio de comandos ao controlador de Hardware, dentre outros. Já a outra camada, permite a comunicação entre HCI e o Hardware. Ela oferece o suporte necessário de maneira que a comunicação realize-se independente do BUS.

2.5 Ciência de Contexto

Nesta seção serão apresentados os principais conceitos e funcionalidades sobre aplicações Sensíveis ao contexto.

2.5.1 Conceitos

Seres humanos são capazes de trocar idéias e refletir sobre novas informações de maneira apropriada devido a alguns fatores, como: o vocabulário que compartilham, os entendimentos de como as coisas funcionam e um entendimento implícito das situações do cotidiano [DEY00]. Quando as pessoas se comunicam, elas são capazes de utilizar informações de situação (ou contexto) para acrescentar algo à conversação. Infelizmente, esta capacidade não é encontrada na comunicação entre humanos e computadores, ainda. Computadores não entendem a linguagem humana, tão pouco entende como as coisas funcionam e não podem obter de maneira independente as informações sobre situações, pelo menos não tão facilmente como os seres humanos. Na computação tradicional, geralmente, os usuários fornecem informações aos computadores por meio de teclado e mouse utilizando, portanto uma maneira explícita de entrada de dados. A computação ubíqua prioriza o desenvolvimento de aplicações cujas interfaces humano computador exploram as formas naturais de interação humana (ex: fala gestos e escrita). Para que os usuários não necessitem fornecer informações de maneira explícita ao computador, as aplicações precisam ser capazes de capturar contextos por meio de sensores físicos e lógicos, e fornecer serviços e informações sem que o usuário precise solicitá-los. Face essas necessidades surgem os chamados tratamentos de contexto computacional, que serão apresentados a seguir.

2.5.2 Contexto

Na literatura encontram-se muitas definições para contexto. Várias definições em contexto são encontradas como localização, identidade de pessoas e objetos próximos, e tratamento de mudanças nesses objetos. A definição de contexto como localização e identificação de pessoas próximas do usuário, controlando informações de hora, período e temperatura do ambiente, é definida por [CAC05]. Outra definição sobre contexto seria a localização e identidade do usuário e de objetos, informações físicas do ambiente e tempo. Dentre as definições existentes, a mais abrangente é fornecida em [DEY00]¸ onde contexto é qualquer informação que possa ser utilizada para caracterizar a situação de uma entidade. Entidade é uma pessoa, um lugar, ou um objeto considerado relevante para a interação entre um usuário e uma aplicação, incluindo o próprio usuário e a aplicação.

Por ser a mais abrangente e referenciada na literatura, essa definição será utilizada no decorrer deste trabalho quando forem usadas as palavras contexto ou informação de contexto.

2.5.3 Utilizações de Contexto

Sistemas pioneiros em ciência de contexto utilizavam informações de localização e identificação de pessoas, via sinais infravermelhos, para prover serviços úteis a usuários, como controle de luzes e temperatura de um dado ambiente e transferência automática de chamadas telefônicas [COS03]. Para obter informações de localização, aplicações mais recentes usam as facilidades providas pelo sistema GPS (Global Positioning System) para prover navegação de carros em cidades e estradas [LIL05]. O sistema GPS também tem sido utilizado em sistemas de guias turísticos inteligentes capazes de escolher um vídeo ou áudio para ser exibido de acordo com informações de localização.

Porém, há muitas outras informações contextuais além de localização e identificação de pessoas e objetos. As maiorias dos sistemas cientes de contexto ainda não incorporam noções de tempo, história, e outras pessoas além do usuário, bem como informações disponíveis nos ambientes. Alguns autores sugerem cinco dimensões para especificação e modelagem de informações de contexto [YAM04], [CAC05]:

- Who (identificação): seres humanos realizam suas atividades e recordam de fatos passados com base na presença das pessoas que os vivenciaram. Sendo assim, o sistema deve prover informações de contexto de todas as pessoas envolvidas em uma determinada atividade;
- Where (localização): assim como as informações de identificação, informações de contexto de localização têm sido muito utilizadas em sistemas cientes de contexto. As pesquisas mostram um interesse particular na utilização dessa informação contextual associada a outras dimensões, por exemplo, a dimensão temporal para explorar a mobilidade de usuários ao longo do tempo;
- When (tempo): informações temporais têm sido usadas para indexar registros capturados ou para informar por quanto tempo um usuário esteve em uma determinada localização. Porém, o conhecimento dessas informações permite fazer inferências ajudando a interpretar as atividades humanas e estabelecer padrões de comportamento. Por exemplo, se uma pessoa idosa passa muito tempo em um banheiro, de forma incomum segundo seu histórico de atividades, há fortes indícios de ela tenha sofrido uma queda e precise de ajuda;

- What (atividade): o objetivo é obter informação, normalmente via sensores, que possibilite interpretar o que o usuário está fazendo. Em sistemas onde várias atividades podem ser desenvolvidas, identificar o que um usuário está fazendo em um determinado momento pode ser uma tarefa complexa;
- Why (intenção): tão difícil quanto perceber o que o usuário está fazendo, é entender o porquê de sua ação. Interpretar informações de contexto que possam caracterizar o estado de uma pessoa talvez seja o maior desafio da computação ciente de contexto.

Geralmente, as informações de contexto que exprimem a atividade e a intenção do usuário são obtidas por combinação de outras informações de contexto (ex: localização, identificação, tempo), ou via sensores como aqueles para medição de temperatura corporal, entonação vocal, batimento cardíaco e pressão arterial.

2.5.4 Tipos de Contexto

Segundo [HEN01], contexto também pode ser classificado como:

- Estático: contexto estático corresponde ao grupo de informações que permanecem fixas durante o tempo de vida da entidade, como números de CPF e RG de uma pessoa. Todas as outras classes de contexto são dinâmicas.
- Dinâmico: informações de contexto dinâmicas podem ser de três tipos de sentido (do inglês sensed), explícito e interpretado.

As informações de contexto dinâmicas sentidas são aquelas capturadas por meio de sensores físicos e lógicos, como localização de uma pessoa e nível de umidade de um ambiente. Informações de contexto dinâmicas explícitas são aquelas fornecidas explicitamente pelo usuário, como sua senha de acesso ou itens de sua agenda particular. A classe de informações dinâmicas interpretadas refere-se aos contextos obtidos de um ou mais contextos por meio de uma regra de interpretação que varia de uma simples condição até algoritmos de Inteligência Artificial mais complexo. Temperatura anormal ou relação de proximidade de entidades são exemplos de contexto dinâmico interpretado.

2.5.5 Requisitos Computação Ciente de Contexto

Dey [DEY00] sugere sete requisitos que um software de computação ciente de contexto deve possuir para facilitar sua utilização, conforme descrito nas seções seguintes.

2.5.5.1 Conceito sobre Informação de Contexto

Provavelmente um dos requisitos mais importantes de um software ciente de contexto é a existência de mecanismos que permitam que uma aplicação especifique quais contextos ela tem interesse (ex: monitoramento da localização do usuário) e que ações devem ser tomadas quando determinado contexto for obtido. Por exemplo, ao monitorar a localização de um usuário, uma aplicação pode transferir chamadas telefônicas para o telefone celular do mesmo quando este não estiver em seu local de trabalho.

2.5.5.2 Tratamento de Informação de Contexto

A carência de uma forma padrão para capturar e tratar informações de contexto dificulta a exploração desse tipo de informação em ambientes de computação ubíqua. Existem duas maneiras comuns de tratar contexto:

- Associar drivers de sensores diretamente nas aplicações;
- Utilizar servidores que omitam os detalhes dos sensores, sem que a aplicação precise se preocupar com detalhes de baixo nível;

Quanto à utilização da informação de contexto, a consulta direta a um sensor é apropriada quando o contexto é necessário somente uma única vez. Quando o contexto houver necessidade de uso repetidas vezes, o mecanismo de publish / subscribe é mais apropriado, pois uma aplicação pode definir um conjunto de condições que determinam quando ela quer ser notificada.

2.5.5.3 Interpretação de Informação de Contexto

Existe a necessidade de estender os mecanismos de notificação e de consulta de sistemas cientes de contexto para permitir que estes recuperem contextos de ambientes distribuídos [EFS06]. Informações de contexto podem passar por muitas camadas de software antes de serem entregues à aplicação devido à necessidade de abstração (interpretação) de mais alto nível. Por exemplo, uma aplicação de captura de informações de reunião precisa ser notificada quando uma reunião estiver para acontecer: em um nível mais baixo, informações de localização e identificação podem ser interpretadas para determinar as várias pessoas se encontram. Em um nível mais alto, esta informação pode ser combinada com registros da agenda dessas pessoas para determinar se a reunião está acontecendo.

Para o desenvolvedor da aplicação, o uso de várias camadas deve ser transparente. Para que isso aconteça, o contexto deve ser interpretado antes de ser usado pela aplicação. Para que a interpretação seja facilmente reutilizada por várias aplicações, ela precisa ser fornecida por um framework [DEY00]. Caso contrário, as aplicações deverão re-programar o mecanismo de interpretação que lhes é necessário.

2.5.5.4 Formas de Comunicação, Distribuição e Transparência

Sensores utilizados para capturar informações de contexto não estão, em sua maioria, alocados no mesmo computador no qual as aplicações cientes de contexto estão sendo executadas. Geralmente, os sensores encontram-se fisicamente distribuídos no ambiente físico [GUT04]. A comunicação distribuída deve ser transparente tanto para os sensores quanto para as aplicações no sentido de que o desenvolvedor de aplicações não necessite especificar e implementar um protocolo de comunicação e um esquema de codificação e decodificação para a transmissão de informações de contexto.

2.5.5.5 Disponibilidade

A busca contínua de componentes de captura de informação de contexto devido ao fato de uma aplicação poder solicitar uma informação de contexto a qualquer momento será necessária aos componentes que capturam contexto sejam executados de forma independente das aplicações, e também que estejam sempre disponíveis [GUT04]. Portanto, os componentes devem ser executados continuamente para permitir que aplicações os consultem sempre que necessário. Em uma conferência, por exemplo, componentes de captura de áudio podem ser requisitados continuamente para o registro de fluxos das inúmeras atividades executadas em paralelo (ex: palestras e tutorais).

2.5.5.6 Armazenar a Informação de Contexto

Devido à necessidade da disponibilidade constante dos dados, faz-se necessário manter históricos de informações de contexto [AUG04]. Um histórico de contexto pode ser usado para estabelecer tendências e predizer futuros valores de contexto. Sem o armazenamento persistente de contextos, esse tipo de análise não poderia ser realizado. Componentes de captura devem adquirir informações de contexto mesmo quando nenhuma aplicação está interessada nas informações de contexto disponíveis naquele momento. Pois, essas informações de contexto capturadas, podem ser utilizadas por uma aplicação que necessite do histórico de um dado de contexto, por exemplo, o histórico de localização de um usuário para predizer sua localização futura.

2.5.5.7 Definições e Parâmetros

Para a utilização de contexto em um ambiente pervasivo distribuído, torna-se indispensável o conhecimento de parâmetros e definições, que auxiliam no desenvolvimento de aplicações para esses ambientes. A tabela 4 descreve alguns parâmetros relacionados ao desenvolvimento de middleware que utilizam aplicações voltadas ao contexto. Segundo [YAM04], existem três parâmetros de comparação: (*i*) o custo computacional que o *middleware* introduz (*ii*) quais os

paradigmas de comunicação que este disponibiliza e (*iii*) que forma de representação de contexto é empregada.

Tabela 04: Classificação e Definição em Sistemas Distribuídos

(Tabela extraída de [YAM04]).

Parâmetros	**Tipos de Sistemas distribuídos**	
	Sistemas distribuídos tradicionais	**Sistemas distribuídos ad hoc e híbridos**
Custo computacional	Composto por nodos fixos, os quais implementam serviços com elevado custo computacional.	Presença de computadores móveis, o que implica em que o núcleo do *middleware* deve ser leve. Deve ser prevista uma instalação seletiva de serviços nos nodos. Se necessário os nodos fixos receberão incumbências de maior custo computacional.
Paradigma de comunicação	São empregadas conexões permanentes, sobre as quais ocorrem comunicações síncronas.	Ocorrência de conexões intermitentes, o que exige suporte para comunicações assíncronas.
Representação do contexto	Pequena dinamicidade do contexto de execução (praticamente estático), o que faculta transparência no mecanismo de adaptação (o *middleware* pode gerenciar sózinho a adaptação).	Elevada dinamicidade do contexto de execução, a qual não permite previsão sobre a disponibilidade de recursos. Esta dinamicidade implica na participação da aplicação na tomada de decisão (uso do conhecimento do usuário), a qual, assim como o *middleware,* deve neste caso ter consciência do contexto em que a execução está ocorrendo.

2.6 Descoberta de Recursos

Para que uma aplicação possa se comunicar com dispositivos de captura de contexto (ex: sensores), esta deve saber que tipos de informações o dispositivo pode fornecer, qual a sua localização e qual o modo de comunicação suportado. Para que a aplicação não precise se preocupar com esses detalhes, a infra-estrutura de software subjacente precisa fornecer suporte à descoberta de recursos. Assim que uma aplicação é iniciada, ela deve especificar o tipo de informação de contexto de seu interesse. Com isso, o mecanismo de descoberta de recursos se responsabiliza por encontrar componentes adequados e fornecer mecanismos de acesso. Segundo [YAM04], a possibilidade de utilizar recursos previamente (estaticamente) conhecidos, diretamente (sem envolver um catálogo), é fundamental para permitir que os recursos sejam descobertos e ligados dinamicamente. As aplicações de interesse operam em ambientes que se modificam constantemente e, para isso, precisam de um serviço de descoberta que localize dinamicamente instâncias de componentes e recursos que satisfaçam as suas necessidades. Existem atualmente serviços que suportam a descoberta de recursos, em termos de topologia de rede ou localização [ZHU02]. Porém, não foram

projetados para considerar especificamente informações contextuais. Cada serviço, geralmente, possui um domínio de aplicação e uma linguagem de consulta específica. Com o objetivo de cobrir estas lacunas, novas arquiteturas têm sido projetadas. Uma proposta é o ONS [BAR02], uma arquitetura para sistemas ubíquos e pervasivos, que provê às aplicações sensíveis ao contexto, identificação transparente de contexto e adaptação de serviços, independente de mudanças no seu contexto. Outra proposta, Q-Cad [COS03], permite as aplicações pervasivas descobrirem e selecionarem recursos considerando o contexto corrente de execução e os requisitos de QoS.

Cita – se outro exemplo de trabalho, que discute a descoberta de recursos é o desenvolvido por [BAR02]. Esse trabalho trata a descoberta de recursos através da arquitetura Pholo [BON05], que fornecendo suporte para o ambiente de execução do Holoparadigma. O Holoparadigma [JOR02] é um modelo multiparadigma voltado para abstrair a modelagem de ambientes móveis e permitir representações mais similares ao mundo real. Sua unidade de modelagem, o ente, permite o desenvolvimento de sistemas tanto centralizados como descentralizados e possui grande capacidade de mobilidade. A arquitetura Pholo é dividida em cinco camadas: camada de rede, sistema operacional, camada de serviços, camada de execução e aplicações. A camada de execução e a camada de serviços são as responsáveis pelos serviços de computação pervasiva na arquitetura. A primeira camada é responsável pela execução dos programas do Holoparadigma. Essa camada é composta por dois serviços: HoloVM e o HNS. O HoloVM é uma máquina virtual que executa as aplicações, de forma distribuída, com suporte à comunicação entre entes, abstraindo toda a complexidade do desenvolvimento. Já o HNS tem o controle de todos os entes em execução, através de uma estrutura em forma de árvore. O objetivo é fornecer informações dos entes para as HoloVMs, como por exemplo, localização de um ente. A camada de serviços é responsável pelo fornecimento dos seguintes serviços: localização física, que informa a localização e aspectos de mobilidade dos entes; contexto para aquisição de informações sobre o mundo real; mobilidade, que consiste na movimentação de um dispositivo físico no espaço físico, na movimentação de um ente modificando o HNS e na migração de um ente (por exemplo, um serviço move-se de um dispositivo para outro); descoberta de serviços, que permite HoloVMs descobrirem servidores HNS e descoberta de serviços desempenhados por entes de uma aplicação.

