UNIVERSIDADE FEDERAL DE PELOTAS

Centro de Desenvolvimento Tecnológico

Programa de Pós-Graduação em Computação

Dissertação

Uma Arquitetura de Software direcionada à Consciência de Contexto na *UbiComp*

MÁRCIA ZECHLINSKI GUSMÃO

Pelotas, 2013

MÁRCIA ZECHLINSKI GUSMÃO

Uma Arquitetura de Software direcionada à Consciência de Contexto na *UbiComp*

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal de Pelotas, como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação

Orientador: Prof. Dr. Adenauer Corrêa Yamin

Pelotas, 2013

Dados de catalogação na fonte:
Leda Cristina Peres Lopes – CRB:10/2064
Biblioteca de Ciência & Tecnologia – UFPel

G982u Gusmão, Márcia Zechlinski

Uma Arquitetura de Software direcionada à Consciência de Contexto na *UbiComp* / Márcia Zechlinski Gusmão. – Pelotas, 2013. – 95 f: gráf. – Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal de Pelotas. Centro de Desenvolvimento Tecnológico. Pelotas, 2013. – Orientador Adenauer Corrêa Yamin.

1. Consciência de contexto. 2. Interoperabilidade de servidores. 3. Arquitetura. I. Yamin, Adenauer Corrêa. II. Título.

CDD: 004.22

Banca examinadora:

Prof. Dr. Cristiano André da Costa

Profª. Drª. Renata Hax Sander Reiser

Prof. Dr. Gerson Geraldo H. Cavalheiro

Dedico. . . este trabalho ao meu filho Gabriel Gusmão Ferraz, meu amigo, meu parceiro de vida, meu amor, meu anjo!!!

AGRADECIMENTOS

Agradeço a minha família, em especial a minha irmã Adriana pela força e colaboração.

Ao IFSul *Campus* Pelotas, pela grande oportunidade de passar por esta experiência tão gratificante a realização do Curso de Mestrado em Ciência da Computação.

Ao meu orientador, Prof. Dr. Adenauer Corrêa Yamin, pelo apoio, paciência, credibilidade e compreensão que me proporcionou.

Aos companheiros de grupo professores João L. B. Lopes, Ana Marilza Pernas e Gizele Ingrid Gadotti; aos alunos de graduação Paulo Raseira Gomes, Miguel Satte Alam Lisboa, Ricardo Borges Almeida, Maurício Madruga de Azevedo e Diego Porto Jaccottet por todo o incentivo e contribuições valiosas.

Aos integrantes do LUPS, *Laboratory of Ubiquitous and Parallel Systems*, sempre prontos a ajudar, Rodolfo e Rodrigo.

Aos professores e funcionários do PPGC, em especial a Martha e Fabiano, pelo apoio, carinho e dedicação demonstrados.

Aos colegas Adriano, Cícero, Henrique, Julio Ruzicki, Julio Francisco, Luis, Lidiane, Melissa, Milena, Murian, Rafael Bandeira, Rafael Bastos, Ricardo, Vagner, Vladimir pela motivação, amizade e carinho diários.

Por último, agradeço a todos aqueles que não foram mencionados, mas que de alguma forma também contribuíram para a elaboração deste trabalho.

“Os ideais que iluminaram o meu caminho
são a bondade, a beleza e a verdade.”
— ALBERT EINSTEIN

RESUMO

GUSMÃO, Márcia Zechlinski. **Uma Arquitetura de Software direcionada à Consciência de Contexto na *UbiComp***. 2013. 95 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal de Pelotas, Pelotas.

A Computação Ubíqua tem como premissa central permitir ao usuário o acesso ao seu ambiente computacional a partir de qualquer lugar, todo o tempo, a partir de qualquer dispositivo. O trabalho desenvolvido contribui para o Subsistema de Adaptação e Reconhecimento de Contexto do EXEHDA. O mesmo propõe para o serviço de consciência de contexto uma arquitetura, que de forma distribuída, ofereça suporte as etapas de aquisição, armazenamento e processamento de informações contextuais, bem como os decorrentes procedimentos de atuação sobre o meio. O objetivo geral é fornecer uma solução capaz de adquirir de forma autonômica informações contextuais, bem como permitir atuação remota sobre o meio, através de dispositivos eletromecânicos. Para avaliar as funcionalidades da proposta utilizou-se o estudo de caso Projeto AMPLUS (*Automatic Monitoring and Programable Logging Ubiquitous System*) que tem por objetivo promover soluções da Computação Ubíqua para o Laboratório Didático de Análise de Sementes (LDAS) da FAEM/UFPEL. Dentre os serviços a serem providos, destaca-se a consciência de contexto em que se encontram os equipamentos do LDAS, com o respectivo registro histórico do contexto, e uma atuação proativa quando necessário.

Palavras-chave: Consciência de contexto, interoperabilidade de servidores, arquitetura.

ABSTRACT

GUSMÃO, Márcia Zechlinski. **Requirements for a software architecture targeted for Context Awareness in Ubicomp**. 2013. 95 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal de Pelotas, Pelotas.

Ubiquitous Computing has as central premise to allow the user access to their computing environment from any place, any time, from any device. The developed work contributes to the subsystem of Adaptation and Context Recognition of EXEHDA. It proposes an architecture to the context awareness service, which in a distributed way, supports the stages of acquisition, storage and processing of contextual information, and the corresponding actuations on the environment. The general objective is to provide a solution able to acquire contextual information in a autonomic way, as well as allow remote acting on the environment, using electromechanical devices. To evaluate the functionality of the proposal was used as case study the Project AMPLUS (Automatic Monitoring and Programable Logging Ubiquitous System), which is designed to promote Ubiquitous Computing solutions to the Laboratório Didático de Análise de Sementes (LDAS) da FAEM/UFPEL. Among the services to be provided, stands out the context awareness which they are the LDAS equipments, with its respective historic track of context, and a proactive actions when necessary.

Keywords: context-awareness, servers interoperability, architecture.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

BOD	Biological Oxigen Demand
ESB	Enterprise Service Bus
EXEHDA	Execution Environment for High Distributed Applications
Gtalk	Google Talk
GPS	Global Positioning System
G3PD	Grupo de Pesquisa em Processamento Paralelo e Distribuído
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
HTTPs	Hypertext Transfer Protocol Secure
IEEE	Institute of Electrical and Electronics Engineers
ISAM	Infraestrutura de Suporte à Aplicações Móveis
LDAS	Laboratório Didático de Análise de Sementes
LED	Díodo Emissor de Luz
LUPS	Laboratory of Ubiquitous and Parallel Systems
P2P	Peer-to-Peer
PDA	Personal Digital Assistant
RDF	Resource Description Framework
RCN	Repositório de Contextos Notificados
RPC	Remote Procedure Call
SMS	Short Message Service
SOAP	Simple Object Access Protocol
UbiComp	Computação Ubíqua
UFPeL	Universidade Federal de Pelotas
WEB	World Wide Web
XML	eXtensible Markup Language
YAML	YAML Ain't Markup Language

SUMÁRIO

1 INTRODUÇÃO

As tecnologias mais profundas são aquelas que desaparecem, elas se integram na vida cotidiana até se tornarem indistinguíveis da mesma (WEISER, 1991). Esta frase do clássico artigo sobre a Computação para o século 21, sintetiza o que é esperado com a Computação Ubíqua (*UbiComp*). O termo trata dos aspectos referentes ao acesso ao ambiente computacional do usuário, isto é, ao espaço ubíquo do usuário, em qualquer lugar, todo o tempo com qualquer dispositivo. Nesta perspectiva a computação e seus diversos sistemas interagem com o ser humano a todo o momento, não importando onde ele esteja, em casa, no trabalho ou na rua, constituindo este de um ambiente altamente distribuído, heterogêneo, dinâmico, móvel, mutável e com forte interação entre homem e máquina (POSLAD, 2009).

Na Computação Ubíqua as aplicações precisam ter consciência do seu contexto de interesse, e quando for o caso, reagir ao mesmo. Essa nova classe de sistemas computacionais, adaptativos ao contexto, abre perspectivas para o desenvolvimento de aplicações mais ricas, elaboradas e complexas, que exploram a natureza dinâmica das modernas infraestruturas computacionais e a mobilidade do usuário.

Para se construir e executar aplicações ubíquas sensíveis ao contexto, há uma série de funcionalidades que devem ser providas, envolvendo desde a aquisição de informações contextuais, a partir do conjunto de fontes heterogêneas e distribuídas, até a representação dessas informações, seu processamento, armazenamento, e realização de inferências para seu uso na tomada de decisão. Registra-se uma tendência de remover estas funcionalidades das aplicações, repassando as mesmas para *middlewares* de provisão de contexto (BELLAVISTA et al., 2012).

Por sua vez, contexto é qualquer informação que pode ser usada para caracterizar a situação de uma entidade (pessoa, lugar ou objeto) que é considerado relevante para a interação entre usuário e a aplicação, incluindo o próprio usuário e a própria aplicação.

Apesar desta visão ter sido introduzida já a algum tempo, na publicação histórica (DEY; ABOWD; SALBER, 2001), ainda existem limitações para o desenvolvimento de aplicações sensíveis ao contexto, pois poucas linguagens e ferramentas estão efetivamente disponíveis para sua programação (LIU; LI; HUANG, 2011).

1.1 Tema

O tema central do trabalho está relacionado a concepção de uma arquitetura de software para suporte a consciência de contexto em uma perspectiva distribuída, tanto para aquisição, como para processamento de informações contextuais e a correspondente atuação no meio. A proposta desenvolvida tem como premissa de desenvolvimento sua integração a um *middleware* para *Ubicomp*, compondo com o mesmo uma solução integrada para atendimento de demandas na área. Do ponto de vista tecnológico foram selecionadas tecnologias escaláveis e robustas para os mecanismos de coleta e atuação junto ao meio.

1.2 Motivação e Objetivos

A perspectiva introduzida pela Computação Ubíqua vem se tornando uma tendência mundial em função do rápido avanço tecnológico dos dispositivos móveis, redes sem fio, redes de sensores, sistemas e dispositivos inteligentes, sistemas distribuídos, grades computacionais, sistemas autônomos e outras tecnologias relacionadas a integração de sistemas. São tecnologias convergentes, que se integram sinergicamente no cotidiano dos usuários. Tendo em vista a dinamicidade, decorrente desta integração, a capacidade de reação ao contexto torna-se um componente importante e fundamental na Computação Ubíqua (KRUMM, 2010).

Aplicações que possam reagir autonomamente a mudanças no ambiente tornam a vida do usuário mais confortável e, por consequência, melhoram a sua qualidade de vida, podendo fazer com que suas necessidades, sejam atendidas de forma pró-ativa, muitas vezes, sem uma solicitação explícita por parte do mesmo.

Nesta visão, o controle da consciência de contexto é essencial para os Sistemas Ubíquos e tem ganho espaço na literatura especializada (COSTA; YAMIN; GEYER, 2008).

Considerando a diversidade presente nos contextos e na natureza das aplicações em geral, prover soluções genéricas que facilitem o desenvolvimento

de aplicações, constitui um desafio para a área de Sistemas Distribuídos e Ubíquos.

Esta dissertação tem por objetivo geral contribuir com o Subsistema de Adaptação e Reconhecimento de Contexto do *Middleware* EXEHDA (YAMIN, 2004). A proposta denomina-se EXEHDA-UC (*Ubiquitous Context awareness*) e propõe para o serviço de consciência de contexto uma arquitetura que, de forma distribuída, ofereça suporte as etapas de aquisição, armazenamento e processamento de informações contextuais e os decorrentes procedimentos de atuação sobre o meio.

Como objetivos específicos do trabalho, são destacadas as seguintes frentes:

- Revisar as premissas de concepção do *middleware* EXEHDA, em particular do Subsistema de Adaptação e Reconhecimento de Contexto;
- Conceber uma arquitetura constituída por servidores de duas naturezas, um destinado a tratar da relação com o meio, replicado em diferentes localizações do espaço celular, e outro com capacidade de compor estados contextuais para toda célula de execução. Os servidores deverão ser dotados de estratégia de interoperabilidade que faculte sua operação combinada tanto intra como inter-células;
- Conceber a infraestrutura de hardware e software necessária para aquisição de informações contextuais e atuação sobre o meio;
- Desenvolver estudo de caso para servidor de contexto, considerando as demandas do Projeto AMPLUS.
- Divulgar ante a comunidade científica os resultados atingidos pela pesquisa através de publicações em eventos e/ou jornais especializados da área.

1.3 Estrutura do Texto

Esta Dissertação está organizada em 6 capítulos. No capítulo 1 é abordado o tema do trabalho, suas motivações e objetivos para o mesmo. No capítulo 2 são tratados os fundamentos conceituais e tecnológicos do trabalho, sendo explorados conceitos em Computação Ubíqua, o tema consciência de contexto, evidenciando os principais conceitos e as etapas de sua aquisição, *Middleware* EXEHDA e por fim as ferramentas *Enterprise Service Bus* (ESB). Uma visão geral e a respectiva modelagem da proposta deste trabalho, será discutida no capítulo 3. Os trabalhos relacionados serão apresentados no capítulo 4,

onde são apresentadas as discussões a cerca dos trabalhos relacionados e o modelo. No capítulo 5 será apresentado o estudo de caso Projeto AMPLUS, sendo suas funcionalidades descritas. O trabalho finaliza no capítulo 6, onde são apresentadas as principais conclusões, os trabalhos futuros, bem como as publicações realizadas.

2 FUNDAMENTOS CONCEITUAIS

Este capítulo tem por objetivo discorrer sobre os principais fundamentos conceituais do trabalho desenvolvido na dissertação. Na sua primeira seção, conceitos pertinentes à Computação Ubíqua são apresentados. Na segunda seção, são explorados conceitos relativos à consciência de contexto, sendo descritos desafios que se apresentam nesta área. Na terceira seção, é apresentado o *Middleware* EXEHDA e seus subsistemas, sendo descritas suas principais características e funcionalidades. O capítulo é finalizado com a seção referente ao estudo realizado sobre ferramentas para interoperabilidade em ambientes distribuídos.

2.1 Computação Ubíqua

Mark Weiser, através de seu artigo histórico (WEISER, 1991), estabeleceu uma visão da Computação Ubíqua (*Ubicomp*), que articulou uma comunidade de pesquisa multifacetada. Esta comunidade tem gerado diversas conferências, periódicos e revistas nos últimos anos.

Com o avanço das tecnologias de comunicação móvel e a miniaturização do hardware, as unidades de processamento estão se tornando invisíveis, dada a sua crescente integração com o meio. *Middlewares* para computação ubíqua tem que administrar três principais características: a mobilidade, a heterogeneidade e a dinamicidade do meio (SILVA et al., 2012).

As premissas da Computação Ubíqua, discutidas pelo cientista Mark Weiser no início dos anos 90, já previam um aumento nas funcionalidades e na disponibilidade de serviços de computação para os usuários finais, ao mesmo tempo em que o custo de gerenciamento dos mesmos iria ficando cada vez menor. Deste modo, a computação não seria exclusividade de um computador, mas de diversos dispositivos conectados entre si.

A Computação Ubíqua, nesta perspectiva, não significa um computador que possa ser transportado para diferentes lugares. Mesmo o mais funcional

notebook, com acesso à Internet ainda foca a atenção do usuário em um único equipamento. Comparando à escrita, carregar este poderoso *notebook* é como carregar um livro muito importante. Por mais significativo que seja este livro, não significa capturar o real poder da Literatura (WEISER, 1991).

A concretização do pensamento de Weiser e, consequentemente, a ubiquidade da informática, terá atingido sua plenitude quando a computação for aplicada com a mesma naturalidade que hoje é utilizada a língua escrita e os motores, agora elétricos e embarcados, para realização de atividades do cotidiano.

Desta forma, a comunidade terá alcançado a era da Computação Ubíqua quando o computador deixar de ser peça única e de uso genérico para multiplicar-se em usos especializados.

Como exemplo desta perspectiva, os óculos são uma interface entre o ser humano e o mundo. O ser humano não concentra suas atenções no óculos, mas no mundo. Uma pessoa cega, ao utilizar sua bengala, percebe o mundo ao seu redor, e não a bengala. Sob esta premissa, o uso da computação na perspectiva das interfaces passa a tornar-se imperceptível (WEISER, 1993).

As aplicações ubíquas, em uma visão mais ampla, devem prever a mobilidade de equipamentos e usuários, denominada mobilidade física, e também dos componentes da aplicação e serviços, chamada de mobilidade lógica. Para isso, as aplicações devem ter o estilo siga-me, facultando que o usuário possa acessar seu ambiente computacional independente da localização, do tempo e do dispositivo utilizado (YAMIN, 2004).

Para tal, as aplicações precisam “entender” e reagir ao ambiente, compreendendo o contexto em que estão inseridas (MACIEL; ASSIS, 2004). Essa nova classe de sistemas computacionais, conscientes do contexto, abre perspectivas para o desenvolvimento de aplicações mais ricas, elaboradas e complexas, que exploram a natureza dinâmica e a mobilidade do usuário. Um desafio central na programação deste tipo de aplicação é possibilitar que as mesmas possam reagir continuamente ao ambiente e permaneçam funcionando mesmo quando o indivíduo se movimenta ou troca de dispositivo (GRIMM; BERSHAD, 2003; COSTA; YAMIN; GEYER, 2008).

Resumindo, pode-se dizer que a Computação Ubíqua constitui um ambiente altamente distribuído, heterogêneo, dinâmico, móvel, mutável e com forte interação entre homem e máquina (LOPES et al., 2012a).

2.2 Consciência de contexto na *UbiComp*

A consciência de contexto refere-se à capacidade de um sistema computacional perceber características do meio que sejam de seu interesse (LIU; LI; HUANG, 2011).

Aplicações conscientes de contexto conhecem o ambiente no qual estão sendo utilizadas e tomam decisões de acordo com mudanças no seu próprio contexto de interesse. Tais sistemas reagem a ações executadas por outras entidades, podendo essas ser pessoas, objetos ou até mesmo outros sistemas, que modifiquem o meio (LOPES et al., 2012).

O contexto pode ser tudo que é significativo para um sistema, tudo que ocorre em um determinado ambiente, tudo que é relevante às aplicações do mesmo.

Alguns pesquisadores, com o intuito de limitar a abrangência desse conceito, propuseram definições referentes ao mesmo. A seguir, duas visões bastante conhecidas referentes a contexto são apresentadas.

Como referência clássica na área, (SCHILIT, 1995) (SCHILIT; ADAMS; WANT, 1994) dividem o contexto em três categorias. A primeira denominada **Computacional** compreende conectividade de rede, custos de comunicação, largura de banda e recursos disponíveis como impressoras, processadores e memória; a segunda entitulada **do Usuário** trata do perfil do usuário, localização, pessoas próximas a ele, humor e outros; e a terceira chamada de **Físico** compreende aspectos de luminosidade, níveis de barulhos, condições do trânsito e temperatura.

A definição mais aceita frequentemente na computação ubíqua para contexto é a citação histórica de (DEY; ABOWD; SALBER, 2001): *"Contexto é qualquer informação que pode ser usada para caracterizar uma situação de uma entidade. Uma entidade é uma pessoa, um lugar, ou um objeto que é considerado relevante para a interação entre um usuário e uma aplicação, incluindo o próprio usuário e a própria aplicação."* Destaca que os contextos mais relevantes para um ambiente computacional são: a localização, a identidade, o tempo e a atividade de uma entidade, ou seja, a enumeração de exemplos de contexto ainda é bastante usada na literatura.

A construção do suporte à consciência de contexto para as aplicações apresenta inúmeros desafios, dentre eles (VENECIAN, 2010):

(i) a classificação do contexto;
(ii) as dimensões das informações de contexto;
(iii) a representação de um modelo semântico formal de contexto;
(iv) o processamento e interpretação das informações de contexto adquiridas;
(v) a disseminação do contexto a entidades interessadas de forma distribuída e no momento oportuno;
(vi) o tratamento da qualidade da informação contextual.

A seguir é feita uma discussão dos vários aspectos pertinentes a estes desafios para o suporte à consciência de contexto.

2.2.1 Classificação do Contexto

O contexto é uma construção dinâmica e, embora em algumas situações o contexto seja relativamente estável e previsível, existem muitas outras onde isso não acontece. Na maioria dos casos é bastante complexo para o projetista de uma aplicação consciente ao contexto identificar o conjunto de todos os estados contextuais que podem existir na aplicação, bem como definir que ações devem ser executadas para os diferentes estados.