3 ESTADO DA ARTE

Este capítulo descreve estudos sobre trabalhos desenvolvidos na área de computação ciente de contexto, apresentando as suas vantagens e desvantagens. A seção 3.1 descreve sobre a Plataforma Inflaware. A Seção 3.2 descreve sobre a arquitetura Mixis. A Seção 3.3 apresenta o sistema SOCAM (Service-Oriented Context-Aware Middleware). A Seção 3.4 apresenta o CybreMinder, uma ferramenta ciente de contexto. A Seção 3.5 apresenta a arquitetura AROUND, que possui a característica de oferecer serviços baseados em escopo. A Seção 3.6 apresenta o protótipo de um guia turístico para a cidade de Aalborg. A Seção 3.7 descreve sobre o Flame2008, um sistema para integração de Web services com o intuito de obter ofertas significantes baseadas na situação e no perfil do usuário. A Seção 3.8 apresenta a plataforma Nexus, que procura integrar vários modelos de contexto com o objetivo de construir um modelo global. A Seção 3.9 apresenta um sistema de redirecionamento de mensagens telefônicas, o ICAMS. A Seção 3.10 apresenta o FieldMap, uma ferramenta que permite adicionar comentários em mapas apresentados em dispositivos móveis. A Seção 3.11 apresenta um sistema de monitoração de arritmia, denominado AMS (Arrhythmia Monitoring System). Por fim, a Seção 3.12 apresenta uma comparação entre as ferramentas.

3.1 Plataforma Infraware

Há diversos projetos em andamento, Filho [FIL06] apresenta uma visão geral da plataforma Infraware, um middleware de suporte ao desenvolvimento e execução de aplicações móveis sensíveis contexto que está sendo concebido no âmbito dos projetos DBMWare [GUP02]. A plataforma Infraware é visualizada na figura 5, é um *middleware* baseado em *Web Services* com suporte arquitetural para o desenvolvimento, construção e execução de aplicações móveis sensíveis ao contexto. A arquitetura conceitual da Infraware estende, em vários aspectos, entre os quais a plataforma WASP, um projeto holandês desenvolvido pela *University of Twente*, Telemática *Instituto* e *Ericsson*. A plataforma WASP [GUP02], concentra-se na interface aplicação-plataforma, definindo uma linguagem para especificar como ela deve reagir a uma correlação de eventos. Possui vários aspectos importantes; Entretanto não trata da questão de descoberta de novos serviços.

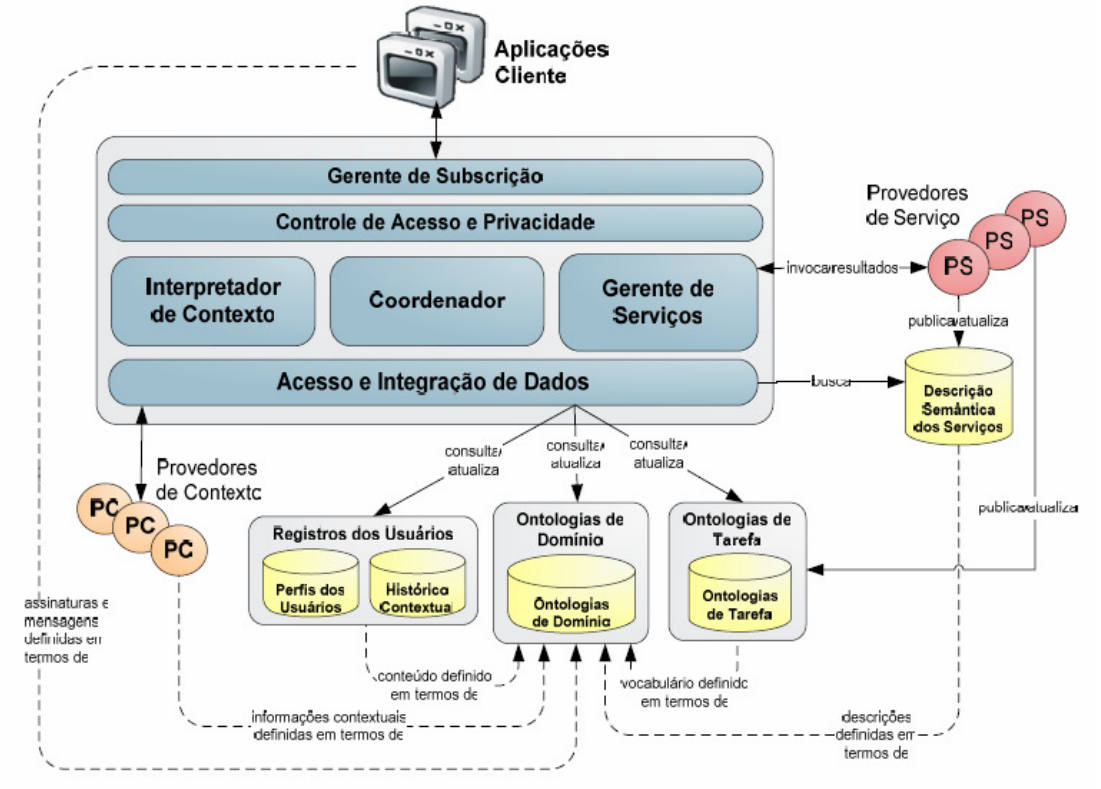

Figura 05: Plataforma Infraware [FIL06].

Entre outras funcionalidades, uma característica marcante da Infraware é o uso de conceitos WebSemântica e ontologias nesta plataforma. Ontologias especificam modelos formais extensíveis que descrevem não somente o domínio das aplicações, também é tratado os serviços. Essa abordagem diferenciada provê meios de configurar as interações da aplicação-plataforma em tempo de execução. A plataforma também pode ser customizada pela adição de novos serviços e entidades estendendo-se as ontologias. Adicionalmente, a adoção de Web Services como tecnologia de distribuição permite que aplicações acessem os serviços oferecidos através de protocolos da Internet e facilita a inclusão de novos serviços à plataforma por terceiros. Essa flexibilidade torna a Infraware adequada ao desenvolvimento de uma larga gama de aplicações [FIL06]. O atendimento integrado a esse conjunto de fatores diferencia a Infraware das demais arquiteturas investigadas.

3.2 Plataforma MIXIS

Outra aplicação que pode ser utilizada é a MIXIS (*Mixed Interaction Space*) que basicamente funciona como um localizador através de um telefone celular, podendo ser adaptável a outros dispositivos pervasivos como *Pda* e *Smartphones*. Além de o sistema possuir diversos *drivers* armazenados localmente para dispositivos móveis, quando o sistema detecta um dispositivo novo faz o *download* de arquivos automaticamente através da webservice [HAN05]. A mixis possui vários mapas que armazenados em sua base de dados, disponibilizando ao usuário a atualização dos mapas existentes ou incluir novos mapas. A figura 6 apresenta um exemplo do diagrama genérico de funcionamento do sistema mixis.

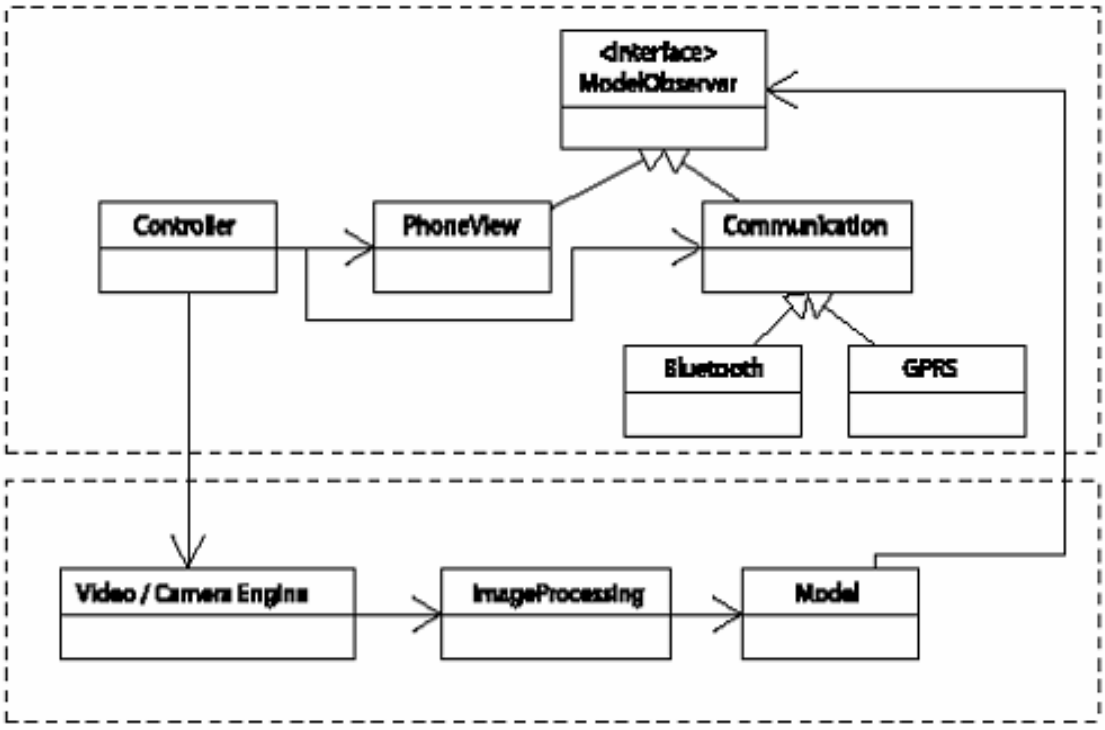

Figura 06: Diagrama genérico da aplicação MIXIs (Figura extraída de [HAN05]).

Este exemplo visualizado na figura 6 exemplifica o funcionamento do diagrama de uso genérico proposto pela aplicação mixis, onde a entidade *controler* detecta o dispositivo móvel e envia para entidade que possui configurações de câmera caso seja um dispositivo novo. Caso os dispositivos já estejam cadastrados na base de dados as informações são enviadas diretamente para a entidade de comunicação. Esta entidade irá receber dados via Bluetooth ou GPRS, enviando os dados para a entidade de interface para que seja disponibilizado pelo receptor. No caso de dispositivo não estiver cadastrado, os dados serão enviados para a entidade image processing que

por sua vez os remete a entidade que controla os modelos com *drivers* disponibilizados, enviando a uma entidade que controla a interface para o receptor. O foco principal desta aplicação é o envio e recepção de imagens. O transmissor pode ser usado para recepção de mapas de localização, onde o receptor pode interagir com o transmissor, este por sua vez irá visualizar a localização do receptor e orientá-lo sobre o caminho a prosseguir, ou visualização de imagens de Raios-X. Um médico poderá de outro local visualizar simultaneamente de diversas direções os dados da imagem sendo recebida a figura 7 pode-se observar algumas utilizações da aplicação. É uma aplicação que pode ser utilizada ou adaptada a outras já existentes, visto que, essa não trata em especifico da questão de descoberta de serviços. Caso houvesse um dispositivo móvel de maior tamanho de tela, a aplicação pode transferir os dados para a tela de maior tamanho, a fim de obter uma melhor visualização das imagens.

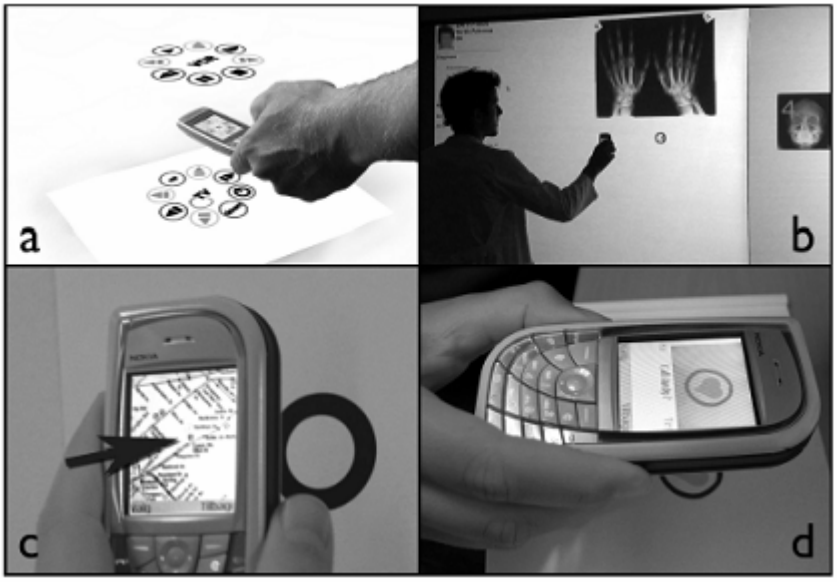

Figura 07: Exemplos de utilização da aplicação MIXIS (Figura extraída de[HAN05]).

3.3 SOCAM

SOCAM [GUZ05] (Service-Oriented Context-Aware Middleware) é uma arquitetura para a construção de um serviço móvel ciente de contexto. O modelo do contexto é baseado em ontologia que provê um vocabulário para representar e compartilhar conhecimento de contexto em um domínio da computação onipresente. O projeto da ontologia do contexto é composto de dois níveis hierárquicos. Um nível contém ontologias individuais sobre vários subdomínios, por exemplo, o

domínio de uma casa, um escritório ou um veículo. Um nível mais alto contém conceitos gerais sobre as outras ontologias, esse nível é chamado de ontologia generalizada ou ontologia de alto nível.

O domínio específico de uma ontologia pode ser dinamicamente re-alocado. Por exemplo, quando um usuário deixa sua casa para dirigir um carro, a ontologia do domínio da casa será trocada automaticamente pela ontologia do veículo [LIL05]. As arquiteturas SOCAM é composta pelos seguintes elementos:

- Provedores de contexto: provêem uma abstração do sensoriamento de baixo nível. Cada provedor de contexto precisa ser registrado em um serviço de registro, o SLS (Service-locating service) para que outros usuários possam descobrir esses provedores. Os provedores de contexto podem ser externos ou internos. Os provedores de contexto externos obtêm o contexto de fontes externas, por exemplo, um serviço de informação do tempo ou um serviço de localização *outdoor*. Os provedores de contexto internos obtêm o contexto diretamente de sensores localizados em um subdomínio, como uma casa, por exemplo;

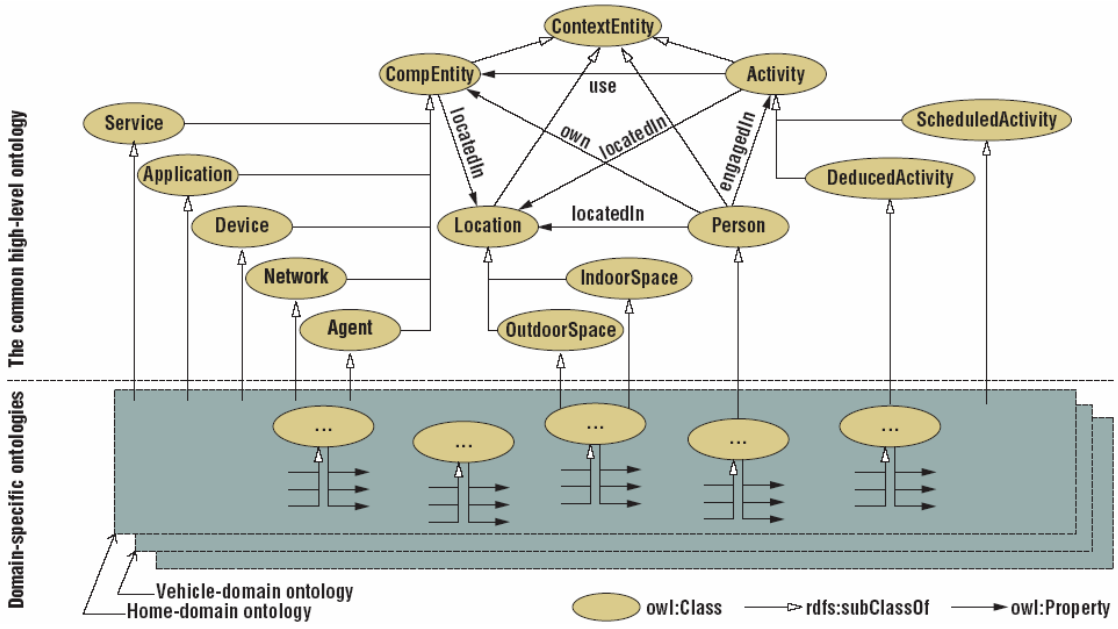

Figura 08: Diagrama hierárquico de uma ontologia de contexto (extraído de [GUZ05]).