Algumas classificações para as informações contextuais foram propostas na literatura com o propósito de apoiar a identificação dos elementos de contexto. Uma dessas classificações divide as informações contextuais em dois tipos (WANG et al., 2004):

1. Contexto primário, básico ou de baixo nível:

   Indica elementos contextuais que podem ser percebidos, automaticamente, por sensores físicos ou lógicos (localização, tempo, condições ambientais, etc).

2. Contexto complexo ou de alto nível:

   Refere-se a informações de contexto fornecidas pelo próprio usuário ou deduzidas, por motores de inferência, a partir de um conjunto de informações de baixo nível (situação do indivíduo, situações e/ou atividades sociais).

Por sua vez, (GREENBERG, 2001) indica que o contexto varia em 5 dimensões: (i) período de tempo, (ii) episódios de uso anteriores conhecidos pelo usuário, (iii) estado das interações sociais, (iv) mudanças nos objetivos internos, e (v) influências do local onde a pessoa se encontra.

Outra classificação separa o contexto em porções estáticas e porções dinâmicas (GAUVIN; BOURRY-BRISSET; AUGER, 2004). A porção estática inclui os parâmetros externos, gerais e relativamente fixos, relacionados ao trabalho do usuário. A porção dinâmica é composta por uma memória dinâmica das ações do usuário, continuamente atualizada quando mudanças, eventos, atividades ou padrões aprendidos de ações ocorrem durante a execução do trabalho.

Korpipaa et al. (KORPIPAA et al., 2003) definiram alguns critérios para a identificação do que escolher como elemento contextual ao construir uma aplicação consciente ao contexto: (i) habilidade para descrever propriedades úteis do mundo real; (ii) habilidade para inferência de contextos complexos; (iii) facilidade ou viabilidade de ser medido ou reconhecido, automaticamente, de forma o mais precisa e não ambígua possível.

Rosa et al. (ROSA; BORGES; SANTORO, 2003) propõem um *framework* conceitual de contexto para sistemas colaborativos que classifica as informações em cinco categorias principais: (i) informações sobre as pessoas e os grupos; (ii) informações sobre tarefas agendadas; (iii) informações sobre o relacionamento entre pessoas e tarefas; (iv) informações sobre o ambiente onde ocorre a interação; e (v) informações sobre tarefas e atividades concluídas.

De acordo com (HENRICKSEN; INDULSKA; RAKOTONIRAINY, 2003), a natureza da informação contextual deve ser levada em conta ao se construir sistemas ubíquos e de computação consciente de contexto. Eis alguns pontos que cabe destacar: (i) características temporais da informação contextual; (ii) informação contextual é imperfeita; e (iii) o contexto tem representações alternativas.

2.2.2 Dimensões das Informações de Contexto

É usual aplicações conscientes do contexto utilizarem as facilidades do sistema de localização GPS (*Global Positioning System*). No entanto, existem outras informações de contexto além de localização e identificação de pessoas e objetos.

Muitos dos sistemas conscientes de contexto não incorpora várias das informações disponíveis em um ambiente, como noções de tempo, histórico e dados de outros usuários.

A representação chamada *framework* 5W+1H (GUTWIN; GREENBERG, 2002) identifica seis questões básicas que devem ser respondidas quando se deseja auxiliar um indivíduo a compreender algo do qual não tem conhecimento prévio. Informações de percepção (consciência de contexto), portanto, são respostas a essas seis perguntas fundamentais, como descritas a seguir na

Tabela 1 (SANTOS V., 2011).

Tabela 1: *Framework* 5W+1H

Quem (*Who*)	Informação de presença e disponibilidade dos indivíduos no grupo, e de identificação dos participantes envolvidos num evento ou numa ação.
O quê (*What*)	Informação sobre a ocorrência de um evento de interesse ao grupo.
Onde (*Where*)	Informação espacial, de localização, o local onde o evento ocorreu.
Quando (*When*)	Informação temporal sobre o evento, o momento em que o evento ocorreu.
Como (*How*)	Informação sobre a maneira como o evento ocorreu.
Por que (*Why*)	Informação subjetiva sobre as intenções e motivações que levaram à ocorrência do evento.

Em combinação com as características de aplicações conscientes do contexto, os trabalhos (ABOWD; MYNATT, 2000) (COSTA; YAMIN; GEYER, 2008) discutem a utilização destas dimensões semânticas de informações de contexto para auxiliar projetistas e desenvolvedores na especificação, na modelagem e na estruturação de informações de contexto de suas aplicações.

2.2.3 Obtenção de Dados Contextuais

Está associada com a forma na qual as informações contextuais são obtidas, podendo ser (MOSTEFAOUI; ROCHA; BREZILLON, 2004): **Sentida** - este tipo de informação pode ser adquirida do ambiente por meio de sensores (temperatura, nível de ruído, dispositivos presentes); **Derivada** - este é o tipo de informação que pode ser obtida em tempo de execução. Por exemplo, é possível calcular a idade de uma pessoa baseada na sua data de nascimento; **Provida** - informação que é explicitamente fornecida à aplicação. Por exemplo, os dados cadastrais de um usuário que sao fornecidos diretamente à aplicação por meio de um formulário.

2.2.4 Modelagem do Contexto

O desenvolvimento de aplicações conscientes do contexto é uma tarefa complexa, tornando o uso das técnicas de modelagem extremamente necessárias. Nesta perspectiva, continua uma frente de pesquisa em aberto, a concepção de arquiteturas de software que deem suporte para a modelagem das informações contextuais, bem como simplifiquem o desenvolvimento de aplicações conscientes do contexto (PERNAS, 2012).

Existem inúmeras abordagens para modelar, dentre as quais pode-se

ressaltar (STRANG; LINNHOFF-POPIEN, 2004):

- Modelos com métodos de marcação: seguem uma estrutura hierárquica de marcação com atributos e conteúdo. Em geral, utilizam-se de linguagens de marcação derivadas da Standard Generic Markup Language.
- Modelos chave-valor: utilizam um modelo simples de atributo e valor, sendo fáceis de gerenciar, contudo têm pouco poder de expressão.
- Modelos gráficos: baseados em notações gráficas, em geral são derivados de adaptações e extensões de modelos gráficos já difundidos, como UML, ORM ou o modelo Entidade Relacionamento.
- Modelos orientados a objetos: esta abordagem tem a intenção de aplicar os principais benefícios do modelo orientado a objetos, notadamente, encapsulamento e reusabilidade, à modelagem de contexto. Nesses casos, o acesso às informações contextuais é feito somente através de interfaces bem definidas.
- Modelos baseados em lógica: define-se o contexto de modo que se possa inferir expressões ou fatos a partir de um conjunto de outros fatos e expressões. Em geral, este modelo possui um alto grau de formalismo.
- Modelos baseados em ontologias: uma ontologia é uma especificação de uma conceituação, isto é, uma descrição de conceitos e relações que existem entre eles, em um domínio de interesse. Nesse modelo o contexto é modelado em ontologias, construindo uma base de conhecimento do contexto.

2.2.5 Interpretação em Sistemas Conscientes do Contexto

A interpretação de contexto pode ser entendida como o conjunto de métodos e processos que realizam a abstração, o mapeamento, a manipulação, a agregação, a derivação, a inferência e demais ações sobre as informações contextuais, com o propósito de facilitar o entendimento de um determinado contexto pelas aplicações e auxiliar na tomada de decisões. O processo de interpretação de contexto consiste na manipulação e refinamento das informações contextuais de um ambiente.

O processo de interpretação de contexto pode ser bastante simples, como derivar o nome de uma rua a partir de suas coordenadas geográficas, ou bastante complexo e oneroso, como inferir o humor de um usuário baseado em seu perfil e na atividade em que ele está realizando. Além disso, o ambiente da computação ubíqua, é extremamente dinâmico e complexo. As informações

contextuais podem estar espalhadas e distribuídas em qualquer lugar e com alto grau de mobilidade. Essa complexidade faz com que haja a necessidade de um suporte computacional às aplicações, de maneira a auxiliá-las na realização de interpretações de contextos. Tais atividades onerosas devem ser abstraídas das aplicações e o módulo Interpretador de Contexto torna-se, portanto, um componente essencial em uma plataforma de suporte a tais aplicações. Ele deve ser capaz de obter e prover informação contextual em diferentes níveis de abstração, conforme o desejo do usuário e de suas aplicações. Uma aplicação pode desejar tanto informações mais brutas, de mais baixo nível ou informações mais abstratas e elaboradas, de mais alto nível, provenientes de um processo de refinamento e interpretação (COSTA; YAMIN; GEYER, 2008).

2.2.6 Raciocínio em Sistemas Conscientes do Contexto

Um dos principais problemas na utilização de informações contextuais é como obter contexto realmente significativo para quem precisa utilizá-las, a partir de um conjunto de informações dispersas e desconexas, obtidas por mecanismos heterogêneos de aquisição. Para isso, funcionalidades de processamento e raciocínio sobre a informação contextual devem ser disponibilizadas. Questões como tratamento da incerteza devem ser consideradas, pois como o contexto evolui bastante com o tempo é difícil inferir com precisão qual é de fato o contexto atual da situação.

Os termos raciocínio e inferência são geralmente utilizados para indicar qualquer processo pelo qual conclusões são alcançadas (RUSSELL S., 2003). O projeto e implementação de um mecanismo para raciocínio de contextos pode variar bastante a depender do tipo do conhecimento contextual envolvido. Idealmente, o processamento do contexto deve ser implementado separadamente do comportamento do sistema e não embutido no código da aplicação (BELOTTI et al., 2004).

O raciocínio é utilizado para verificar a consistência do contexto e para inferir contexto implícito (alto nível), a partir de contextos explícitos (baixo nível) (WANG et al., 2004). A consistência é necessária pois, muitas vezes, a informação contextual adquirida automaticamente pode apresentar erros e ambigüidades. Por exemplo, um sensor de presença pode detectar o celular de um usuário em sua casa e deduzir que o mesmo está em casa. Porém, um outro sensor de presença baseado em câmeras detecta a presença do usuário em seu escritório. Essas duas informações são conflitantes e precisam ser resolvidas.

2.2.7 Persistência em Sistemas Conscientes do Contexto

A necessidade de manter o histórico de informações de contexto é um requisito ligado à aquisição de informações de contexto bem como à disponibilidade contínua dos componentes de captura de informações de contexto. Um histórico de contexto pode ser utilizado para estabelecer tendências e predizer valores futuros de informações de contexto. Sem o armazenamento dessas informações, esse tipo de análise não é possível de ser realizada.

2.3 *Middleware* EXEHDA

Este seção registra, de modo resumido, a revisão feita sobre o *middleware* EXEHDA e seus subsistemas, proposto inicialmente em (YAMIN, 2004).

O EXEHDA é um *middleware* adaptativo ao contexto e baseado em serviços que visa criar e gerenciar um ambiente ubíquo, bem como promover a execução, sob este ambiente, das aplicações que expressam a semântica siga-me. Estas aplicações são distribuídas, móveis e adaptativas ao contexto em que seu processamento ocorre, estando disponíveis a partir de qualquer lugar, todo o tempo (LOPES et al., 2012a).

2.3.1 Aspectos funcionais e arquiteturais

O *middleware* EXEHDA tem como objetivo definir a arquitetura para um ambiente de execução destinado às aplicações da computação ubíqua, no qual as condições de contexto são pró-ativamente monitoradas e o suporte à execução deve permitir que tanto a aplicação como ele próprio utilizem estas informações na gerência da adaptação de seus aspectos funcionais e não-funcionais. Entende-se por adaptação funcional aquela que implica a modificação do código sendo executado. Por sua vez, adaptação não funcional é aquela que atua sobre a gerência da execução distribuída. Também a premissa siga-me das aplicações ubíquas deverá ser suportada, garantindo a execução da aplicação do usuário em qualquer tempo, lugar e dispositivo (LOPES et al., 2012b).

As aplicações alvo são distribuídas, adaptativas ao contexto em que executam e compreendem a mobilidade lógica e a física. Na perspectiva do EXEHDA, entende-se por mobilidade lógica a movimentação entre dispositivos de artefatos de software e seu contexto, e por mobilidade física o deslocamento do usuário, portando ou não seu dispositivo (ISAM, 2009).

(i) Arquitetura de Software

A arquitetura de software do *middleware* EXEHDA é apresentada na Figura 1. Os principais requisitos que o EXEHDA deve atender são: (i) gerenciar tanto aspectos não funcionais como funcionais da aplicação, de modo independente; (ii) dar suporte à adaptação dinâmica de aplicações; (iii) disponibilizar mecanismos para obter e tratar informações de contexto; (iv) empregar informações de contexto na tomada de decisões; (iv) decidir as ações adaptativas de forma colaborativa com a aplicação; e (v) disponibilizar a semântica siga-me, permitindo ao usuário iniciar as aplicações e acessar dados a partir de qualquer lugar, e executar as aplicações continuamente mesmo em deslocamento.

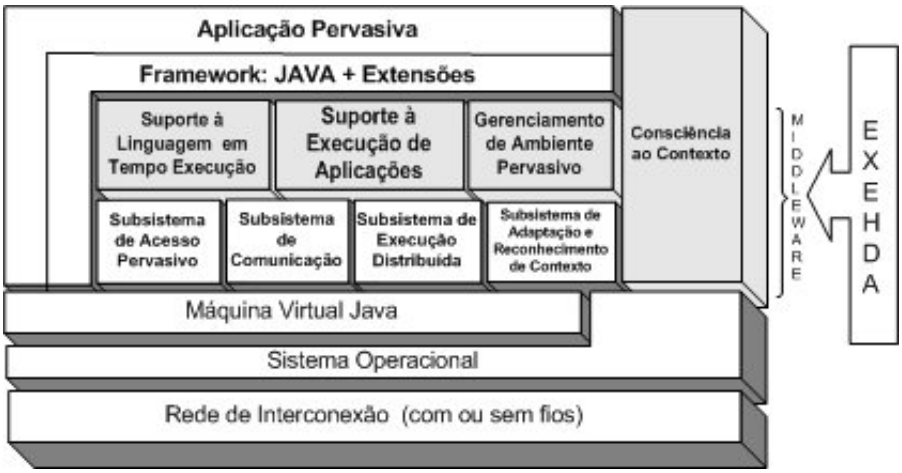


Figura 1: Arquitetura de Software do Middleware EXEHDA (YAMIN, 2004)

(ii) Ambiente Ubíquo Disponibilizado

O ambiente ubíquo é equivalente ao ambiente computacional onde recursos e serviços são gerenciados pelo EXEHDA com o propósito de atender os requisitos da computação ubíqua. A composição deste ambiente envolve tanto os dispositivos dos usuários, como os equipamentos da infraestrutura de suporte, todos instanciados pelo seu respectivo perfil de execução do *middleware*. A integração dos cenários da computação em grade, da computação móvel e da computação consciente de contexto é mapeada em uma organização composta pelo conjunto de células de execução do EXEHDA, conforme pode ser observado na Figura 2 (ISAM, 2009).

O meio físico sobre o qual o ambiente ubíquo é definido por uma rede infraestruturada, cuja composição final pode ser alterada pela agregação dinâmica de nodos móveis. Os recursos da infraestrutura física são mapeados para três abstrações básicas, as quais são empregadas na composição do

ambiente ubíquo (YAMIN, 2004):

- EXEHDAcels: indica a área de atuação de uma EXEHDAbas, e é composta por esta e por EXEHDAnodos. Os principais aspectos considerados na definição da abrangência de uma célula são: o escopo institucional, a proximidade geográfica e o custo de comunicação;
- EXEHDAbase: é o ponto de convergência para os EXEHDAnodos. É responsável por todos os serviços básicos do ambiente ubíquo e, embora constitua uma referência lógica única, seus serviços, sobretudo por aspectos de escalabilidade, poderão estar distribuídos entre os vários dispositivos;
- EXEHDAnodo: são os dispositivos de processamento disponíveis no ambiente ubíquo, sendo responsáveis pela execução das aplicações. Um subcaso deste tipo de recurso é o EXEHDAnodo móvel. São os nodos do sistema com elevada portabilidade, tipicamente dotados de interface de rede para operação sem fio e, neste caso, integram a célula a qual seu ponto-de-acesso está subordinado. São funcionalmente análogos aos EXEHDAnodos, porém eventualmente com uma capacidade mais limitada (por exemplo, PDAs).

O ambiente ubíquo é formado por dispositivos multi-institucionais, o que gera a necessidade de adotar procedimentos de gerência iguais aos utilizados em ambientes de grade computacional (*Grid Computing*) (YAMIN, 2004). O gerenciamento da organização celular do ambiente ubíquo resguarda a autonomia das instituições envolvidas. Apesar de não contemplar mecanismos de gerência específicos para recursos especializados como impressoras, *scanners*, dentre outros o EXEHDA permite a catalogação de tais recursos como integrantes de uma determinada célula do ambiente ubíquo, tornando-os, desta forma, passíveis de serem localizados dinamicamente e serem utilizados pelas aplicações ubíquas.

(iii) Composição de Serviços

A prerrogativa de operação em um ambiente altamente heterogêneo, onde não só o *hardware* exibe capacidades variadas de processamento e memória, mas também as bibliotecas de software disponíveis em cada dispositivo, motivaram a adoção de uma abordagem na qual um núcleo mínimo do *middleware* tem suas funcionalidades estendidas por serviços carregados sob demanda. Esta organização retrata um padrão de projeto referenciado na literatura como microkernel. Some-se a isto o fato de que esta carga sob

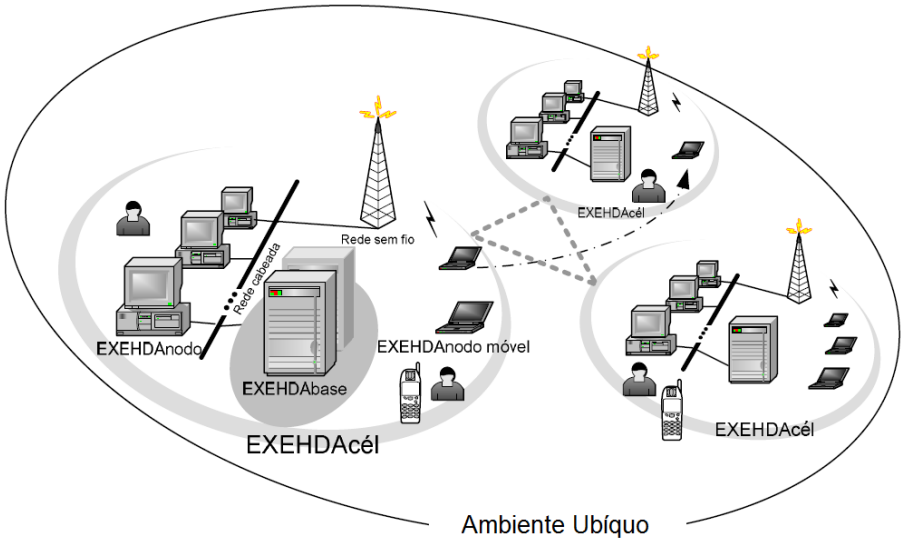


Figura 2: Ambiente ubíquo suprido pelo EXEHDA (YAMIN, 2004)

demanda tem perfil adaptativo. Deste modo, poderá ser utilizada versão de um determinado serviço, melhor sintonizada às características do dispositivo em questão. Isto é possível porque, na modelagem do EXEHDA, os serviços estão definidos por sua interface, e não pela sua implementação propriamente dita.

A contraproposta à estratégia microkernel de um único binário monolítico, cujas funcionalidades cobrissem todas as combinações de necessidades das aplicações e dispositivos, se mostra inviável na computação ubíqua, cujo ambiente computacional apresenta uma alta heterogeneidade de recursos de processamento.

Entretanto, o requisito do *middleware* em manter-se operacional durante os períodos de desconexão planejada motivou, além da concepção de primitivas de comunicação adequadas a esta situação, a separação dos serviços que implementam operações de natureza distribuída em instâncias locais ao EXEHDAnodo (instância nodal), e instâncias locais a EXEHDAbase (instância celular). Neste sentido, o relacionamento entre instância de nodo e intância celular assemelha-se à estratégia de *proxies*, enquanto que o relacionamento entre instâncias celulares assume um caráter P2P (*Peer-to-Peer*). A abordagem P2P nas operações intercelulares vai ao encontro do requisito de escalabilidade. O *middleware* EXEHDA é formado por quatro subsistemas: Subsistema de Execução Distribuída, Subsistema de Comunicação, Subsistema de Acesso Ubíquo e Subsistema de Reconhecimento de Contexto e Adaptação, cuja organização pode ser observada na Figura 3.