- Serviço de Localização de Serviço (SLS): permite a usuários, agentes e aplicações descobrirem e localizarem diferentes provedores de contexto;
- Interpretador de contexto: provê contexto de alto nível através da interpretação de contexto de baixo nível. Dessa forma, o interpretador também é tratado como um provedor de contexto e

pode ser registrado no SLS. O interpretador de contexto é dividido em *reasoner* e *knowledge* base (KB). O *reasoner* tem a função de prover contexto de alto nível baseado no contexto de baixo nível e detectar inconsistência e conflitos na base de conhecimento. KB provê um conjunto de API para que outros componentes possam consultar adicionar, remover ou modificar conhecimento de contexto, além de conter a ontologia de um contexto de um subdomínio e suas instâncias. Essas instâncias podem ser especificadas pelos usuários, no caso de contexto definido, ou adquirido de vários provedores de contexto;

- Serviços cientes de contexto: são aplicações que fazem uso dos diferentes níveis da arquitetura SOCAM. Os desenvolvedores dessas aplicações podem predefinir regras e especificar que métodos devem ser invocados quando uma condição for verdadeira. Todas as regras são salvas em um arquivo e pré-carregadas no *reasoner*.

SOCAM foi projetado como um serviço de componentes independentes que podem ser distribuídos numa rede heterogênea e podem se comunicar entre si. Todos os componentes são implementados em Java. Para a comunicação entre os componentes distribuídos é usado Java RMI, que permite objetos distribuídos invocarem métodos de outros objetos remotos. A interação entre os componentes do SOCAM ocorre, resumidamente, da seguinte forma: um provedor de contexto pode adquirir dados sobre o contexto de vários sensores heterogêneos. Diferentes provedores de contexto registram seus serviços no SLS. As aplicações móveis ciente de contexto são capazes de localizar um provedor e obter dados sobre um determinado contexto. O interpretador de contexto e o serviço móvel ciente de contexto também podem ser registrados no SLS, o diagrama hierárquico pode ser visualizado através da figura 8.

O modelo SOCAM segue uma arquitetura semelhante ao padrão Web Service, na qual os serviços são registrados em um diretório público e podem ser encontrados e utilizados por outros serviços. Porém, a arquitetura não é independente de linguagem de programação, pois os componentes trocam mensagens usando Java RMI, tornando difícil a integração entre servidores heterogêneos. Além disso, as regras de contexto devem ser carregadas previamente no sistema para que passem a funcionar.

3.4 CybreMinder

CybreMinder é uma ferramenta ciente de contexto que ajuda a criar e gerenciar lembranças [DEY00a]. Uma lembrança é um tipo especial de mensagem que enviamos para nós mesmos ou para

outros, para informar sobre atividades futuras que devemos tratar. Por exemplo, um colega de trabalho nos envia uma mensagem (ou seja, uma lembrança) pedindo para levarmos uma cópia do trabalho para a próxima reunião. O CybreMinder usa informações de contexto para informar o usuário sobre uma determinada lembrança. Uma importante informação de contexto usado pela ferramenta é a localização. A localização pode ser em relação a algum lugar ou em relação à outra pessoa, por exemplo: lembrar de comprar um presente ao chegar a um determinado lugar, também quando estiver próximo de meu colega de trabalho, me avise de comentar sobre a reunião de trabalho.

O sistema foi implementado em Java e é dividido em duas partes principais:

- criar lembranças: o CybreMinder apresenta uma interface semelhante ao de um e-mail, com assunto, corpo da mensagem, uma lista de outras pessoas a quem a lembrança interessa nível de prioridade, data e hora de expiração e a situação (contexto) que a mensagem deve ser entregue, por exemplo, o lugar e a hora;
- entregar lembranças: a lembrança é entregue quando a situação (contexto) associada for satisfeita ou porque o tempo expirou. O CybreMinder decide qual a melhor forma de entregar a mensagem ao usuário. A forma de entrega default é mostrar a lembrança num visor disponível junto com um sinal de áudio, mas isso pode ser configurado pelo usuário para cada lembrança. A lembrança pode ser entregue via e-mail, celular, handheld, entre outros.

CybreMinder também permite fazer lembranças complexas, como por exemplo, Joana precisa fazer uma chamada telefônica para Pedro quando ela chegar ao seu escritório, quando ela tiver tempo livre na sua agenda e seu amigo não estiver ocupado. Para criar esta situação, o usuário precisa criar três situações, são elas: Joana está em seu escritório, o nível de atividade de Pedro está baixo e Joana tem pelo menos uma hora livre antes do próximo compromisso.

Os tipos de contexto percebidos pela ferramenta são: agenda do usuário, ambiente físico e social. Além disso, a determinação de situação é complexa, ou seja, a forma que o CybreMinder utiliza para programar uma situação de contexto em que a lembrança deve ser entregue, não é fácil de ser usada por um usuário comum.

Para perceber o contexto associado com a lembrança, o CybreMinder usa o Contexto Toolkit. O Contexto Toolkit é um software que ajuda a construir aplicações cientes de contexto. Ele promove três principais conceitos para construir aplicações cientes de contexto: separar a aquisição de contexto do uso de contexto; agregação de contexto e interpretação de contexto. Agregação e interpretação facilitam a aplicação a obter o contexto requerido.

De [DEY00a]. descreve o Contexto Toolkit consistindo em três blocos básicos:

- Widget: agregador e interpretador de contexto que pode ser visualizado através da figura 9. Widgets são responsáveis principalmente por coletar e encapsular informação sobre um dado contexto, como localização. Os widgets também dão suporte a serviços que permitem afetar o ambiente, como por exemplo, controlar a intensidade de luz de um local de acordo com a luminosidade detectada pelos sensores;
- agregação: responsável por unir toda informação de contexto de uma entidade particular (pessoa, lugar ou objeto). O interpretador de contexto é usado para abstrair ou interpretar o contexto. Por exemplo, um widget pode fornecer um contexto de localização na forma de latitude e longitude, mas uma aplicação pode requerer a localização como o nome da rua. Na arquitetura, isso é realizado pelos interpretadores de contexto. O mecanismo de comunicação entre os componentes é XML sobre HTTP;
- aplicação: são as aplicações que usufruem dos componentes do Context Toolkit.

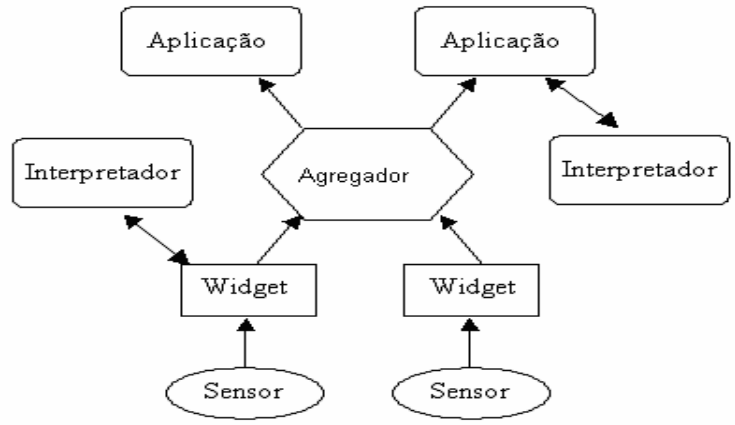

Figura 09: Componentes do Context Toolkit (extraído de [DEY00a]).

3.5 AROUND

A arquitetura AROUND provê uma infra-estrutura de localização de serviços que permite as aplicações selecionarem serviços que são associados a uma determinada área espacial [JOS03]. A principal diferença entre a arquitetura AROUND e as outras aplicações LBS é que ela utiliza dois modelos distintos de seleção de serviços:

- modelo baseado em distância. Neste modelo, o cliente seleciona os serviços localizados dentro de sua região de proximidade, ou seja, um raio é criado ao redor da posição atual do cliente e ele seleciona os serviços de seu interesse que estiverem dentro desse raio. A desvantagem desse

modelo é que quanto maior o raio, mais coisas que não são de interesse do usuário estarão dentro de sua área de proximidade;

- modelo baseado em escopo. Neste modelo, cada serviço tem seu escopo associado a um espaço físico. O cliente seleciona aqueles serviços que o escopo inclui em sua própria localização, isto é, o cliente é capaz de descobrir um serviço se o mesmo estiver localizado dentro do escopo desse serviço. Por exemplo, um servidor de mapas de municípios de um estado que é oferecido apenas naquela região coberta pelos mapas. Na representação da figura 10, um cliente é representado pelo circulo 'C' e está dentro dos escopos em torno do circulo central visualizado na figura 10, portanto, ele pode selecionar os serviços que são oferecidos nesses escopos.

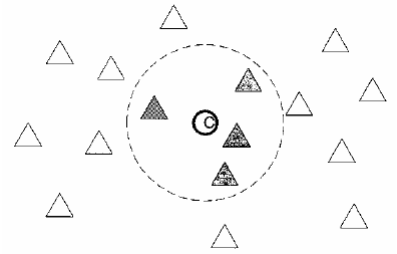

Figura 10: Seleção baseada em distância (Figura extraída de [JOS03]).

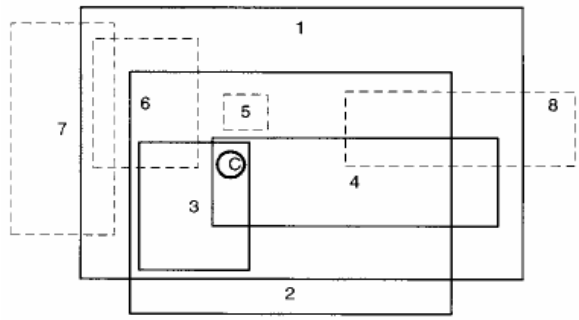

Figura 11: Seleção baseada em escopo (Figura extraída de [JOS03]).

No modelo baseado em distância, o foco está na localização do provedor de serviço, enquanto no modelo baseado em escopo, o foco está na área geográfica definida pelo uso do serviço. Estas diferenças fazem com que cada modelo seja o melhor para um determinado tipo de serviço. O modelo baseado em distância é o mais adequado para serviços que tenham uma forte associação com um específico ponto no espaço, como por exemplo, um restaurante. Por outro lado, o modelo baseado em escopo é o mais adequado para serviços que não têm uma ligação com um ponto específico no espaço, tais como, serviços de mapas e previsão do tempo.

O modelo baseado em escopo, representado pela figura 11, descreve o mecanismo utilizado pela arquitetura AROUND para associar serviços com localização. O escopo de um serviço é

registrado em um conjunto de contexto de localização, escopos são definidos como ambiente, na figura 11 pode ser visualizado 8 ambientes e um controlador central definido pela letra c. Neste caso, contexto de localização é uma representação simbólica de uma área num espaço físico, como por exemplo, um departamento de um campus universitário, um laboratório dentro de um departamento, uma praça pública, um bairro, etc. Na arquitetura AROUND, os contextos de localização podem ser ligados por relacionamentos uni direcionados, formando um grafo, onde o objetivo é aumentar o processo de descoberta de serviços. Relacionamentos entre contextos estabelecem um mecanismo para a propagação de consultas de um contexto fonte para um contexto destino. Na arquitetura, existem dois tipos de relacionamentos:

- contido: se refere à inclusão espacial de áreas dentro da área de outro contexto. A figura 12 apresenta um exemplo em que um serviço "A" registrado no contexto "Campus" está disponível em todo lugar porque todos os outros contextos estão contidos no contexto "Campus". Por outro lado, o serviço "C" está registrado no contexto "Lab. 1", portanto, esse serviço está somente disponível neste contexto;
- adjacente: expressa a proximidade espacial entre dois contextos de localização, por exemplo, os quartos de um edifício, onde cada quarto tem um contexto de localização. Isso permite que usuários consultem serviços próximos mesmo estando fora do escopo.

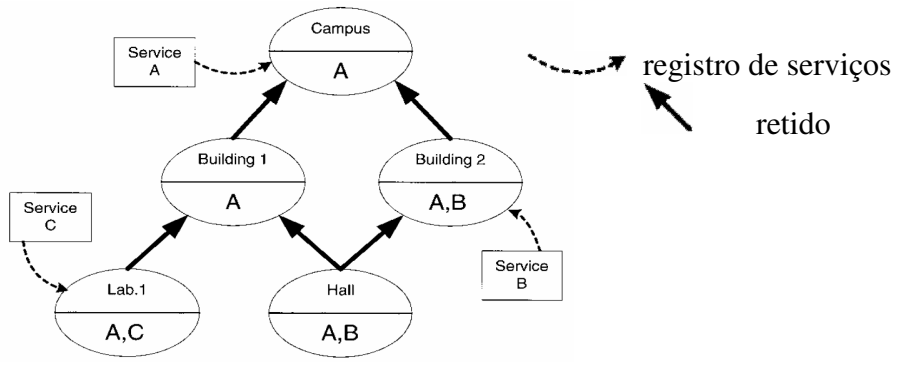

Figura 12: Localização de serviços através de relacionamentos entre os contextos de localização (extraído de [JOS03]).

Quando o usuário, carregando um dispositivo com um cliente AROUND, se move para uma nova área, os serviços que estão registrados nessa área são descobertos e os ícones são mostrados na tela do dispositivo do usuário. Este pode ativar um serviço clicando no ícone. Por exemplo, um serviço de informação de ônibus que informa os ônibus que passam na rua onde o usuário está localizado e os pontos de ônibus mais próximos.

A principal desvantagem da arquitetura AROUND é que ela não leva o contexto do usuário em consideração ao apresentar os serviços. Apenas a localização e o deslocamento do usuário são utilizados pela ferramenta para descobrir em que área de serviço ele está. A comunicação entre os componentes é baseada em CORBA [JOS03].

3.6 Online Aalborg Guide

Online Aalborg Guide [AND03] é um protótipo construído usando o framework baseado em LBS desenvolvido no departamento de ciências da computação da Universidade de Aalborg. O framework é usado para implementar um guia on-line para turistas que visitam a cidade de Aalborg. As características básicas da ferramenta são:

1. Visualizar a localização dos PoI (Point of Interest) mais próximos;
2. Permite ao usuário salvar os PoI para uso posterior, como o bookmark de um Web browser;
3. Permite ao usuário adicionar novos PoI, submetendo o nome e a descrição do mesmo;
4. O usuário pode adicionar novos comentários e fotos a um PoI já existente;
5. Os provedores podem enviar anúncios de acordo com as preferências e localização do usuário;
6. Permite ao usuário encontrar o menor caminho para um PoI;
7. Obter informações sobre outros usuários;
8. O usuário pode editar o seu perfil antes e durante uma viagem;
9. Serviço de mapas. Um mapa do ambiente é exibido a toda hora com uma indicação da posição do usuário e os PoI mais próximos.

O Online Aalborg Guide usa uma mistura de tecnologia push e pull, como se pode perceber pelas características anteriores. Por exemplo, os PoI mais próximos são continuamente atualizados e mostrados na tela do celular. Dessa forma, o usuário não precisa interagir com o dispositivo para ver que PoI estão próximos. Porém, se o usuário deseja obter mais informações sobre um PoI, ele deve fazer uma requisição ao servidor e a informação é apresentada na tela do dispositivo. O protótipo desenvolvido utiliza o celular Nokia 7650 conectado via Bluetooth a um Emtac GPS. O programa que é instalado no celular é chamado de GPSOne. Como se pode perceber na figura 13, o Emtac GPS constantemente provê coordenadas ao celular. Quando o celular recebe as coordenadas, o GPSOne começa a prover serviços LBS para o usuário. O celular se conecta ao servidor através do protocolo HTTP. Se o GPSOne precisa fazer o download de algum mapa raster, uma conexão ao

servidor de mapas é estabelecida e o mapa é retornado no formato JPEG. O protótipo implementado contém apenas as características que foram listadas anteriormente.