Sendo assim, os componentes da aplicação em execução em determinado dispositivo podem permanecer operacionais desde que, para satisfação de uma dada requisição pelo *middleware*, o acesso a um recurso externo ao dispositivo

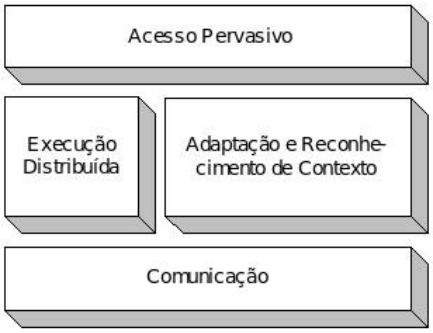


Figura 3: Organização dos subsistemas do EXEHDA (YAMIN, 2004)

seja prescindível. Por outro lado, a instância celular, em execução na base da célula, provê uma referência para os outros recursos, no caso da realização de operações que requeiram coordenação distribuída. Portanto, observe-se que a EXEHDAbase é, por definição, uma entidade estável dentro da EXEHDAcel, permitindo que os demais integrantes (recursos) da célula tenham um caráter mais dinâmico com relação a sua disponibilidade (presença estável) na célula.

Os sistemas distribuídos modernos constituem um ambiente bastante complexo, formado por (i) sistemas legados; (ii) sistemas fechados provenientes de empresas; e (iii) sistemas desenvolvidos pelos grupos de pesquisa interessados. Estes sistemas caracterizam-se pela distribuição e heterogeneidade quanto ao hardware e software empregados, situação típica da computação ubíqua.

Nesse cenário, um problema enfrentado com frequência consiste na integração de sistemas, com o intuito de sincronizar seus dados e/ou desenvolver novas funcionalidade.

2.4 Suporte a Interoperabilidade em Ambientes Distribuídos

O Sistema de Consciência de Contexto do EXEHDA, na perspectiva desta dissertação de mestrado, é um sistema distribuído de elevada escalabilidade e heterogeneidade. Esta seção tem como objetivo sistematizar conceitos para concepção da arquitetura de software proposta no Capítulo 3.

De modo geral, a literatura da área de engenharia de software indica que os esforços de integração costumam superar os de desenvolvimento. Nesse sentido, o uso de ferramentas que facilitem a integração mostram-se oportunas (MENGE, 2007).

Existem diversas ferramentas para suporte ao processo de integração, desde bibliotecas de classes até completas infraestruturas para comunicação e integração baseadas em *Enterprise Service Bus* (ESB) (MENGE, 2007).

A definição da uma ferramenta ESB adequada às necessidades de um projeto

deve considerar as diferentes funcionalidades a serem integradas. Ressalte-se que não existe uma padronização de nomenclatura a ser utilizada (FRANTZ, 2008).

Assim, tendo como referência um estudo desenvolvido em (FRANTZ, 2008), a próxima seção apresenta as principais características de ferramentas ESB, sistematizando conceitos que serão utilizados quando da implementação da interoperabilidade dos equipamentos do Serviço de Consciência do EXEHDA.

Uma abordagem para suporte a interoperabilidade em ambientes distribuídos corresponde ao emprego de uma arquitetura ESB (*Enterprise Service Bus*), a qual permite integrar sistemas, usando troca de mensagens. O objetivo central de um ESB é realizar o processamento de uma mensagem recebida de um sistema, transformando-a em um formato padrão que possa ser entendido por outro sistema. Os sistemas não precisam ter sido planejados para atuar de uma forma integrada ou mesmo utilizar um padrão comum, mas eles irão valer-se de uma arquitetura de software que possibilita uma comunicação padrão, conforme ilustra a Figura 4.

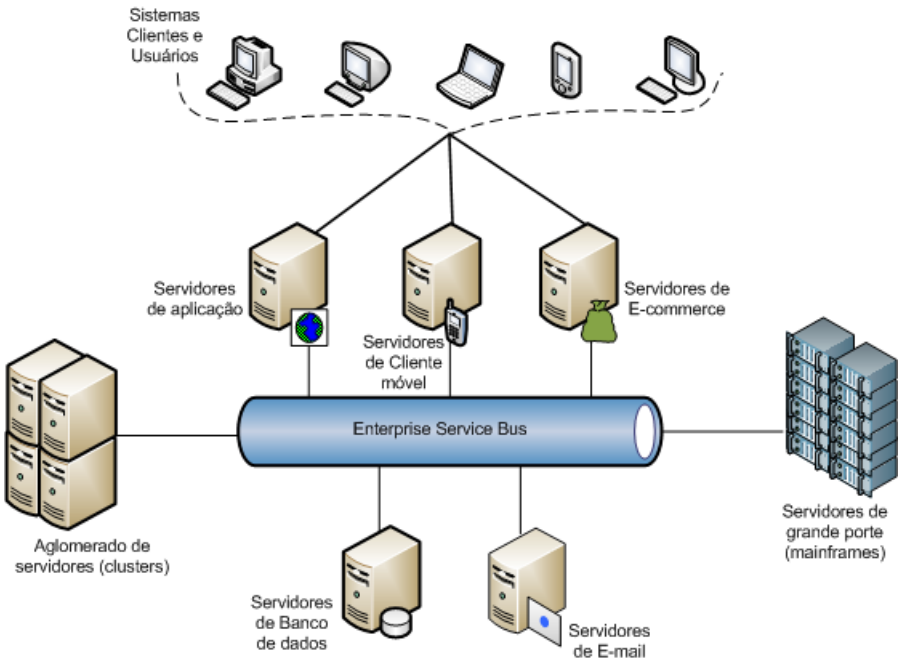


Figura 4: Arquitetura típica ESB (adaptado de (MICROSOFT, 2013))

Considerando o estudo realizado por (FRANTZ, 2008), que considerou as ferramentas ESB: CAMEL (CAMEL, 2011), SPRING (SPRING, 2011) e BIZTALK (BIZTALK, 2011) são destacadas as características resumidas a seguir.

1. Contexto: diz respeito ao contexto de integração no qual a ferramenta pode ser empregada. Nesse estudo são identificados dois contextos de uso: (i) *Enterprise Application Integration* (EAI), integração de aplicações com

objetivo de sincronizar dados e/ou implementar novas funcionalidades; e (ii) *Business to Business Integration* (B2BI), cujo objetivo é integrar aplicações de diferentes instituições para implementar processos interinstitucionais.

2. Padrão Arquitetural: corresponde ao padrão empregado na concepção da arquitetura de software da ferramenta. Usualmente o padrão *pipes* e *filters* é adotado na integração de sistemas. Nesse padrão, um evento dispara uma sequência de passos de processamento, cada um responsável por uma função específica bem delimitada.

3. Nível de abstração: o uso de modelos independentes de plataforma (*Platform Independent Model* - PIM) viabiliza o projeto de soluções que não dependem fortemente da tecnologia empregada. As ferramentas devem ser capazes de traduzir o modelo PIM em um modelo específico da plataforma (*Platform Specific Model* - PSM).

4. Tipo de modelo: os modelos podem ser operacionais (O) ou declarativos (D). Os modelos operacionais contemplam uma biblioteca de classes que podem ser combinadas para criação de soluções de integração. Por sua vez, os modelos declarativos permitem um nível de abstração elevado, utilizando linguagens específicas de domínio (*Domain Specific Language* - DSL) com representação gráfica ou XML, que é traduzida automaticamente para código executável. Outra forma é através de arquivos de configuração XML, que são interpretados por um mecanismo de tratamento.

5. Tipo de DSL: as linguagens específicas de domínio podem ser: (i) internas, consiste em uma linguagem que é definida usando como base a sintaxe de uma linguagem de propósito geral; e (ii) externas, não seguem a sintaxe de qualquer linguagem de propósito geral.

6. Transações: as transações podem ser de dois tipos: (i) *short term* - ST, onde as ações que requerem mudança de estado são executadas quando todos os componentes envolvidos estão disponíveis para essa execução; e (ii) *long term* - LT, executam as ações conforme possível e caso alguma falhe são executadas ações de compensação para desfazer os efeitos das que já foram realizadas.

A arquitetura das ferramentas, via de regra, segue o estilo *pipes* e *filters* devido as suas características de encapsulamento, alta coesão, recombinação e reuso dos dados. Por sua vez, o emprego de transações permite projetar soluções robustas, capazes de minimizar as falhas que podem ocorrer durante a integração.

2.5 Considerações sobre o capítulo

Neste capítulo foram abordados aspectos sobre a Computação Ubíqua, sua origem, conceitos e características considerados mais importantes, assim como, alguns de seus projetos históricos. Este capítulo tratou também da consciência de contexto e seus principais conceitos. Na sequência foi descrito o *middleware* EXEHDA e seus subsistemas e por fim, uma discussão sobre interoperabilidade em ambientes distribuídos também foi comtemplada.

3 EXEHDA-UC: VISÃO GERAL E MODELAGEM

Este capítulo apresenta os aspectos de concepção do EXEHDA-UC sendo destacadas as principais decisões tomadas no desenvolvimento da pesquisa.

O trabalho desenvolvido contribui com o Subsistema de Adaptação e Reconhecimento de Contexto do EXEHDA propondo para o serviço de consciência de contexto uma arquitetura, que de forma distribuída, ofereça suporte às etapas de aquisição, armazenamento e processamento de informações contextuais, bem como os decorrentes procedimentos de atuação sobre o meio.

A concepção do EXEHDA-UC compreende dois tipos principais de servidores: Servidor de Borda e Servidor de Contexto. O Servidor de Borda enquanto responsável pela interação com o meio através de sensores e atuadores, e o Servidor de Contexto atuando no armazenamento e processamento das informações contextuais.

A arquitetura proposta provê comunicação (i) entre os Servidores de Borda e o Servidor de Contexto; (ii) entre os Servidores de Contexto localizados em células diferentes; e (iii) com outros serviços do *middleware* ou aplicações. Um visão geral desta arquitetura é apresentada na Figura 5, na qual os componentes da mesma são mapeados sobre o ambiente ubíquo gerenciado pelo EXEHDA. Uma descrição das abstrações básicas empregadas na composição do ambiente ubíquo encontra-se na Seção 2.3.1.

O objetivo geral é fornecer uma solução capaz de adquirir de forma autonômica informações contextuais, bem como permitir atuação remota sobre o meio através de dispositivos eletromecânicos.

No meio ubíquo, as diferentes informações estão espalhadas pelo ambiente. Para que o usuário tenha acesso a essas informações é necessário que elas sejam colhidas do meio através de dispositivos sensores geograficamente distribuídos. Estas informações também devem ser registradas de forma conveniente para seu futuro processamento tanto pelo *middleware* como pelas aplicações.

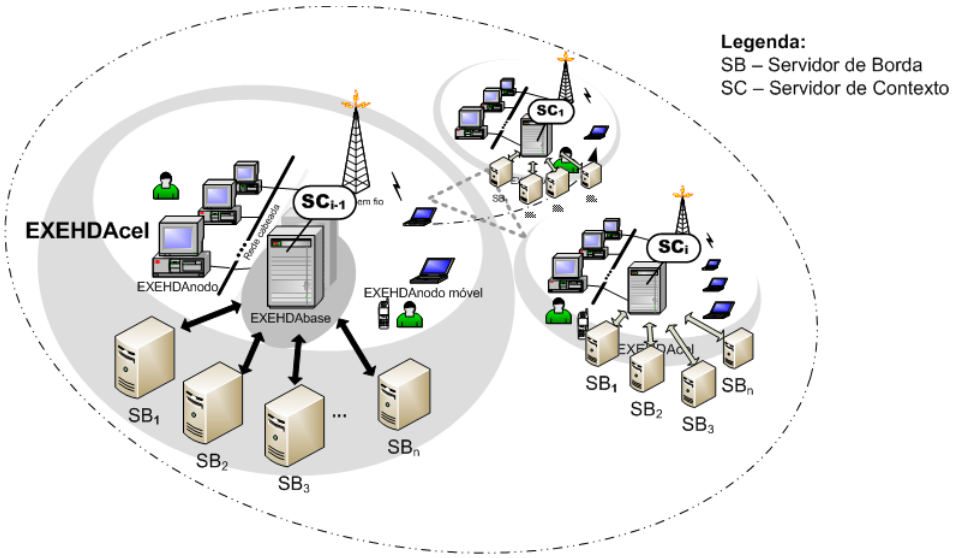


Figura 5: EXEHDA-UC: Visão Geral

Além de verificar informações sobre o meio, pode ser de interesse do usuário alterar características físicas do ambiente através do acionamento de dispositivos em geral. Por exemplo: um pesquisador controla remotamente a temperatura de uma sala onde realiza experimentos e recebe a informação de que a temperatura está abaixo de um determinado valor e deseja, então, acionar o sistema de aquecimento. Nesse caso, o EXEHDA-UC deve prover funcionalidades para receber o comando para ligar o sistema de aquecimento. Deste modo, os servidores do EXEHDA-UC não apenas coletam e publicam informações, como também podem ser empregados para prover atuação sobre o meio.

A descrição do EXEHDA-UC, feita a seguir, está organizada a partir de seus servidores e respectivas funcionalidades, sendo realizadas as necessárias associações entre os mesmos, e com os outros serviços do *middleware* EXEHDA.

3.1 Servidor de Borda

A arquitetura proposta para o Servidor de Borda contempla três módulos, um destinado a tratar a rede de sensores; outro para efetuar as publicações e outro para tratar a rede de atuadores. Uma visão da arquitetuta do Servidor de Borda está disponível na Figura 6. O Servidor de Borda possui dois fluxos principais de funcionamento, um proveniente da rede de sensores (Módulo de Sensoriamento); e outro iniciado a partir do Servidor de Contexto em direção a rede de atuadores (Módulo de Atuação).

Os componentes da arquitetura serão descritos na continuidade, com uma discussão de suas funcionalidades considerando a arquitetura como um todo.

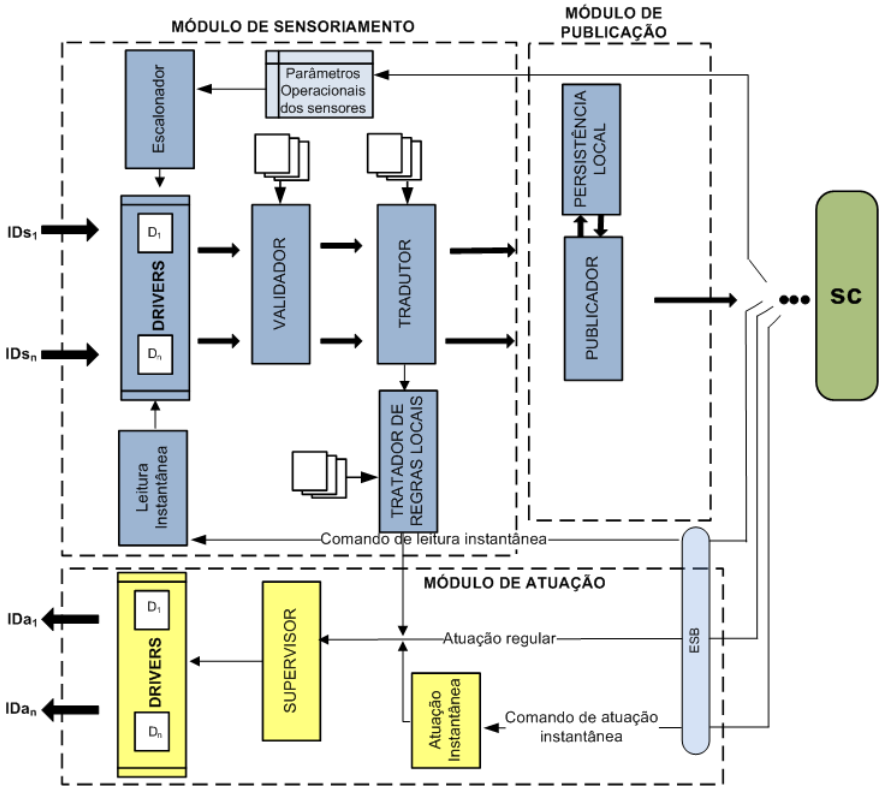


Figura 6: Arquitetura do Servidor de Borda

3.1.1 Módulo de Sensoriamento

O Módulo de Sensoriamento provê o tratamento de uma rede de sensores permitindo uma individualização de processamento por sensor. Este tratamento é responsável por aspectos desde gerência física (interfaces, frequência de leitura, etc.) até normalização computacional (validação, tradução, etc.) dos valores aquisitados.

Nesta seção serão descritos os componentes do módulo de sensoriamento, os quais foram concebidos tendo como referência os objetivos apresentados na Seção 1.2.

(i) Parâmetros Operacionais dos Sensores

O componente Parâmetros Operacionais dos Sensores especifica o *driver* a ser utilizado para leitura dos diferentes sensores, bem como a agenda de aquisição de dados a ser praticada. Esta agenda trata individualmente os sensores, permitindo a especificação de leituras periódicas e/ou atreladas a datas e/ou horários específicos.

Por ser um componente definido pelo usuário, o mesmo é editado no Servidor de Contexto, aproveitando as especificações disponíveis no repositório de informações contextuais. Uma vez pronto o arquivo YAML (YAML, 2010) correspondente é transferido para o Servidor de Borda.

Um exemplo do arquivo YAML é apresentado na Figura 7, que pode ser utilizado pelos usuários na configuração do EXEHDA-UC.

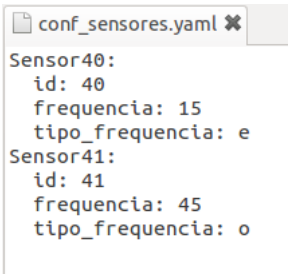


Figura 7: Parâmetros operacionais dos sensores (YAML)

(ii) Escalonador

O componente Escalonador, a partir dos Parâmetros Operacionais dos Sensores, dispara leituras considerando o interesse do usuário.

O Escalonador emprega o relógio do equipamento para disparo dos seus procedimentos, sendo possível especificar a frequência com que estes disparos ocorrem. Por exemplo, no estudo de caso desenvolvido nesta dissertação (vide Capítulo 5) ocorre uma ativação do Escalonador a cada minuto.

A cada ativação é disparado um parser YAML para interpretação de qual sensor deverá ser lido e ativado o *driver* correspondente.

(iii) Drivers

O *driver* do sensor é responsável pela aquisição do valor referente a grandeza física medida pelo sensor. É usual cada tipo de sensor possuir uma forma específica de acesso para obtenção dos dados.

A estratégia de encapsular a operação do *driver* do sensor tem por objetivo evitar que as diferenças operacionais de cada *driver* se projetem nas outras camadas de software da arquitetura. Dentre outros aspectos este encapsulamento contribui para a manutenção (troca de sensores, atualização de *drivers* por parte do fabricante, etc.).

O *driver* de cada sensor é individualizado por um código identificador. Isso permite com que um novo *driver* possa ser adicionado, ou um *driver* antigo seja alterado, sem a necessidade de outras especificações.

(iv) Validador

O componente Validador avalia se uma determinada coleta de dados sensorados deve ser efetivamente publicada ou não, empregando para isso critérios especificados pelo usuário.

Esses critérios são baseados em regras, por exemplo, uma regra pode estabelecer que somente devem ser publicados valores cuja variação seja superior em 5% da leitura anterior, ou publicar valores que estão em uma determinada faixa.

A regra a ser utilizada está identificada no componente Parâmetros Operacionais dos Sensores, a partir do ID do sensor (IDs).

(v) Tradutor

O componente Tradutor objetiva adequar o dado coletado à natureza da aplicação do usuário, deste modo faixas de temperatura podem ser convertidas em descrições do tipo: ”Alta”, ”Média”, ”Baixa”, através de uma regra procedural.

Este componente contribui para otimizar os volumes de dados transferidos entre os servidores, também aumentando a legibilidade dos mesmos.

(vi) Leitura Instantânea

O componente Leitura Instantânea foi incluido na arquitetura com o objetivo de permitir a leitura de um determinado sensor sob demanda das aplicações a qualquer momento. As requisições de leitura podem ser provenientes de aplicações de usuários ou de Servidores de Contexto como consequência da execução de regras.

Este componente emprega um barramento ESB para recepção assíncrona das solicitações e, a partir do ID do sensor, dispara o *driver* correspondente.

(vii) Tratador de Regras Locais

O componente Tratador de Regras Locais é responsável por tratar regras de contingência, com a intenção de evitar que os dispositivos envolvidos atinjam estados críticos. As mesmas irão atuar sobre o mecanismo de gerência da atuação ativando ou desativando equipamentos eletromecânicos (atuadores).

O componente Tratador é ativado sempre que ocorrer a leitura de um determinado sensor. Este componente adquire elevada importância quando a comunicação com o Servidor de Contexto for interrompida por problemas de infraestrutura de rede.