Mapas *rasters* são os dados mais estáticos na aplicação LBS. Porém, esses mapas demandam uma maior capacidade de armazenamento. Por causa do pequeno poder de armazenamento do terminal do cliente, não é possível carregar todos os mapas. No protótipo, um cachê de mapas tem sido implementado para carregar unicamente os mapas necessários dependendo da localização do usuário. Somente os mapas ao redor do usuário são baixados do servidor. Depois de certo tempo, a memória do dispositivo do usuário estará cheia devido à quantidade de mapas que foram baixados. Uma estratégia adotada para resolver essa situação é apagar os mapas menos recentemente usados.

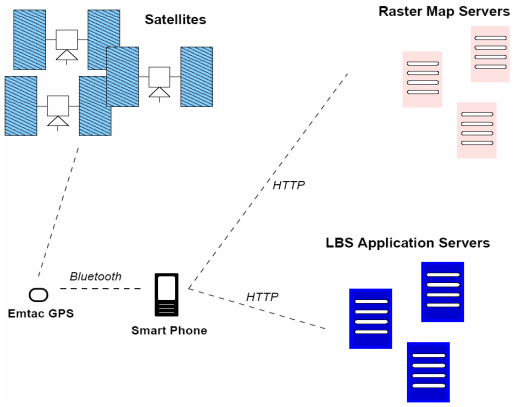

Figura 13: Arquitetura do Online Aalborg Guide (extraído de [AND03]).

3.7 Flame2008

Flame2008 [WEI04] é um protótipo de uma plataforma de integração para Web services inteligentes personalizados para as olimpíadas de 2008 em Beijing. A principal idéia desse projeto é, através de um dispositivo móvel, realizar 'push' de ofertas significantes baseadas na situação atual e no perfil do usuário. Todas as dimensões do contexto (localização, calendário, perfil, etc.) são representadas através de ontologias. Com agregação dessas dimensões, o Flame2008 define situações que são registradas no sistema. Por exemplo, através de uma notação em F-Logic [WEI04], é possível registrar a seguinte situação:

WatchingCompetition : BeingAtEvent [position ->> loc#Stadium;
localAction ->> act#AnyAction; userState ->> cal#Leisure].

Este exemplo descreve a situação "assistindo a uma competição". O usuário se encontra nessa situação se estiver localizado no estádio executando qualquer ação e na sua agenda aquele horário está marcado como hora de lazer. O Flame2008 pode ser composto de várias ontologias,

como mostrado no exemplo anterior, pois, ‘loc’, ‘act’ e ‘cal’ são *namespaces* para diferentes ontologias. Definida a situação em que o usuário se encontra e o seu perfil, o sistema se encarrega de buscar serviços que se encaixam nesses parâmetros. O perfil é composto por interesses, preferências e dados pessoais do usuário. Interesses são informações estáticas que não se alteram com a mudança de contexto, por exemplo, se o usuário possui interesse por música clássica e obras de arte. As preferências podem depender da situação, por exemplo, um usuário em sua cidade pode preferir comida italiana quando vai a um restaurante, mas quando ele está viajando pode preferir experimentar a comida local. Cada usuário pode manter seu perfil através de um conjunto de propriedades que o caracterizam. Além disso, há sensores, que obtêm informações do ambiente atual do usuário (localização e tempo). Os resultados são instâncias de uma ontologia de alto nível que são usadas para implicitamente construir um “perfil de situação”. Ele é semanticamente comparado com os perfis de todas as situações conhecidas pelo sistema, e implicitamente comparada com todos os perfis de serviços registrados. A desvantagem é que ele não trata o relacionamento com outros usuários, ou seja, o sistema não monitora o contexto de contatos do cliente. Além disso, a busca por produtos e serviços é baseada apenas na categoria do produto e no perfil do usuário, não se importando com outras características do produto ou propriedades do serviço.

3.8 Nexus

Nexus [DUR04] é uma plataforma com o propósito de dar suporte a todos os tipos de aplicações cientes de contexto através de um modelo de contexto global. Servidores de contexto podem ser integrados à plataforma para compartilhar e usufruir das informações providas pelos outros servidores de contexto. Esses servidores de contexto são chamados de modelos locais (ou modelos de contexto) [KAV04]. O modelo local contém diferentes tipos de informações do contexto. Por exemplo, um modelo que representa as estradas, um modelo que representa as casas em uma cidade, um modelo que representa as pessoas, entre outros. No caso do modelo de pessoas, são usados sensores para manter os dados atualizados no modelo.

A plataforma Nexus é organizada como mostra a figura 14:

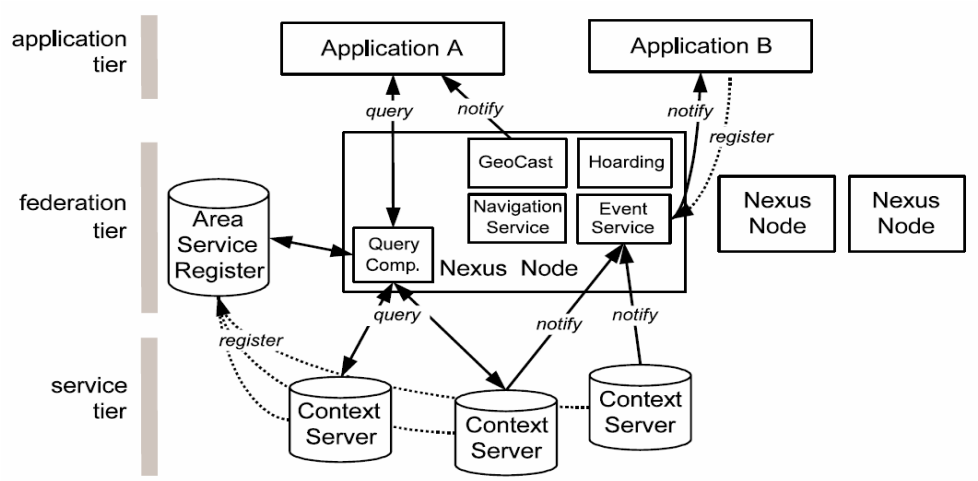

Figura 14: Plataforma Nexus (extraído de [DUR04]).

Os servidores de contexto (context servers) presentes na camada de serviço (service tier) armazenam os modelos locais. Para ser integrada a plataforma, os serviços devem seguir certa interface descrita em XML e registrada em um serviço chamado Área *Service Register* (ASR) para que possam ser descobertos dinamicamente. A camada de federação (federation tier) funciona como um mediador entre as aplicações e os servidores de contexto. Ele possui a mesma interface dos servidores de contexto, mas não armazenam modelos. Ele analisa a consulta da aplicação, determina os servidores de contexto que podem responder a consulta e distribui a consulta para esses serviços. O nodo Nexus também possui a capacidade de adicionar serviços que possuem suas próprias interfaces e que usam os modelos de contexto para processar informação e fornecê-la para as aplicações. A figura mostra quatro tipos de serviços da plataforma Nexus:

- serviço de evento (event service) monitora eventos espaciais e permite o processamento de predicados espaciais, tais como: “monitorar quando eu estiver próximo de dois amigos”;
- serviço de navegação (navigation service) calcula a rota usando os dados disponíveis nos modelos locais;
- geocast é responsável por enviar mensagens para todos os usuários em uma determinada área geográfica;
- serviço Hoarding é responsável pelo processo de desconexão da aplicação de uma área de comunicação, ou seja, se uma aplicação está prestes a sair de uma área de comunicação de rede sem fio.

No topo da arquitetura estão as aplicações cientes de contexto que podem usar a plataforma de três formas diferentes:

1) aplicações podem enviar consultas e obter informações sobre o ambiente ao redor, como por exemplo, PoI e amigos próximos, etc.
2) aplicações podem registrar um evento para receber uma notificação quando certo estado do mundo ocorrer;
3) aplicações podem usar os serviços do nodo Nexus, como os serviços de navegação, para enriquecer as funcionalidades da aplicação.

A plataforma Nexus possui uma arquitetura semelhante ao padrão Web service [ACK04]. Os servidores de contexto funcionam como provedores de serviços que são registrados em um serviço de diretório. O ASR funciona como um serviço de diretórios semelhante ao UDDI no padrão Web service. Porém, na plataforma Nexus, a descoberta de serviços é realizada pelo nodo Nexus na camada de federação e não pelas aplicações como no padrão Web service. A plataforma Nexus monitora o espaço físico, ou seja, o contexto do ambiente. O Nexus é capaz de transmitir anúncios para vários usuários em uma determinada área, mas não faz busca automática de produtos ou serviços de interesse do usuário. Além disso, a plataforma não é capaz de deduzir informação a partir dos dados do contexto.

3.9 ICAMS

ICAMS [NAK04]. É um sistema cliente-servidor que usa celulares para tornar a comunicação mais eficiente através de informações de localização e agenda dos usuários. O sistema usa o serviço de localização PHS (Personal Handphone System) oferecido pela companhia de telecomunicações japonesa NTT Do CoMo. O celular atualiza a sua localização a cada 15 minutos e tem uma precisão de 100 metros. Basicamente, o ICAMS auxilia no redirecionamento de mensagens telefônicas ou e-mail. Os usuários do sistema ICAMS são amigos que desejam compartilhar localização e outras informações. Quando um usuário acessa o servidor, um script PHP (*Hypertext Preprocessor*) consulta o banco de dados e retorna um arquivo CHTML chamado Top Page. Este arquivo apresenta os outros usuários do sistema (os amigos) em ordem de proximidade. Ícones e setas são usados para indicar se um amigo está se movendo e em que direção em relação ao usuário. Quando o usuário clica no nome de um amigo no Top Page, um segundo PHP consulta o banco de dados e retorna um CHTML chamado *MemberPage*. Essa página contém mais detalhes sobre aquele amigo: nome, o último local em que esteve, se o amigo está parado ou se movendo, à distância em relação ao usuário e as opções de contato (telefones, e-mails e outros). O usuário pode clicar no número de telefone ou no e-mail para estabelecer uma comunicação. Se o

amigo estiver em sua casa, por exemplo, o telefone residencial é listado primeiro no *MemberPage*. Mas se o amigo estiver numa reunião no trabalho, o seu e-mail de escritório será o primeiro na lista. Através do Web browser do celular, o usuário pode entrar com sua programação (agenda) selecionando o dia, hora e conteúdo da programação. Em seguida, o usuário pode ordenar os contatos para a programação criada em ordem de preferência. Dessa forma, seus amigos saberão qual a melhor forma de entrar em contato para cada situação do dia. Esse sistema possui algumas desvantagens:

- baixa precisão em relação à localização;
- informações do contexto que são analisadas pelo são localização do usuário e sua agenda;
- o redirecionamento das mensagens não é automático.

3.10 FieldMap

FieldMap [FIL07] é uma ferramenta escrita em Java para mostrar mapas e permitir anexar comentários. Ela foi construída para ser usada em PDA's, desktops e laptops. O usuário pode adicionar novos pontos de interesse, desenhar no mapa, marcar com pontos e polígonos e adicionar comentários escritos ou por voz para ajudar a recordar sobre informações do ambiente. Pode-se perceber que esta ferramenta não faz qualquer análise do contexto do usuário. Apenas a localização e o caminho percorrido pelo usuário são mostrados no mapa para que ele possa se orientar. Essa ferramenta é mais utilizada na área de pesquisa na qual o ambiente está sendo estudado, como, por exemplo, na área de arqueologia.

3.11 AMS

O AMS (Arrhythmia Monitoring System) [LIS04] é um trabalho na área de tele medicina com o objetivo de prever ataques cardíacos conhecidos como arritmia. O AMS coleta sinas de um ECG (eletrocardiograma) combinado com os dados de um GPS e os transmitem a uma estação remota para visualização e monitoração. A arquitetura do sistema é composta dos seguintes componentes:

- wearable sever: é um pequeno coletor de dados onde o paciente fica conectado. Ele é composto por um ECG (eletrocardiograma) que coleta as atividades elétricas do músculo cardíaco através de biosensores;

- central server: é um pequeno servidor localizado próximo ao cliente. Ele executa várias funções: compressão, tratar sinais do GPS e detectar sinais de arritmia rudimentares. O central server é geralmente um dispositivo móvel como um Palm, que conectado ao wearable server e a um GPS, transmite esses sinais para o call center;
- call center: é através do qual um profissional médico monitora o sinal ECG e transmite um alerta caso detecte alguma situação de risco.

Os dados dos biosensores são armazenados em um buffer e a cada 20 segundos, esse buffer é transmitido para o call center. O sistema também possui um botão de "pânico" pelo qual o cliente pode enviar um alerta para o sistema central que se encarrega de chamar um serviço de emergência (190, por exemplo) passando a localização do usuário.

3.12 Comparação entre Arquiteturas

As ferramentas estudadas neste capitulo foram comparadas conforme as características funcionais de um sistema pervasivo capaz de analisar o contexto, tanto do usuário quanto do ambiente. A escolha utilizada para comparar as principais características entre ferramentas basearam-se em autores citados nos trabalhos descritos no capitulo 3 são elas:

1) Monitoração em tempo de execução: para perceber alterações no contexto, é indispensável que as informações obtidas dos sensores sejam as mais atuais possíveis;
2) Monitoração de vários tipos de contexto: as ferramentas foram comparadas conforme a classificação do contexto descrita na seção 2.5.4;
3) Capaz de monitorar o contexto de terceiros: além de monitorar o próprio contexto, um sistema ciente de contexto deve ser capaz de perceber o contexto de terceiros (parentes e amigos);
4) Sistema SIG onipresente: essa característica permite que o usuário visualize a sua localização, a localização de seus contatos e os PoI através de um mapa digital disponível em seu dispositivo móvel;
5) Análise do perfil do usuário para o fornecimento de mapas personalizados: através da análise do perfil do usuário, informações sobre suas preferências podem ser extraídas para construir um mapa com os PoI que estejam de acordo com esse perfil;
6) Anúncio de produtos ou serviços: esta característica faz parte do contexto do ambiente e permite com que o usuário receba anúncios de produtos ou serviços que estejam próximos de sua localização.

7) Orientado a serviço: no âmbito desse trabalho, essa característica não é tão importante quanto às anteriores, porém ela permite que novos serviços heterogêneos sejam adicionados a arquitetura de forma padronizada, acrescentando mais funcionalidades ao sistema.

Tabela 05: Comparativo entre os sistemas analisados.

características / Sistemas	Monitora em tempo execução	Monitoração de vários tipos de contexto	Capaz de monitorar contexto de terceiros	Sistema SIG onipresente	Análise perfil do usuário para o fornecimento de dados personalizados	Anúncio de produtos ou serviços	Orientado a serviço	Replicação de dados	Contexto Local	Contexto Global
SOCAM	Sim	Baseado em ontologia	Sim			sim			Sim	sim
MIXIS	Sim	Baseado em web semântica		sim			Sim	Sim		sim
CYBRE MINDER	Sim	Sim	Sim		Sim				Sim	
AROUND	Sim			Sim			sim		Sim	Sim
ONLINE AALBOARG GUIDE	Sim	Apenas localização e perfil		Sim	Sim		sim	Sim		
FLAME 2008		Apenas localização e perfil		Sim	Sim	Apenas o tipo				
NEXUS	Sim	Apenas localização e perfil	Sim	Sim					Sim	Sim
ICAMS		Apenas localização e perfil		Sim				Sim	100mts	
AMS	Sim	Apenas localização e perfil		Sim						
INFLAWARE		Acesso a serviços	Sim			sim	Sim		Sim	
Modelo Proposto	Trabalhos Futuros	sim	Sim	Trabalho futuro	Sim	Sim	Sim	Modelo cliente servidor	Sim	Sim

A tabela 5 apresenta um estudo comparativo entre as ferramentas que possuem as características citadas anteriormente. Os campos marcados com 'SIM' indicam que a ferramenta possui a determinada funcionalidade. Os campos sem marcação indicam que a ferramenta não possui a funcionalidade especificada. Quando alguma ferramenta possui alguma funcionalidade, mas com restrições, essas restrições são descritas no campo correspondente na tabela, segundo [LIM07], estas são funcionalidades fundamentais que uma plataforma pervasiva deva possuir e algumas dessas características também serão utilizadas no modelo proposto.

3.13 Considerações Finais

Neste capítulo, foram descritos alguns sistemas cientes de contexto, como funcionam, suas principais vantagens e desvantagens. Em seguida, uma tabela comparativa mostra quais os sistemas apresentados possuem as funcionalidades desejáveis para um sistema ciente de contexto capaz de monitorar o contexto dos usuários e realizar pesquisas por produtos dinamicamente.

4 MODELO DE ARQUITETURA UTILIZADO

O modelo de descoberta utilizado neste trabalho tem por objetivo atender a plataforma alvo em um ambiente pervasivo, disponibilizando quais serviços estarão disponíveis no dispositivo do usuário. Esse modelo é voltado para aplicações genéricas, podendo ser utilizado em diferentes cenários, tais como: bibliotecas onde os usuários poderiam efetuar consultas à base de dados locais sobre os livros disponíveis e localizar em que estante se encontra, através de imagens visualizadas. CTI de hospitais, detecção de aluno em campus universitário, entre outras adaptações e utilizações em plataformas pervasivas de uso genérico.

4.1 Plataforma alvo

O modelo de descoberta de serviço proposto nesse trabalho visa fornecer o desenvolvimento de um gerenciador de serviços a uma plataforma existente [RIE06]. Esse modelo auxiliará no funcionamento do projeto final da plataforma pervasiva. Essa plataforma [RIE06] tem como objetivo integrar dispositivos eletro-eletrônicos em um ambiente pervasivo fornecendo os serviços existentes na rede pervasiva para os dispositivos integrados, além de disponibilizar os serviços desses dispositivos para a rede. Além disso, essa plataforma deve fornecer os serviços e recursos necessários para a computação pervasiva, adapta qualquer dispositivo eletro eletrônico a um ambiente pervasivo, onde essa plataforma é capaz de disponibilizar os serviços e recursos desse dispositivo para todos os outros dispositivos e servidores conectados a rede pervasiva. A plataforma HW/SW é composta por quatro camadas, conforme a figura 15: camada física, camada SO, middleware e camada de aplicação.

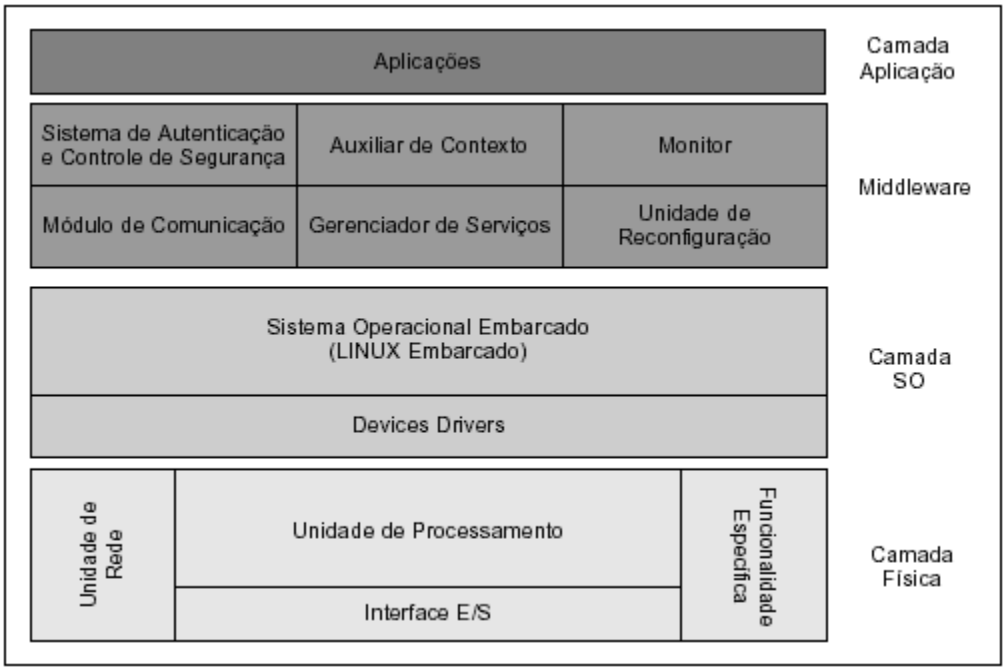

Figura 15: Estrutura da plataforma em camadas [RIE06].

A camada física visa o tratamento dos dispositivos físicos e é implementada em HW. Essa camada é dividida em quatro componentes:

1. Unidade de Rede, utilizada para conectar a plataforma na rede pervasiva;
2. Unidade de Funcionalidade Específica, utilizada para auxiliar e controlar a funcionalidade eletroeletrônica do dispositivo;
3. Interface E/S, utilizada para a comunicação e interface com os usuários, ambientes e dispositivos;
4. Unidade de Processamento, unidade responsável gerenciamento e execução das aplicações e das outras unidades.

A camada SO tem como objetivo tratar as funcionalidades de um sistema operacional para o sistema embarcado. É dividida em duas subcamadas, são elas, subcamada de Device Drivers que gerência os componentes da camada física e subcamada do Sistema Operacional Embarcado que gerência as tarefas da aplicação que executa no dispositivo e determina os serviços disponibilizados pelo mesmo, além de tratar as limitações do sistema.

O Middleware constitui um conjunto de componentes necessários para auxiliar a integração do dispositivo com a rede pervasiva e disponibilizar os serviços necessários que compõem o middleware da arquitetura. Essa camada é formada por seis componentes básicos: Módulo de Comunicação, que integra o dispositivo na rede e gerência a comunicação do dispositivo com outros dispositivos; Gerenciador de Serviços que fornece gerenciamento dos serviços e recursos

desse dispositivo para a aplicação pervasiva; Unidade de Reconfiguração que auxilia a arquitetura SW na adaptação de novos componentes SW; Sistema de Autenticação e Controle de Segurança que ajuda na segurança do ambiente, fornecendo serviços de autenticação; Auxiliar de Contexto que auxilia na detecção de contexto a arquitetura; Monitor que fornece o monitoramento do dispositivo para a aplicação, informando erros e problemas. A camada Aplicação é um fragmento das aplicações que executam no ambiente específica para o dispositivo embarcado. Um exemplo de aplicação seria disponibilizar os serviços de uma cafeteira para o ambiente pervasivo [RIE06].

4.2 Modelo proposto

O modelo proposto divide-se em dois níveis. O nível interface consiste num protocolo de descrição de recursos/serviços, usando XML, através do qual algumas aplicações relacionadas ao contexto foram descritas na arquitetura desenvolvida por [RIE06]. Em uma primeira análise o sistema solicitava login e senha do usuário, esse login e senha funcionariam como um arquivo cliente instalado no dispositivo do usuário, contendo também o ID do usuário. Através do ID o sistema terá acesso ao cadastro "descrição" do dispositivo e quais recursos desse dispositivo estarão disponíveis para os demais usuários da rede pervasiva. Após o login no sistema, a plataforma existente terá a informação sobre a localização dos usuários e seus dispositivos com os recursos/serviços disponíveis, através do registro das informações do ambiente onde foi efetuado. Entretanto, segundo [GEY08] o acesso a dispositivos pervasivos através de mecanismos de login e senha, além de ter uma considerável perda de desempenho, devido ao fato da conexão e desconexão constante ao sistema, faz surgir à necessidade de interação maior do usuário. Ficando fora do escopo da computação pervasiva, onde o acesso tende a ser imperceptível ao usuário segundo [WEI91].

4.3 Movimento dos Componentes da Aplicação

A figura 16 tem por objetivo mostrar um exemplo do funcionamento de fluxo de dados entre os diferentes controles de serviços. Onde primeiramente o dispositivo é detectado pelo sensor da plataforma existente, gerando um fluxo de dados que é detectado pela descoberta de serviço (DS).

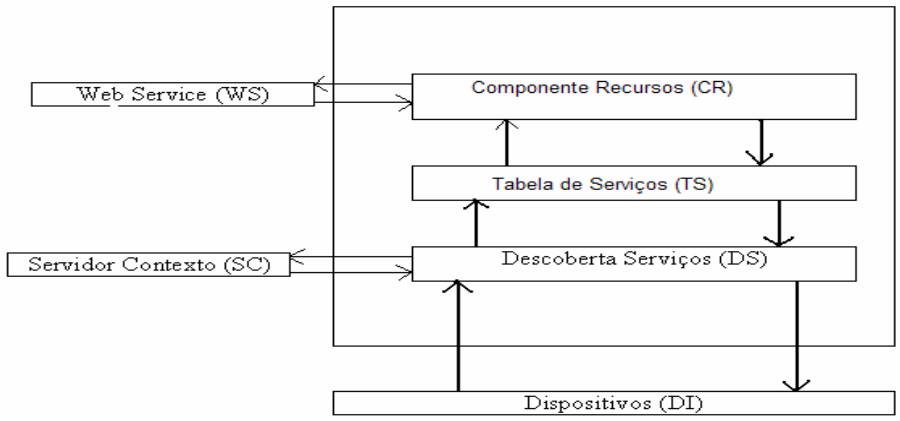

Figura 16: Modelo Gerenciador de descoberta de Serviços Pervasivos.

(DI) Dispositivos: É realizada a detecção do usuário através de seu dispositivo cadastrados no sistema, diferentes dispositivos, podem ser cadastrados para um mesmo usuário;

(DS) Descoberta Serviços: Onde é realizada a verificação de dispositivos, repassando as informações ao servidor de contexto (SC), que por sua vez verifica os serviços e usuários cadastrados no dispositivo informado pelo (DI). Feito isso, os dados são reenviados ao modulo descoberta serviços (DS), que os transmite ao módulo tabela serviços (TS), disponibilizando localmente todos os serviços disponíveis pelo dispositivo no ambiente local e atualizados aos demais usuários do sistema, essa atualização ficará disponível no servidor de contexto (SC);

(SC) Servidor de Contexto: Onde serão armazenados informações contextuais do usuário e seus dispositivos, bem como os serviços relacionados em todos os dispositivos. Este servidor de contexto funciona em escala global, recebendo e enviando informações contextual aos módulos descoberta serviços (DS) que estarão atuando localmente;

(TS) Tabela serviços: Onde são armazenados localmente os serviços e recursos de dispositivos de cada usuário, estas informações estão disponíveis localmente. Também será responsável pelo controle de serviços que estão em uso, havendo possibilidade de implementar em trabalhos futuros um controlador tempo e uso de serviços. O usuário poderá verificar quanto tempo ainda resta pra determinado serviço estar disponível, ou ainda verificar se determinado serviço ocioso, ou está sendo utilizado, determinar tempo de ociosidade e disponibilizá-lo ao sistema para utilização de outros usuários;

(CR) Componente de recursos: Gerencia os serviços disponíveis localmente e os transmite. Caso haja alguma solicitação externa a outro ambiente através (WS) web services, como por exemplo: a localização do usuário.
(WS) Web Services: Este componente é responsável por prover comunicação e troca de dados e informações entre diferentes redes e formas de comunicação com a rede pervasiva local. O desenvolvimento desse componente será tema de trabalhos futuros.

4.4 Funcionamento do Sistema Proposto

O módulo (DS) Recebe informações do dispositivo (DI), consulta à tabela de dados local (TS) para verificação do cadastro do dispositivo. Se o dispositivo é cadastrado no sistema, essas informações são processadas e enviadas ao servidor de contexto (SC), no servidor de contexto é realizada uma verificação dos serviços disponíveis pelo dispositivo em questão e também é feita a atualizado do status no sistema, sobre quais dispositivos e serviços estão disponíveis. Após essa atualização, o servidor de contexto retransmite para módulo de descoberta serviços às informações para serem enviadas ao modulo tabela serviços (TS), a fim de que as informações estiverem disponibilizadas localmente, e também é enviado um aceite ao dispositivo solicitante, disponibilizando o seu acesso ao sistema.

4.5 Modelagem e Implementação do Modelo Proposto

No modelo proposto, o gerenciador de recursos possui diversas tarefas para realizar e é responsável tanto pela requisição dos serviços quanto para o atendimento de uma solicitação. Para a requisição dos serviços, esse componente ainda deve fazer a busca e/ou descoberta de serviços e a solicitação do serviço. Já para o atendimento de uma solicitação externa, o gerenciador de serviços fica aguardando por uma solicitação. Ao receber a solicitação do recurso, ele passa a solicitação para o módulo dispositivo que busca a localização do recurso e o local com a descrição do mesmo. Também será realizados controle com o histórico de utilização do recurso, e principalmente o controle do tempo de utilização através da classe histórico do recurso com hora inicial e hora final de uso do recurso solicitado, bem como o controle da sua localização. A figura 17 apresenta o diagrama de classes do gerenciador de serviços.

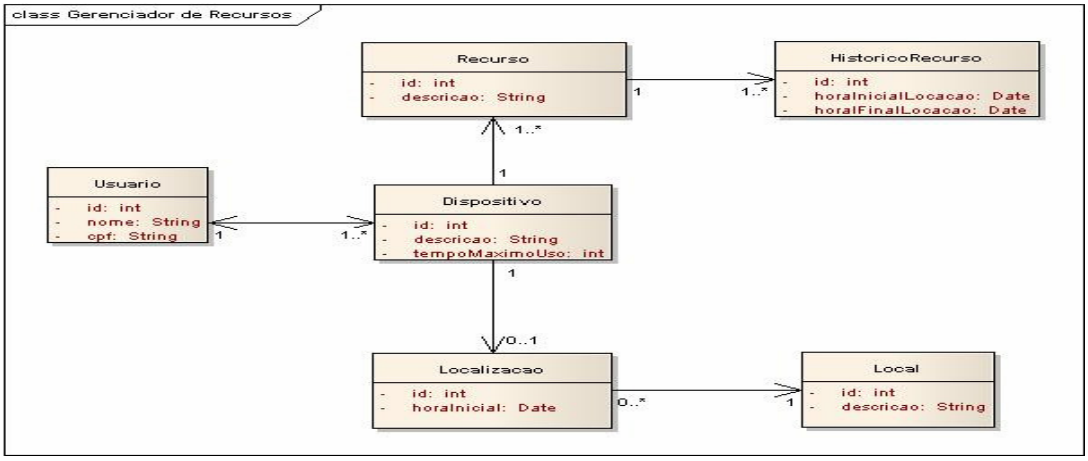

Figura 17: Diagrama de Classes do Gerenciador de Serviços.

A classe Usuário faz a solicitação de um pedido de registro do dispositivo no ambiente pervasivo. A classe dispositivo é responsável por solicitar e buscar a localização com a descrição e o tempo de utilização do dispositivo no sistema. A classe Localização possui os dados da localização do recurso e informações referente ao tempo de uso para cada recurso solicitado que é controlado através da classe Local. A classe local armazena informações sobre a localização do recurso solicitado. A classe Recurso possui a descrição dos recursos de cada dispositivo e por fim, a classe histórico recurso, que possui a finalidade de armazenar e controlar o tempo e localização dos recursos. Pode-se visualizar na figura 18 o diagrama de classes de acesso ao sistema, quando o usuário adentrar no ambiente seu login é feito pelo sistema através do cadastro do dispositivo que o mesmo possui. A localização do usuário é registrada no local onde foi efetuado o login registro do usuário, caso o usuário troque de localização o sistema efetuará o registro de saída e entrada do ambiente, o fluxo de seqüência dessas informações foram descritos na seção 4.6.

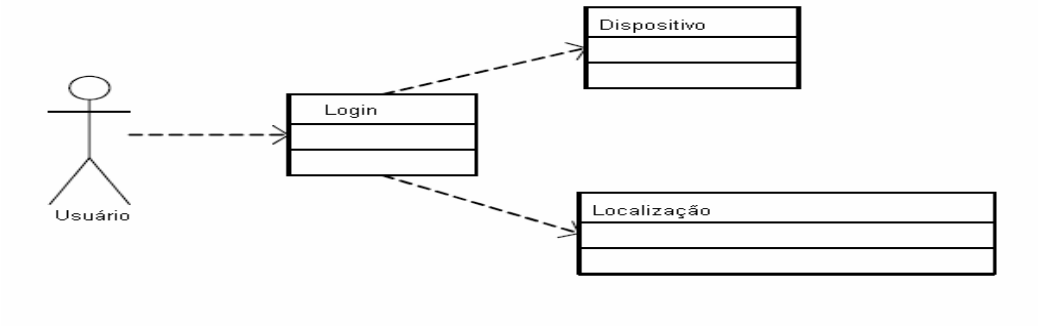

Figura 18: Diagrama de classes de acesso ao sistema

4.6 Modelo de Sequência

Na figura 19 é exemplificado o fluxo de dados relativos à seqüência em que as informações se relacionam de acordo com o proposto.