3.1.2 Módulo de Publicação

O Módulo de Publicação é responsável por coordenar o principal fluxo de dados entre os Servidores de Borda e o Servidor de Contexto, promovendo a publicação de todos os dados coletados e garantindo uma persistência local dos mesmos nos períodos que a publicação ficar inviabilizada.

(i) Publicador

O Publicador interopera com o Módulo de Aquisição do Servidor de Contexto. As submissões dos dados coletados acontecem empregando o barramento ESB do Módulo de Aquisição, de forma particularizada por aplicação.

Como procedimento de segurança as transações ocorrem sobre o protocolo HTTPS (*HyperText Transfer Protocol Secure*), o que garante que os dados serão transferidos criptografados, evitando acessos indevidos aos mesmos.

(ii) Persistência Local

Uma vez que todo o dado coletado deve ser publicado no Servidor de Contexto, é de fundamental importância que seja efetuado um armazenamento local das informações, para garantir que em caso de falha na comunicação, os dados sejam mantidos até que possam ser publicados.

Para isso, foi adotada uma estratégia de fila, onde são armazenadas todas as informações a serem publicadas, como o valor do sensor, data e hora de coleta, etc. Essa fila, por sua vez, é armazenada em disco no formato de um arquivo promovendo a persistência. O registro da data e hora da coleta é necessário para o gerenciamento da validade de uma informação de contexto no que diz respeito a possíveis atuações. Coletas de dados contextuais mais antigas são mantidas com intenção de registro histórico.

3.1.3 Módulo de Atuação

Nesta seção serão abordados os componentes que dizem respeito à gerência dos atuadores, sendo as funcionalidades dos componentes Atuação Instantânea, Supervisor e *Driver* do atuador descritas a seguir.

(i) Atuação Instantânea

A Atuação Instantânea é um componente com funcionalidade análoga ao serviço de leitura instantânea dos sensores, recebendo comandos assíncronos, a qualquer momento, através um barramento ESB (vide Figura 8 (iii)). Esses comandos são originários tanto do Servidor de Contexto como consequência da execução de uma regra, como de uma aplicação controlada pelo usuário. Os parâmetros empregados são o ID do atuador e os correspondentes padrões de operação (tempo de duração, potência de ativação, etc.), os quais são repassados ao componente Supervisor para tratamento.

(ii) Supervisor

O componente Supervisor aglutina comandos de atuação provenientes de três fontes distintas: da atuação regular; da atuação instantânea; e aqueles provenientes do Tratador de Regras Locais.

Uma vez recebidos os parâmetros para controle da atuação, o componente Supervisor tem autonomia para ativar o *driver* necessário para o atuador que está sendo gerenciado.

Com o intuito de identificar comportamentos anômalos no que diz respeito à gerencia dos atuadores, este componente conta com uma especificação por atuador (IDa) de quantas vezes no máximo para uma unidade de tempo é esperado que ocorram transições no estado do atuador. O objetivo com isto é identificar eventuais conflitos entre as regras no Servidor de Contexto e as regras de contingência no componente Tratador de Regras Locais, ou ainda, inconformidades entre comandos de atuação instantânea disparados pelo usuário e as regras ativas nos servidores.

Também é este componente que trata os parâmetros de acionamento dos atuadores, implementando através de *drivers* procedimentos de ativação e desativação, de regulação de potência operacional, tempo de validade do procedimento de atuação, dentre outros.

Este comportamento está sintetizado na Figura 8, na qual pode-se ver dois laços de controle: (i) um externo passando pelo Servidor de Contexto e regido pelas regras presentes neste, (ii) outro interno ao Servidor de Borda, o qual é regido pelas regras de contingência locais. Também se pode ver na Figura 8 (iii) o comando de atuação instantânea, o qual não é regido por regras mas sim disparado por iniciativa do usuário.

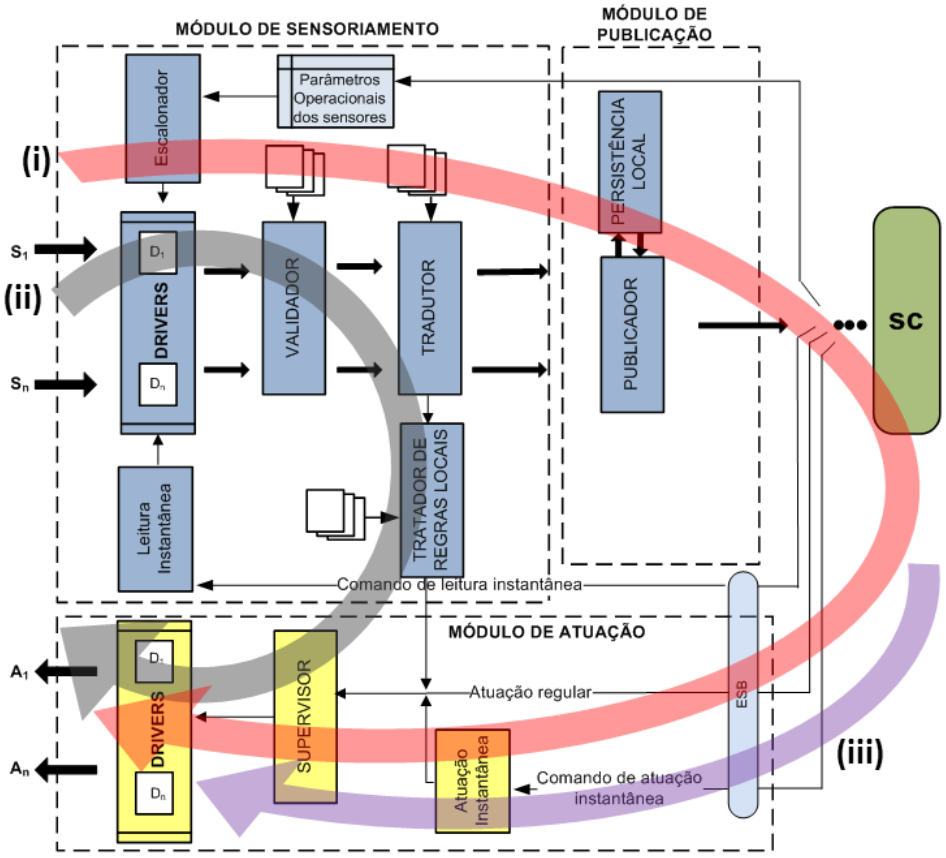


Figura 8: Visão geral do controle da atuação

(iii) Drivers

Os *drivers* dos atuadores tem objetivo similar aqueles dos sensores, ou seja, encapsulam os procedimentos específicos de cada atuador, na maioria das vezes empregando bibliotecas e/ou softwares disponibilizados por fabricantes.

Esta abordagem preserva a propagação de aspectos de implementação para as camadas superiores da arquitetura do Servidor de Borda.

3.2 Servidor de Contexto

O Servidor de Contexto foi organizado em seis módulos autônomos, que interoperam no provimento das funcionalidades necessárias ao Serviço de consciência de contexto do EXEHDA.

Uma visão geral do mesmo é ilustrada na Figura 9, na qual é caracterizada a relação com os Servidores de Borda, outros serviços do *middleware*, outros Servidores de Contexto Remotos e aplicações de usuários.

Cada um destes módulos é responsável por uma etapa da consciência de contexto, desde sua aquisição até o momento em que é armazenado e/ou repassado a quem o solicitou. A seguir serão descritos os módulos que o

compõem.

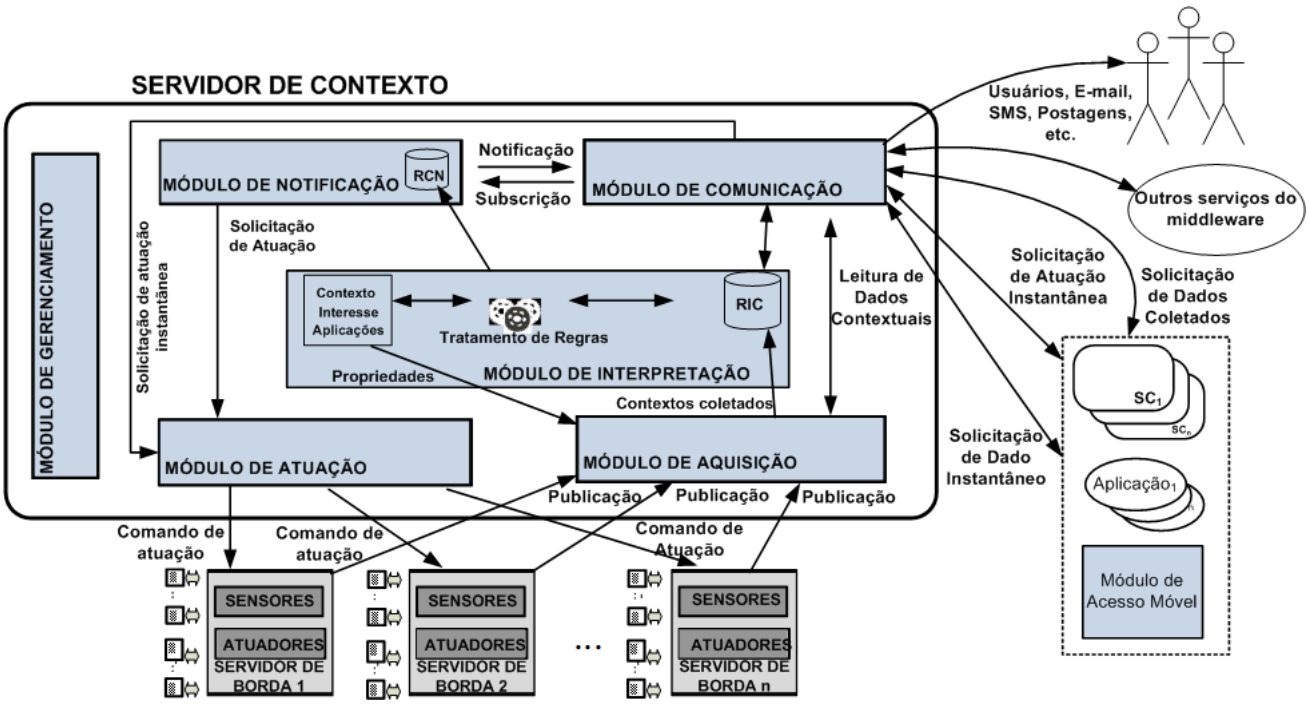


Figura 9: Arquitetura do Servidor de Contexto

3.2.1 Módulo de Aquisição

O Módulo de Aquisição é responsável por prover suporte à captura das informações contextuais, coletadas pelos servidores de borda considerando sensores lógicos (interfaces de software) e/ou hardware.