- Usuário chega ao ambiente;
- O sistema detecta a presença do usuário no ambiente local;
- As informações são repassadas para a plataforma pervasiva, que por sua vez; solicita informações sobre o dispositivo cadastrado no sistema;
- As informações são repassadas e processadas para o servidor de contexto;
- As informações contextuais são retornadas para a plataforma;
- De acordo com as informações de contexto a plataforma monta e armazena as informações do serviço disponível localmente.

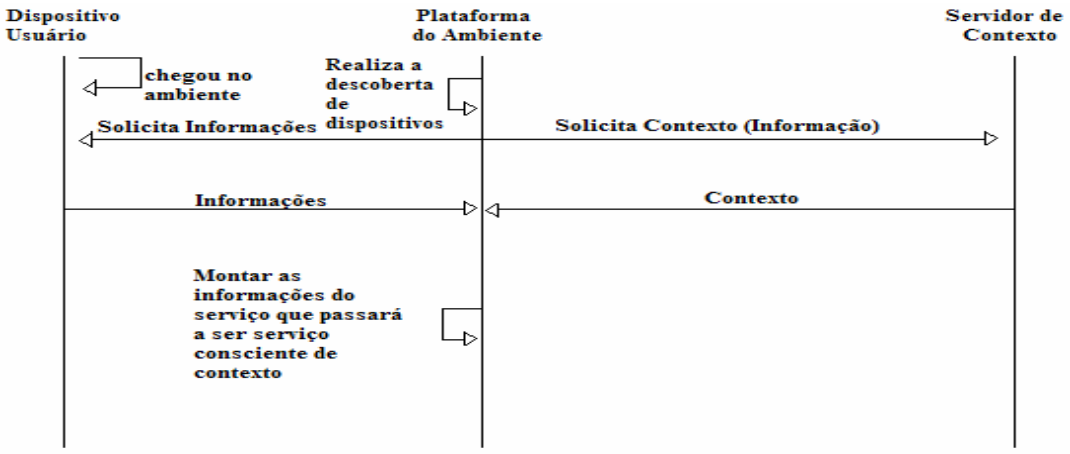

Figura 19: Modelo de Seqüência

A montagem descrita no ultimo fluxo de dados gerados pelo modelo de seqüência é realizada localmente, podendo ser utilizado o modelo descrito por [RIE06], ou um pc com o sistema instalado. No caso outro usuário entrar no mesmo ambiente local, exemplo, em uma mesma sala n usuários o tratamento é realizado pela plataforma do ambiente local, exceto quando for solicitado informações de recursos que não estão disponíveis localmente. Neste caso a plataforma do ambiente local gerara um fluxo de solicitação de informações ao servidor de contexto, o mesmo retorna as informações contextuais do sistema.

5 APLICAÇÃO E VALIDAÇÃO DO MODELO PROPOSTO

Neste capitulo será descrita a aplicação, as telas geradas pelo sistema e os logs gerados na validação e funcionamento do sistema.

5.1 Descrição da Aplicação

O modelo proposto é uma aplicação distribuída que necessita de muitos recursos para atender as suas funcionalidades e suas finalidades. Uma das finalidades dessa aplicação é atender inúmeros usuários, permitindo o cadastro de seus dispositivos e seus recursos. A segunda finalidade é permitir alta interatividade com esses usuários, disponibilizando diversos recursos para fazer o cadastro e a visualização de suas tarefas. Essa finalidade tem como proposta atender o requisito onipresença. A terceira funcionalidade do modelo proposto é compartilhar recursos com outros usuários. E por fim, a última finalidade é fornecer o rastreamento e a localização do usuário, para auxiliá-lo na descoberta de recursos presentes no mesmo ambiente. Também poder buscar através do gerenciador de recursos a disponibilidade em outros ambientes, baseado nessas funcionalidades será apresentado um cenário para essa aplicação nas seções posteriores.

5.1.1 Cenário de Aplicação

Esse cenário mostra a seguinte situação envolvendo a computação pervasiva no sétimo andar do prédio 32 “Informática da PUCRS” onde há os gabinetes de professores, baias de estudantes, laboratórios de pesquisas GAPH e o GSE que são locais que estão no mesmo andar são todos ambientes distintos. Podendo ocorrer à seguinte situação, baseando-se no modelo proposto e descrito na seção 4.2, que pode ser visualizado na figura 16 da mesma seção:

Entrada no Ambiente. Ao entrar em um dos ambientes citados, o aluno será identificado pelo sistema através do modulo (DI), descrito na seção 4.2;

Usuário. O usuário poderá ter um computador como dispositivo de uso cadastrado no ambiente local, e estar utilizando, disponibilizando, ou apenas possuindo vários outros dispositivos no mesmo ambiente, módulos (DS), (SC) e (TS);

Detecção Usuário. Após a entrada do usuário no ambiente o sistema efetuará a detecção do dispositivo cadastrado e os recursos relacionados ao mesmo, esse tratamento é realizado pelo módulo (DS), descrito na seção 4.2;

Mudança de Acesso. Caso esse usuário resolva trocar seu dispositivo com um colega, além das características do equipamento poderia haver mudanças também no tipo ou quantidade de recursos e serviços disponibilizados, surgindo assim à necessidade de criação e gerenciamento de acesso no sistema para realizar essas alterações. Esse tratamento é realizado pelo módulo (SC), descrito na seção 4.2. Embora, tais funções de cadastro de usuários e dispositivos competem somente ao administrador do sistema;

Disponibilidade. O usuário poderá através de o sistema verificar e utilizar os serviços disponíveis no mesmo ambiente, como exemplo, um recurso tocar som pode estar disponível no dispositivo no mesmo ambiente, essa funcionalidade se adaptaria perfeitamente ao modelo proposto por [RIE06]. O modelo proposto pode adaptar-se a essa aplicação utilizando os módulos descritos na seção 4.2. Inclusive com a utilização do módulo WS em caso de utilização de ambientes diferentes, através da disponibilização de serviços de tocar um som de aviso no ambiente onde o usuário se encontrar, devido à descoberta de serviços e recursos proposta por esse trabalho e tratada através do modelo descrito na seção 4.2.

5.1.2 Telas do Modelo Proposto

Nesta seção serão apresentadas as telas geradas pelo sistema na ocasião da simulação e validação da aplicação. Podem-se observar telas relativas ao login e senha, que tem por finalidade o acesso do administrador do sistema para eventuais modificações ou alterações que podem ser feitas. Após, é apresentado à figura 22 com o banco de dados vazio. Esse banco de dados segue o modelo proposto na seção 4.2 representado pelo módulo (SC). Entre outras telas de cadastro de usuários, dispositivos e recursos. E por fim, as telas com as validações geradas pelo sistema proposto.

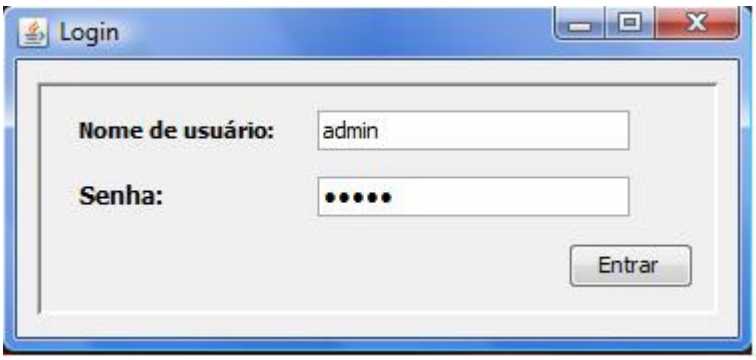

Figura 20: Login de acesso ao sistema.

A figura 20 mostra a tela de login, tendo a finalidade apenas de entrada no sistema por um usuário que terá privilégios de administrador para o tratamento de eventuais contratempos que poderão ocorrer, por exemplo, um usuário poderá ter um dispositivo com vários tipos de serviços. Entretanto, poderá optar em disponibilizar apenas alguns ou ter a necessidade de aumentar a quantidade de tempo disponibilizado para tal recurso ou serviço.

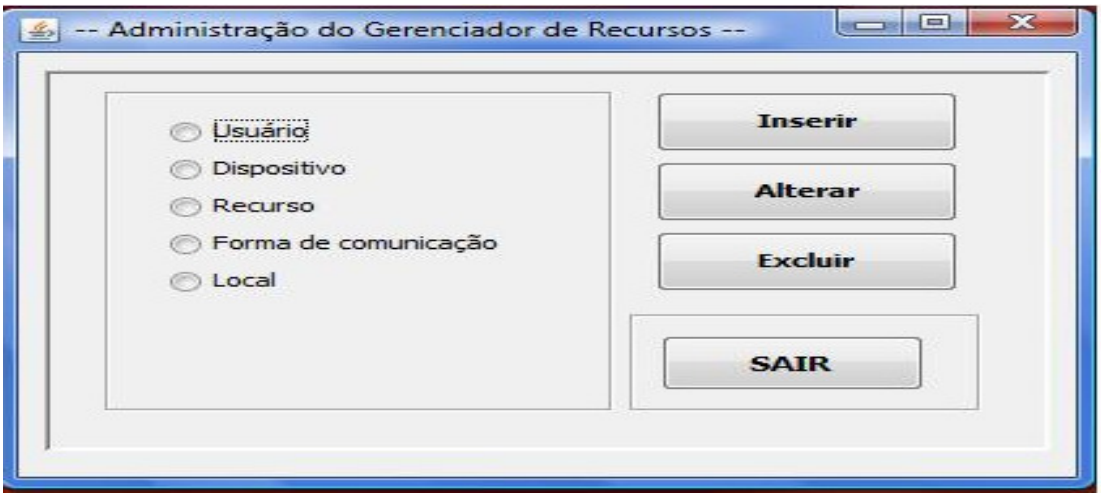

Figura 21: Administração do Gerenciador de Recursos.

Após a entrada no sistema pelo administrador, é disponibilizado o acesso a várias informações que podem ser visualizadas na figura 21. Nesta tela o administrador poderá incluir, alterar e excluir um usuário, dispositivo, recurso, forma de comunicação ou local onde determinado dispositivo está sendo disponibilizado.

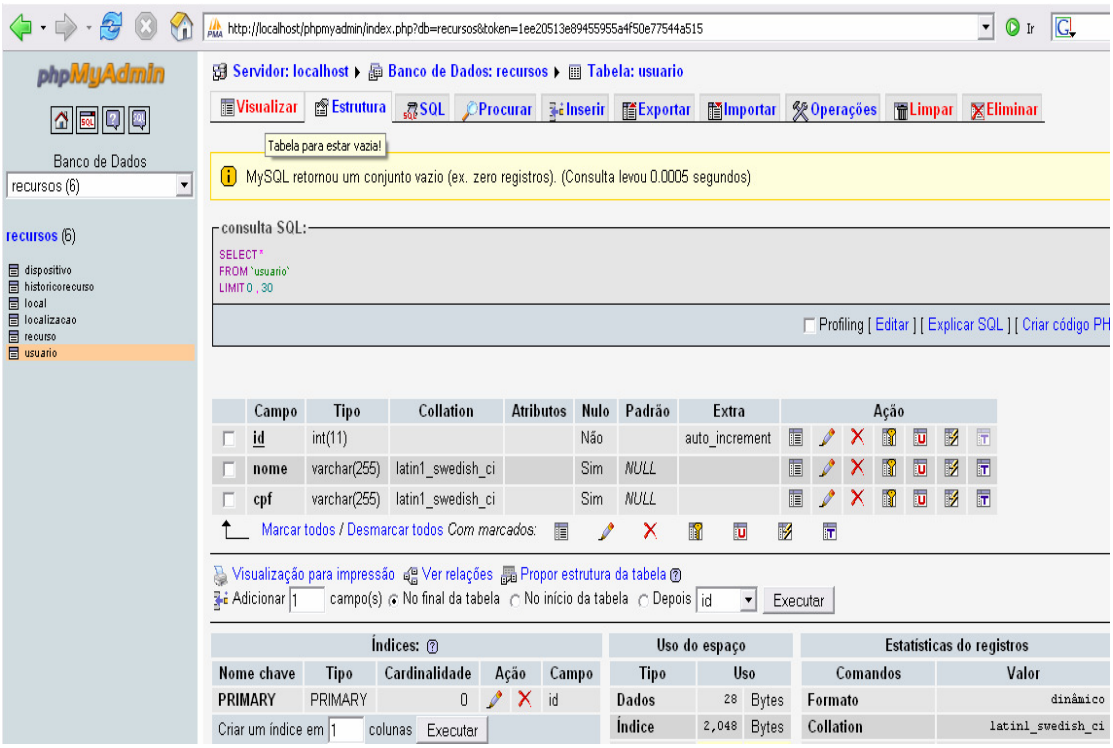

Figura 22: Banco Dados vazio.

Na figura 22, pode ser visualizado o banco de dados do sistema completamente vazio, somente com o cadastro das classes do sistema sem nenhum dado inserido. Posteriormente, será apresentado nas demais figuras os acessos e controles do sistema, com as devidas inclusões, alterações. No sistema proposto, essa tela é representada pelo módulo (SC), módulo destinado ao armazenamento de dados referentes ao usuário no ambiente local.

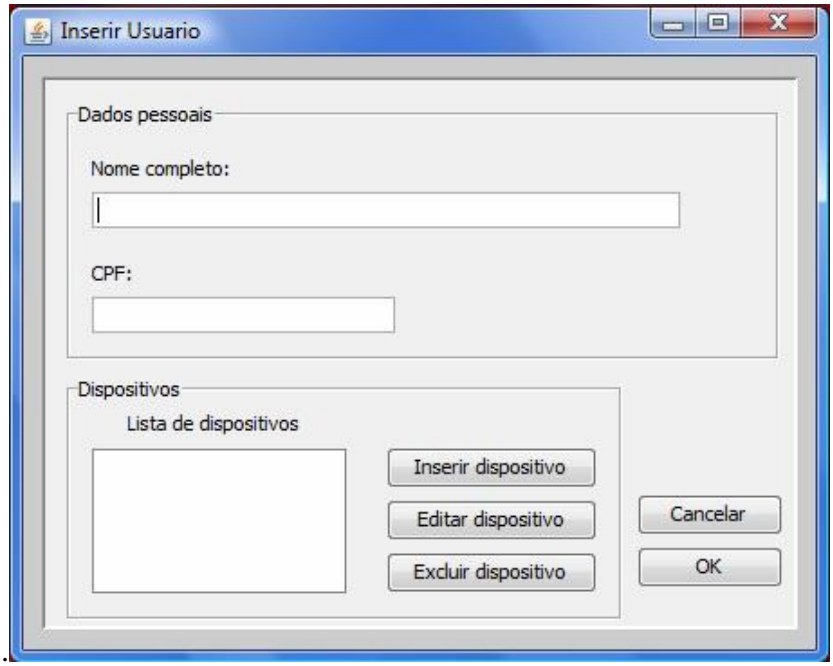

Figura 23: Cadastro de usuário.

A figura 23 apresenta a tela de cadastro do usuário, com a descrição dos campos nome e CPF do usuário. Também através desta tela é possível inserir, editar e excluir dispositivo do usuário. No sistema proposto essa tela é representada pelo módulo (SC) que onde serão armazenados os dados referentes ao usuário.

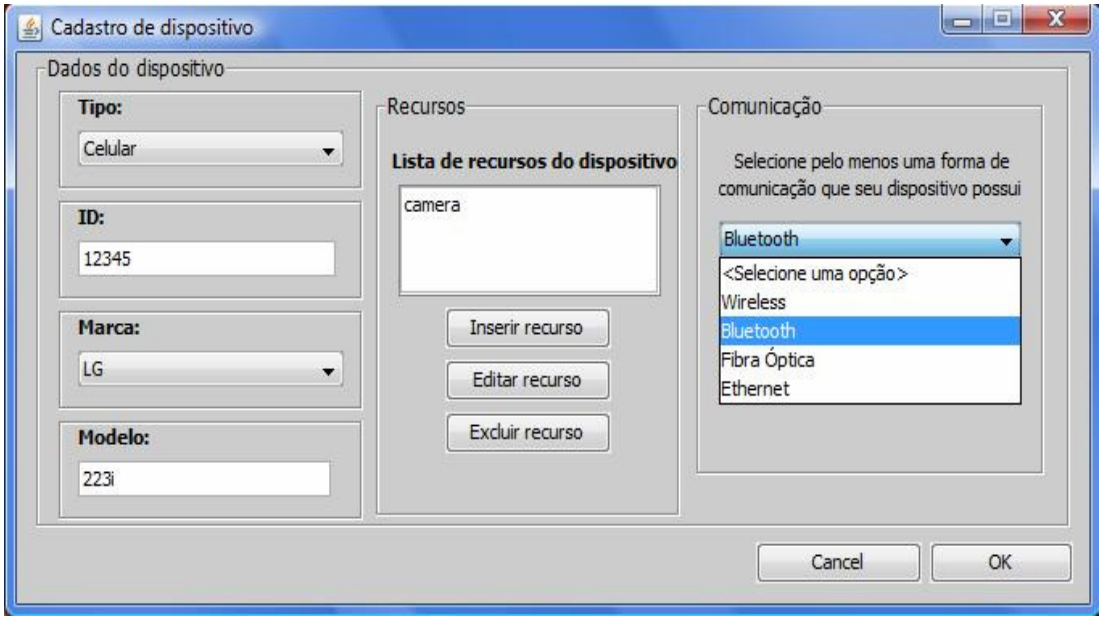

Figura 24: Tela cadastro de dispositivo.

Conforme a figura 24 pode-se visualizar a tela de cadastro de dispositivo que contém os dados com o tipo do dispositivo, um ID que deve ser único para cada dispositivo, marca e o modelo do mesmo, está tela também possui uma lista de recursos que podem inseridos, editados para posterior atualização e exclusão do mesmo, No sistema proposto essa tela é representada pelo módulo (SC) onde serão armazenados os dados referentes ao usuário, os recursos serão armazenado no módulo (CR).

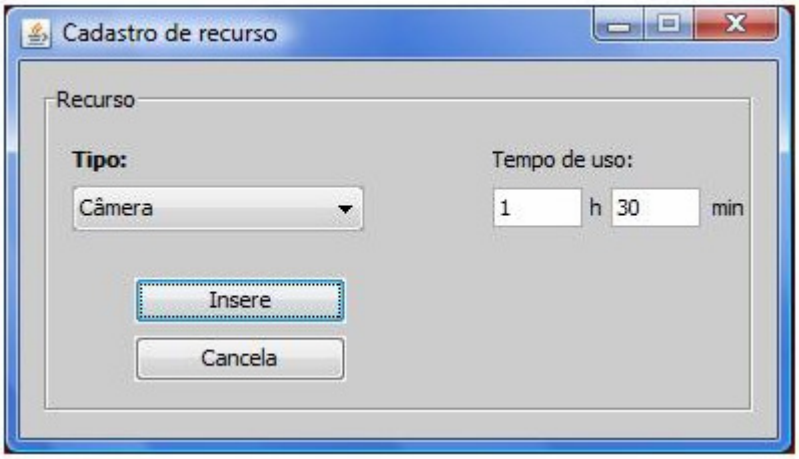

Figura 25: Tela cadastro de recursos.

A figura 25, mostra o cadastro de recursos que poderão ser inseridos o tipo de recurso de cada dispositivo, por exemplo, um celular pode conter como recurso câmera, filmadora ou outro recurso de acordo com a marca e modelo de cada aparelho. Um notebook poderia ter como recurso uma quantidade x de armazenamento disponibilizado como recurso do próprio dispositivo e para uso de usuários do sistema, esse dispositivo poderia ter outros recursos como tamanho de tela maior do que normalmente um celular possui. Através desse cadastro, também é possível especificar o tempo de uso para cada recurso, impedindo assim, que determinado usuário fique infinitamente utilizando determinado recurso.

5.2 Validação do Sistema

O objetivo dessa seção é mostrar o funcionamento da aplicação, visualizar o comportamento do sistema e fornecer uma validação funcional, mostrando a realização do serviço através de telas e logs capturadas no sistema. Para alcançar esse objetivo, seis passos são relacionados a essa aplicação para serem avaliados localmente:

- Inicialização do ambiente e dos dispositivos;
- Solicitação de acesso a aplicação pelo administrador do sistema;
- Simulação de entrada de um usuário no ambiente;
- Atualização de local e data entrada no usuário no ambiente;
- Verificação do dispositivo e recursos disponibilizados;
- E verificação do tempo Maximo de uso de cada recurso.

Em cada simulação, serão analisado todas passo a passo as operações realizadas entre os usuários, dispositivos e recursos, onde em cada passo da simulação.

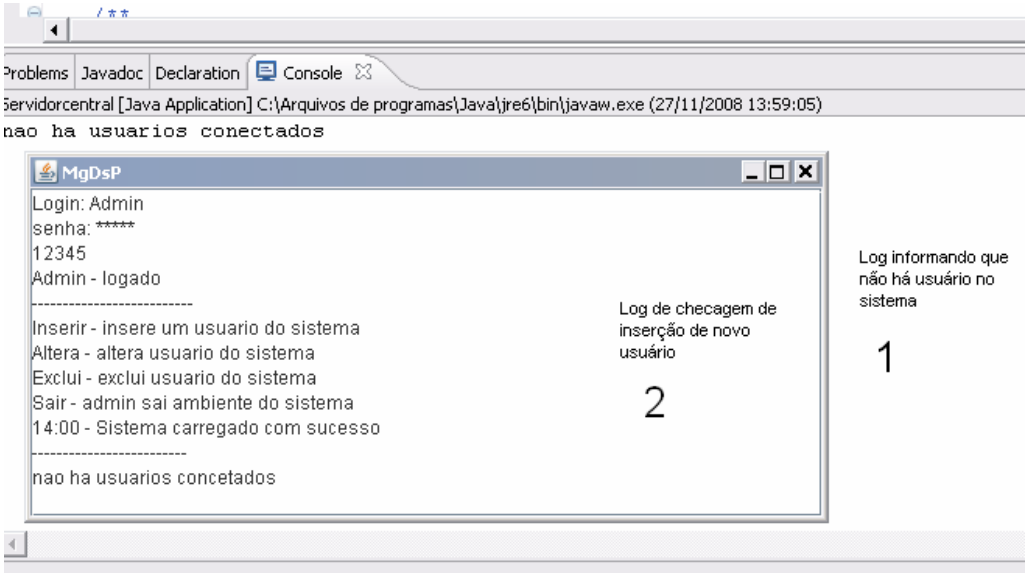

Figura 26: Acesso ao sistema pelo Administrador

Na figura 26, observar-se, passo 1, o console da aplicação informando que não há usuários no sistema, esse passo 1 refere-se ao log gerado após a entrada do administrador no sistema. Quando o sistema é inicializado pelo administrador, é informado que não há usuários conectados, observar-se também no log a data e horário em que foi realizado o registro. O acesso pelo administrador pode ser visualizado pelo passo 2 da mesma figura 26 onde é informada a validação do login e senha, na seqüência as funções habilitadas para utilização do sistema que são mostradas, inserir, altera, exclui, sair. Essas descrições podem ser visualizadas de acordo com as figuras 20 e 21, pode-se visualizar também o ambiente no qual o administrador terá acesso. Também é mostrado no log o horário que o sistema foi inicializado e posteriormente o log com horário e dados relativos a usuário conectados ao sistema, neste caso em especifico não havia usuários conectados.

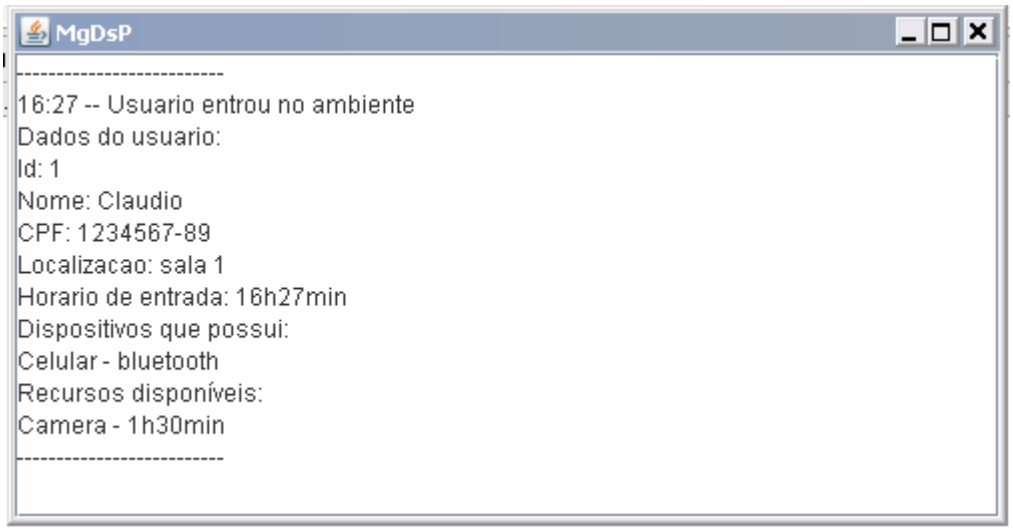

Figura 27: Entrada de usuário no sistema

A figura 27 apresenta o log de registro que o sistema gerou quando o usuário entrou no ambiente, visualiza-se o controle de horário neste caso 16:27 Hs. Os respectivos dados do usuário contendo o Id, o Nome e CPF, também é registrado a localização do usuário que entrou no ambiente, neste caso, sala 1, um novo registro de tempo. Entretanto esse registro de tempo relativo ao ambiente sala 1, é relativo ao fato do usuário poder se movimentar em um ambiente pervasivo, tornando assim, necessário o registro de entrada e saída em ambiente diferentes. Também pode ser visualizado o log com os dados referentes aos dispositivos que usuário possui neste caso um aparelho celular comunicando-se através de bluetooth e por fim os recursos disponíveis no dispositivo com o respectivo tempo disponibilizado para utilização, na ocasião de 1h e 30min, ou seja, esse é o tempo limite, que outro usuário que esteja utilizando o sistema terá para utilizar o recurso, caso necessite.

Algumas características que foram definidas poderão ser modificadas, podemos destacar, por exemplo, inserção de outras características sobre o usuário foi inseridos apenas nome e CPF, apenas para efetuar a simulação e validação do sistema. Embora não foi objetivo deste trabalho estudos, que envolvessem conceitos de tolerância a falhas. Na figura 28 pode ser visualizado o log de outro usuário entrando no ambiente, a localização do novo usuário agora é na sala 2, dispositivo detectado notebook com comunicações Bluetooth e wireless, os recursos disponíveis são armazenamento em disco com tempo de 2h e câmera com tempo definido de uso de 1h e 30min. A implementação ou melhoria da própria aplicação visando o controle do tamanho e tipo de acesso ao meio de armazenamento disponibilizado por alguns dispositivos como recursos é tema para posteriores trabalhos.

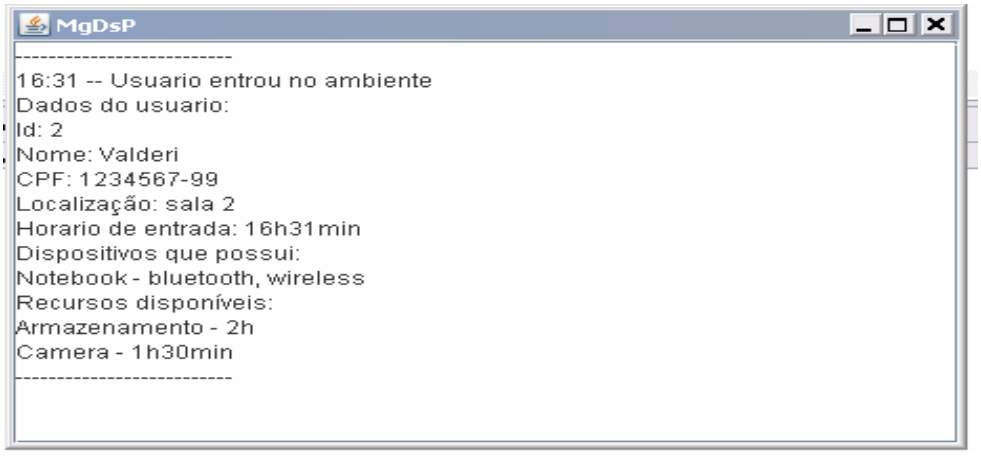

Figura 28: Nova entrada de usuário no sistema

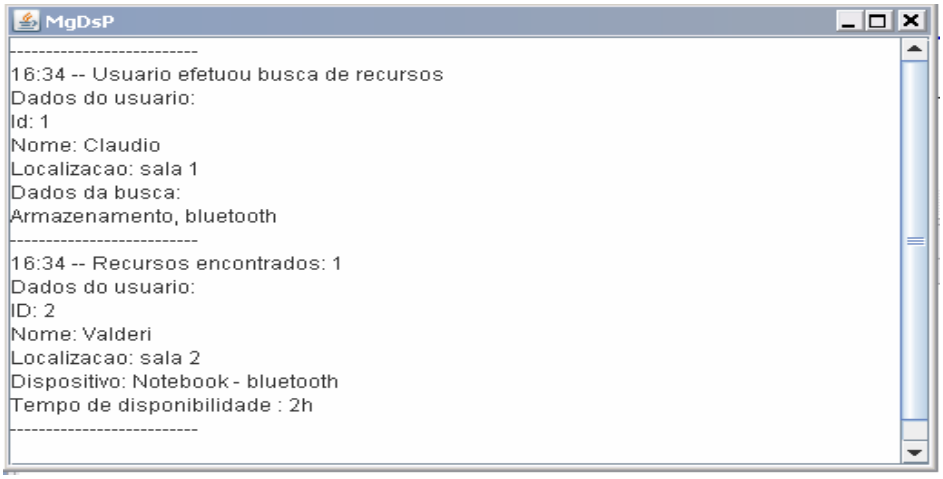

Figura 29: Busca de recursos

A figura 29 apresenta o log de registro do usuário efetuando busca de recursos disponíveis no sistema, informando o horário e dados pessoais como Id e nome, localização e a solicitação dos dados que está buscando registrados no log gerado pelo sistema. Em seguida são retornados através do log simulado às informações sobre os recursos encontrados, apenas 1 recurso, conforme solicitação e os dados do usuário que possui esses recursos, como: Id, nome, a localização sala 2, dispositivo encontrado e forma de comunicação utilizada e o tempo de disponibilidade para uso desse recurso que neste caso é de 2h. Essas informações ficam disponíveis na tabela de serviços figura16 que possui finalidade de gerenciar os serviços disponibilizados pelo dispositivo do usuário e acessam seus recursos, como as informações encontradas estavam em ambientes distintos houve acesso ao servidor de contexto de uso global. Neste caso simulado houve a utilização de um banco de dados mysql [SUN08]. A seguir pode ser visualizada a figura 30, do banco de dados com registro alterados, depois de serem feitas as simulações descritas acima e confirmadas através dos logs gerados e visualizados nas figuras 26,27,28 e 29.

A figura 30 apresenta 4 passos que descrevem os registros da simulação realizada, utilizando o banco de dados mysql disponibilizado pela [SUN08], validações sobre o mesmo foram detalhadas no logs gerados pelo sistema figuras 26,27, 28 e 29; Segue uma breve descrição sobre a figura 30:

- Na descrição 1 da figura 30, é visualizado o registro do usuário: neste caso foram feitos registros que podem ser confirmados através do logs gerado pelo sistema, figura 27, 28: claudio, valderi;
- Na descrição 2 da figura 30, é visualizado o registro do dispositivo utilizado pelo usuário: celular;

- Na descrição 3 da figura 30, é visualizado o registro do recurso do dispositivo utilizado pelo usuário: câmera, o tempo de utilização deste recurso tem por finalidade controle de utilização de determinado recurso;
- Na descrição 4 da figura 30, é visualizado o registro da localização do usuário: sala 1;
- Na descrição 5 da figura 30, é visualizado o registro do horário em o usuário entrou no ambiente pervasivo, sala 1.

Figura 30: Registro banco de dados

5.3 Considerações Finais

Neste capítulo, foram descritos todos os passos que envolveram a validação da proposta, foi descrito como foi implementado, testado e o funcionamento descrito através de logs e telas do modelo proposto. No próximo capitulo serão descritas as considerações finais, conclusões e contribuições de pesquisa sobre esse trabalho, bem como sugestões sobre trabalhos futuros que podem contribuir com o desenvolvimento deste trabalho e com a pesquisa envolvendo a computação pervasiva.

6 CONCLUSÃO E TRABALHOS FUTUROS

A computação pervasiva é o atual cenário computacional que vem sendo considerado o mais recente paradigma computacional desse século. O interesse de diversos grupos de pesquisa, tanto acadêmicos quanto industriais, para a computação pervasiva, cresce em escala global. Esse novo cenário computacional contempla o desenvolvimento de aplicações com novas funcionalidades em ambientes inteligentes, como por exemplo, uma casa inteligente, entre outros citados ao longo desse trabalho. A computação pervasiva disponibiliza ao usuário acesso computacional de modo invisível e onipresente. A construção de aplicações e ambientes pervasivos, no entanto, proporciona muitos desafios. Um desses desafios é tratar o contexto de usuários, dispositivos e recursos disponibilizados, sem desconsiderar outros fatores que englobam um sistema pervasivo. Por isso, o presente trabalho apresentou a implementação de um modelo gerenciador de serviços para plataformas pervasivas sensíveis ao contexto que seja capaz de adaptar um ou mais dispositivos embarcados a um determinado ambiente pervasivo. Essa plataforma pode ser facilmente integrada em uma rede pervasiva e pode fornecer recursos e serviços de um ou mais dispositivos embarcados a outros dispositivos existentes no ambiente conectados a rede pervasiva.

Nesse trabalho, foram estudados alguns conceitos relacionados à computação pervasiva, e ciência ou sensibilidade ao contexto como alguns autores descrevem [GEY08]. O conceito de ambientes, aplicações, dispositivos, plataformas e redes pervasivas foram mostrados para visualizar as diferenças com a computação tradicional. Dois conceitos foram importantes na definição do trabalho. O conceito utilizado para arquitetura pervasiva foi aplicado para todos os dispositivos pervasivos existentes no ambiente, enquanto o conceito de plataforma é baseado em um dispositivo computacional genérico capaz de permitir o reuso de aplicações em software. Os requisitos necessários para o desenvolvimento de um ambiente pervasivo foram estudados a fim de auxiliar na concepção do modelo proposto no trabalho. Esses requisitos também foram utilizados para o estudo relacionado a arquiteturas e plataformas pervasivas. O estudo de arquiteturas pervasivas foi importante para avaliar as necessidades que os ambientes inteligentes necessitam, em termos de serviços e recursos.

As pesquisas na área de computação pervasiva tendem a se desenvolver, devido às várias formas de comunicações que emergem com velocidades de envio e recebimento de dados de altíssimas velocidades. Dispositivos com funcionalidades jamais imaginadas, agregação de funcionalidades através de disponibilização de recursos e serviços. Aplicações sensíveis ao contexto

será desafio ainda fonte de muito trabalho, este trabalho propôs um modelo, que entre outras funcionalidades, sendo uma principal o fato de ser genérico suficiente para funcionar adequadamente em diversos tipos de dispositivos e formas de comunicações.

6.1 Contribuições de Pesquisa

Este trabalho teve como principal objetivo apresentar um modelo gerenciador de descoberta de serviços para plataformas pervasivas com possibilidade de desenvolvimento e execução de aplicações móveis, sensíveis ao contexto:

- Contribuiu para pesquisa em computação pervasiva, através do modelo proposto;
- Apresentou algumas iniciativas de plataformas de suporte a aplicações móveis sensíveis ao contexto;
- Descreveu o projeto conceitual da plataforma existente, um modelo de descoberta de serviços para o desenvolvimento e à construção de aplicações móveis, sensíveis ao contexto;
- Apresentou um estudo de caso em que se aplica o modelo gerenciador de descoberta de serviços ao contexto proposto nesse trabalho. Tal cenário trata de um sistema de gerenciamento adaptável a plataformas pervasivas;
- Estudou, analisou e desenvolveram componentes e serviços específicos que permitiram um tratamento eficaz para a computação pervasiva, tais como tratamento de falhas, métodos mais eficientes para segurança, comunicação entre dispositivos totalmente heterogêneos.
- Expuseram as considerações finais, avaliações sobre o trabalho realizado e uma breve discussão sobre trabalhos futuros.

6.2 Trabalhos Futuros

Essa seção trata sobre trabalhos a serem realizados futuramente no mesmo tópico de pesquisa, também pretende expor pontos a serem mais bem estudados e verificados que não puderam ser estudados neste trabalho exaustivamente, são eles:

- Adaptação e estudos exaustivos com a finalidade de utilização e adaptação a WebService;
- Utilização e adaptação a sistemas pervasivos que utilizam o conceito de ontologias;
- Utilização e adaptação a sistemas e plataformas pervasivas que utilizam o conceito de WebSemântica;
- Definição de controle e acesso ao recurso de armazenamento, disponibilizado como recurso por alguns dispositivos;
- Desenvolvimento de técnicas de adaptação de contexto local e global com uso de diferentes plataformas pervasivas, adaptadas ao modelo gerenciador proposto;
- Estudo sobre utilização e adaptação do modelo a uma Rede de Sensores sem Fios (RSSFs);
- Tratamento de questões relativas à segurança do modelo proposto, bem como na troca de informações entre usuários e dispositivos;
- Desenvolvimento e adaptação do modelo proposto aos demais módulos propostos pela arquitetura proposta por [RIE06].

REFERÊNCIAS BIBLIOGRÁFICAS

[ADJ99] Adjie-Winoto, W., Schwartz, E., Balakrishnan, H., e Lilley, J. "The design implementation of an intentional naming system". Operating Systems Review, Volume 34 , Issue 22 (April, 2000) ACM, NY - USA.

[AND03] Andersen, K. V. B., Cheng, M., Klitgaard-Nielsen, R. "Online Aalborg Guide – Development of a Location-Based Service". Technical report, Aalborg Universitet, 2003.

[AND04] Andreão, R. V. "Segmentation de battements ECG par approche markovienne: application à la détection d'ischémies". Tese de Doutorado, Dep. EPH, Institut National des Télécommunications, Evry, França, 2004., 143 p.

[AUG04] Augustin, I. "Abstrações para uma Linguagem de Programação Visando Aplicações Móveis em um Ambiente de Pervasive Computing". Tese de Doutorado, UFRGS, 2004, 196 p.

[AVA02] Avancha, S., Joshi, A., e Finin, "Tenhanced service discovery in Bluetooth". *Computer Communications*, Departamento de Informática Pontifícia Universidade Católica do Rio de Janeiro (PUC-RJ). 2002, p. 2 - 8.

[BAR02] Barbosa, A. C. P., Porto, F., Melo, R. N. Configurable Data Integration Middleware System. J. Braz. Comp. Soc., v. 8, n. 2, p. 12-19, 2002.

[BAR08] Bardram, Jakob E. e Thomas R. Hansen. "Centre for Pervasive Healthcare Department of Computer Science", University of Aarhus Aabogade 34, DK8200, Aarhus N., Denmark, Ultimo Acesso em abril 2008.

[BIL03] Billo, A. E; "Uma pilha de protocolos Bluetooth adaptável à aplicação". Trabalho de Conclusão de Curso, UFSC 2003, 82 p.

[BON05] Bonatto, D, J. Barbosa, G. Cavalheiro, J. Ramos. "Pholo: uma arquitetura para computação pervasiva utilizando o holoparadigma," In: VI Workshop em Sistemas de Alto Desempenho (WSCAD), 2005, capturado em Http://www.inf.unisinos.br /˜barbosa/textos/ WSCAD_ 2005. pdf. Junho 2006, 8p.

[BUR02] Burkhard, T J., Henn, H., Hepper, S., Rindtorff, K., Schack, T. "Pervasive Computing: Technology and Architecture of Mobile Internet Applications". Ed. Addison- Wesley, 2002. 410 p.

[CAC05] Cáceres, C., Fernández, A., Ossowski, S. "CASCOM - Context-Aware Health-Care Service". Co-ordination in Mobile Computing Environments. *ERCIM News*, 60, 2005.

[CFP06] CfPH - Centre for Pervasive Healthcare. 2006. Capturado em http://www. pervasivehealthcare.dk/. Março 2006.

[CHE02] Chen, G., KOTZ, D. "Context Aggregation and Dissemination in Ubiquitous Computing Systems".In: Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications, Callicoon, New York, pp. 105-114, June, 2002.

[CHE04] Chen, H. "An Intelligent Broker Architecture for Pervasive Context-Aware Systems". Ph.D. Thesis. University of Maryland, USA, 2004, 174 p.

[COS03] Costa, P. D. "Towards a Services Platform for Context-Aware Applications". Master Thesis, University of Twente, Enschede, The Netherlands, 2003.

[DEY00] Dey, A. K. "Providing Architectural Support for Building Context-Aware Applications". Ph.D. Thesis. Georgia Institute of Technology, 2000, 146 P.

[DEY00a] Dey, A. K. Gregory D. Abowd, "CybreMinder: A Context-Aware System for Supporting Reminders. NSF CAREER Grant # 9703384, a Motorola University Partnerships in Research grant and the ITO division of DARPA through the Expeditions/Ubiquitous Computing program., 2000, 15 p.

[DUR04] Dürr, F., Höhnle, N., Nicklas, D., Becker, C., Rothermel, K. "Nexus - A Platform for Context-aware Applications." In: Proceedings GI-Fachgespräch "Ortsbezogene Dienste", Hagen, Alemanha, 2004.

[EFS06] Efstratiou, C. "Coordinated Adaptation for Adaptive Context-aware Applications". Ph.D. Thesis, Lancaster University. Egnos Project, 2004. Capturado em http://www.esa.int/esaNA/GGG63950NDC_index_0.html. Maio 2006.

[FIL06] Filho, J. G. P, et. al. "Infraware: Um *Middleware* de Suporte a Aplicações Móveis Sensíveis ao Contexto", UFES, Unisantos. Capturado em: www.sbrc2007.ufpa.br/ anais/2006/ST/SC2_4.pdf; 2006, 4 p.

[FIL07] FieldMap. FieldMap 3.3.6 Manual. Capturado em: http://www.piranesi.dyndns.org /FieldMap/manual/. Dezembro 2007.

[FIL07] FieldMap. FieldMap 3.3.6 Manual. Capturado em: http://www.piranesi.dyndns.org/ FieldMap/manual/. Dezembro 2007.

[GEY08] Geyer, C. F. R. et al. "Um Serviço Adaptativo para Descoberta de Recursos em Larga-Escala na Arquitetura ISAM". In: Workshop de Redes Peer-to-Peer - WP2P 2005, 12 p.

[GUP02] Gupta, R., Talwar, S., Agrawal, D. "Jini Home Networking: a Step toward Pervasive Computing". *IEEE Computer*, New York, vol.35 - 8, Aug. 2002. pp.34-40.

[GUT04] GU, T., et al. “A Middleware for Building Context-Aware Mobile Services”. In: Proceedings of IEEE Vehicular Technology Conference, Milan, Italy, 2004.

[GUZ05] Gu, T., Pung, H. K. e Zhang, D. Q. “A Service-oriented Middleware for Building Context-aware Services”. *Journal of Network and Computer Applications*, vol: 28 - 1, Jan. 2005 pp.1–18.

[HAN01] Hansmann, U., Merk, L., Nicklous, M.S., Stober, T. “Pervasive Computing Handbook”, Ed. Springer, 2001. 409 p.

[HAN05] Hansen, T. R., Eriksson, E., Lykke-Olesen, A. “Mixed Interaction Space” — Expanding the Interaction Space with Mobile Devices Centre for Pervasive Healthcare & Centre for Interactive Spaces, ISIS Katrinebjerg, Department of Computer Science, University of Aarhus. Outubro 2007.

[HAZ08] Hazas, M., Kray, C., Gellersen, H., Agbota, H., Kortuem, G. “Computing Department”. Lancaster University, United Kingdom. Ultimo Acesso em março 2008.

[HEN01] Henricksen, K., Indulska, J., Rakotonirainy, A. "Infrastructure for pervasive computing: challenges", GI Jahrestagung, 2001, pp.214-222.

[JOR02] Jorge, L.V.B. “Holoparadigma: um modelo multiparadigma orientado ao desenvolvimento do software distribuído.” Tese de Doutorado, UFRGS, 2002, 213p.

[JOS03] Jose, R., Moreira, A., Rodrigues, H., Davies, N. “The AROUND Architecture for Dynamic Location-Based Services”. *Mobile Networks and Applications*, vol. 8, 2003, p. 377-387.

[KAV04] Kaveh, N., Hercock, R. G. “NEXUS: Resilient Intelligent Middleware.” *BT Technology Journal*, vol. 22 - 3, Jul. 2004, pp. 209–215.

[LEA07] Leão, L. O. “Modelo integrado de computação pervasiva para ambiente médico hospitalar de tratamento intensivo”. Dissertação Mestrado, PUCRS, 2007. 98 p.

[LIL05] Li Li, L. L. “Communications Reserach Centre of Canada”; A Lightweight Service Discovery Mechanism for Mobile Ad Hoc Pervasive Environment Using Cross-Layer Design, “OSGi-based service gateway arquiteture for intelligent automobiles”, p. 861-865, Junho 2005.

[LIM07] Lima, S. L. “Um protocolo para descoberta e seleção de recursos grades móveis Ad Hoc”. Tese de Doutorado, PUC RIO 2007, pp. 27 e 28.

[LIS04] Liszka, K. J., Mackin, M. A., Lichter, M. J., York, D. W., Pillai, D., Rosenbaum, D. S. “Keeping a Beat on the Heart”. *IEEE Pervasive Computing*, vol. 03 - 4, Out./dez.

2004, pp. 42-49.

[LYY02] Lyytinen, K. e Yoo, Y. “Issues and Challenges in Ubiquitous Computing”, *Communications of the ACM*, vol. 45 - 12, Dez. 2002.

[MAT04] Mattes, L., Kiatake, L., Marques, E., Zuffo, J, "Linguagem lógica formal para expressar segurança em ambientes pervasivos". In: IV Workshop em Segurança de Sistemas Computacionais, 2004.

[MUR01] Murphy, A., Picco, G. e Roman, G. C. “Lime: a middleware for physical and logical mobility”. In: 21st International Conference in Distributed Computing Systems. Proceedings, Phoenix, Arizona, USA (ICDCS 2001).

[NAK04] Nakamishi, Y., Takahashi, K., Tsuji, T., Hakozoki, K. “ICAMS: A mobile communication tool using location and schedule information”. *IEEE Pervasive Computing*, vol. 3 - 1, Jan./mar. 2004, pp. 82-88.

[RIE06] Ries, L. H. L. “Uma Plataforma para Integrar Dispositivos Eletrônicos em Ambientes Pervasivos”. Dissertação de Mestrado, PUCRS, 2006. 80 p.

[SAT01] Satyanarayanan, M. “Pervasive computing: vision and challenges”. *IEEE Personal Communication*, Ago. 2001, pp. 10-17.

[SUN08] Sun Microsystem. “Sun Java System Application Server”. Capturado em: http://www.sun.com/software.Março 2008.

[WEI04] Weibenberg, N., Voisard, A., Gartmann, R. “Using Ontologies in Personalized Mobile Applications”. In: Proceedings Intl. ACM GIS Conference, ACM Press, 2004, pp. 2-11.

[WEI91] Weiser, M. “The Computer for the Twenty-First Century”. *Scientific American*, Set. 1991, pp. 94- 10.

[YAM04] Yamim, A. C. “Arquitetura para um Ambiente de Grade Computacional Direcionado às Aplicações Distribuídas, Móveis e Conscientes do Contexto da Computação Pervasiva”. Tese de Doutorado, Programa de Pós-Graduação em Computação, UFRGS / Porto Alegre – RS, 2004, 194 p.

[ZHU02] Zhu, F., Mutka, M., e Ni, L. “Classification of service discovery protocols in pervasive computing environments”. Technical Report MSU-CSE 02-2, Michigan State University, East Lansing, 2002.