A publicação de dados de contexto no Módulo de Aquisição considera os seguintes parâmetros:

- o intervalo de tempo entre medições;
- a flutuação mínima do valor coletado para que aconteça a publicação;
- a inserção na faixa de valores que deverão ser publicados.

Este módulo apresenta um comportamento de servidor cujas funcionalidades são implementadas através de um barramento tipo ESB, permitindo que Servidores de Borda, sempre que ocorrerem variações significativas nos dados contextuais, possam publicá-los.

3.2.2 Módulo de Atuação

O Módulo de Atuação é responsável pelo controle (ativação, desativação e configuração) dos atuadores, após ser notificado pelos outros módulos do Servidor de Contexto.

O mesmo recebe o IDa do atuador envolvido e os parâmetros operacionais a serem utilizados e interopera com os Servidores de Borda para disparo dos dispositivos eletromecânicos pertinentes.

O Módulo de Atuação interage através do barramento ESB do Servidor de Borda. Mais especificamente, possui as seguintes atribuições:

- repassar ao Servidor de Borda os itens de configuração para ativação, desativação e configuração dos atuadores;
- notificar ao Servidor de Borda quais atuadores serão necessários para atender as demandas das aplicações em uso;
- submeter ao Servidor de Borda os comandos para leitura instantânea dos sensores;
- submeter ao Servidor de Borda os comandos de atuação regular e atuação instantânea.

De modo geral, o Módulo de Atuação é responsável por disparar no ambiente ubíquo ações que mudem o estado do meio, viabilizando o uso de serviços de consciência de contexto em aplicações de controle e automação.

3.2.3 Módulo de Interpretação

O Módulo de Interpretação de contexto tem como principal função realizar tarefas de manipulação e dedução das informações contextuais, utilizando para isto informações especificadas nos Contextos de Interesse das aplicações.

Este módulo foi objeto de estudo dos trabalhos (VENECIAN, 2010) e (WARKEN, 2010) os quais exploraram o uso de ontologias para suporte semântico quando do processamento das informações contextuais.

A arquitetura do EXEHDA-UC oferece suporte para que as funcionalidades do Módulo de Interpretação fossem mantidas operacionais.

O Módulo de Interpretação de contexto mantém um repositório de contextos coletados, onde são armazenados as informações contextuais obtidas pelo Módulo de Aquisição de contexto, provendo a possibilidade de processamento histórico dos contextos.

Os objetivos do Módulo de Interpretação incluem:

- Manter consistente o Repositório de Informação Contextuais (RIC) (vide Seção 3.2.4), gerenciar o raciocínio sobre as informações contextuais mantidas neste repositório, inferindo novos contextos a partir de regras lógicas de inferência e de fatos contextuais definidos no Contexto de Interesse das Aplicações;

- Atualizar o repositório contextual, de modo a manter um histórico. Esse histórico poderá servir como uma base de aprendizado que possibilite melhorar a inferência em interações futuras;

3.2.4 Repositório de Informações Contextuais

O Repositório de Informações Contextuais (RIC) foi concebido para atender as demandas de armazenamento de dados de contexto. A sua estrutura vem sendo revisada a medida que acontecem trabalhos relacionados ao *middleware* EXEHDA (VENECIAN, 2010) (WARKEN, 2010). Nesta dissertação de mestrado foram incluídas tabelas no RIC para atender as demandas de gerenciamento de usuários e agendamento de uso dos equipamentos compartilhados.

A estrutura prototipada traduz a organização da arquitetura do *middleware* EXEHDA, contemplando as relações entre as aplicações, componentes, sensores, ambientes e os contextos de interesse. O RIC armazena ainda os dados de configuração da arquitetura e as publicações provenientes dos sensores existentes no ambiente ubíquo monitorado.

Dentre as informações de configuração, são destacados os aspectos relacionados aos sensores e aos contextos em que esses estão envolvidos. Por exemplo, os limites máximos e mínimos dos valores publicados para cada sensor pertencente aquele contexto de interesse. Esses dados são utilizados pelas regras do Tratador de Regras, o qual dispara as ações pertinentes em função do estado contextual.

As informações de contexto capturadas pelo Servidor de Borda são registradas no RIC com o seu valor correspondente, *timestamp* da coleta, gerente de borda de origem e o *timestamp* de registro. Para agilizar as consultas, os registros são indexados por sensor, coletor e *timestamp* da coleta.

As informações do RIC estão disponíveis para as aplicações em execução nos diferentes nodos que integram a estrutura computacional ubíqua provida pelo *middleware* EXEHDA.

3.2.5 Módulo de Notificação

O Módulo de Notificação é responsável por notificar o resultado do processamento contextual realizado pelo Módulo de Interpretação. O mesmo mantém um histórico das notificações em um banco de dados denominado Repositório de Contextos Notificados (RCN), o qual é necessário para procedimentos de auditoria.

O Módulo de Notificação opera recebendo subscrições de todos os serviços e/ou aplicações que desejem notificações a respeito dos estados contextuais, interoperando através do Módulo de Comunicação.

De modo mais específico, possui as seguintes funcionalidades:

- Realizar a notificação a respeito da variação do estado das informações contextuais aos outros serviços do *middleware* que empregam regularmente as mesmas, por exemplo o Serviço da Adaptação (WARKEN, 2010);
- Receber subscrições de aplicações e/ou serviços remotos, que oportunamente necessitem do estado das informações contextuais e notificá-los com as informações desejadas;
- Manter o RCN atualizado;
- Repassar ao Módulo de Atuação as decisões referentes as alterações a serem realizadas no meio.

Deste modo, pode-se dizer que passam pelo Módulo de Notificação todas as decisões de atuação provenientes do tratamento autonômico de regras de processamento contextual.

3.2.6 Módulo de Comunicação

O Módulo de Comunicação é empregado por Servidores de Contexto remotos e/ou aplicações quando da solicitação de dados contextuais e/ou o disparo de atuadores. O mesmo provê a disseminação de informações para outros serviços do *middleware*, bem como o envio de mensagens aos usuários.

O mesmo recebe as requisições através de um barramento ESB, e interopera com outros componentes do EXEHDA-UC empregando mensagens próprias, bem como com os usuários utilizando protocolos públicos direcionados ao envio de mensagens através da rede celular (SMS - Short Message Service), do Google Talk, e e-mails.

O Módulo de Comunicação contempla um repositório responsável por armazenar informações necessárias para o envio de alertas. Dentre as informações tratadas existem nome, número do telefone celular, email, conta do Gtalk. Neste repositório também ficam armazenadas todas as mensagens disparadas pelo Módulo.

3.2.7 Módulo de Gerenciamento

O Módulo de Gerenciamento tem por objetivo permitir ao usuário um gerenciamento confortável das configurações do Servidor de Contexto. O mesmo provê facilidades para que sejam especificados os diferentes aspectos dos sensores e atuadores, bem como informações dos equipamentos cujo contexto está sendo aquisitado.

Uma descrição de suas funcionalidades está disponível no Anexo B.

3.3 Módulo de Acesso Móvel

O Módulo de Acesso Móvel foi concebido com a intenção de prover acesso móvel ao EXEHDA-UC. O mesmo está organizado em dois blocos: (bloco A) exibição de informações contextuais e (bloco B) alertas pró-ativos (vide Figura 10).

Particularmente, a disponibilização de alertas pró-ativos em uma plataforma de hardware que possa acompanhar o usuário enquanto este desempenha suas atividades nos mais diferentes lugares, se mostrou um procedimento que potencializa a ubiquidade da solução de consciência de contexto disponibilizada.

Uma prototipação deste módulo foi realizada no trabalho (JACCOTTET, 2012), voltada para *smartphones* e/ou tablets que empregam o sistema operacional Android (GOOGLE, 2012).

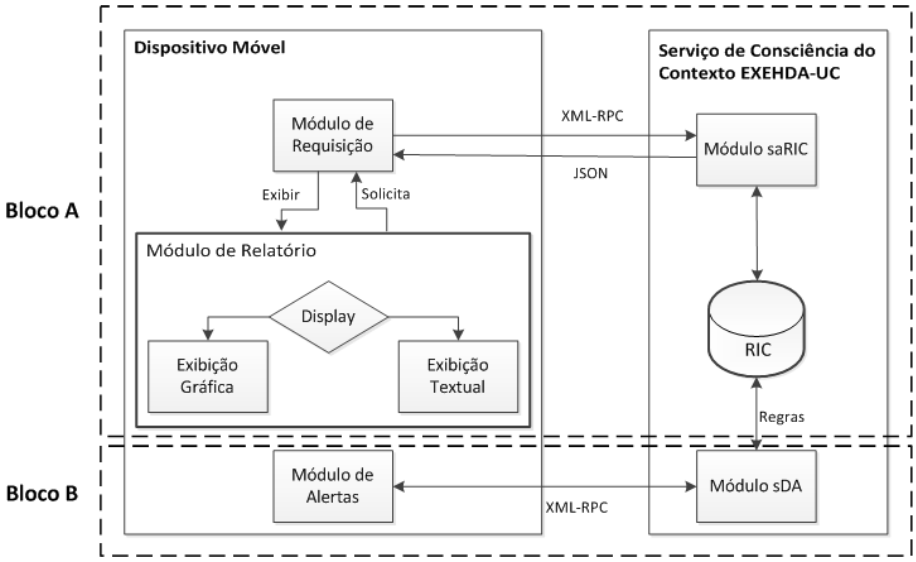


Figura 10: Visão geral do Módulo de Acesso Móvel

3.3.1 Bloco de Exibição de Informações Contextuais

O Bloco de Exibição de Informações Contextuais executa no dispositivo móvel e disponibiliza ao usuário relatórios tanto gráficos como textuais acerca das informações contextuais de seu interesse.

A comunicação do bloco com o Servidor de Contexto acontece empregando um protocolo de dois passos. No primeiro é feita uma inspeção das informações de contexto que estão sendo tratadas. No segundo são solicitados dados específicos para determinada informação contextual em um intervalo de tempo.

Suas funcionalidades foram organizadas em três módulos, conforme descrito a seguir.

(i) Módulo de Relatório

Este módulo trata da exibição das informações contextuais no dispositivo móvel. As principais operações disponibilizadas através da sua interface são:

- Requisição da lista de quais informações contextuais são tratadas pelo Serviço de consciência de contexto EXEHDA-UC;
- Seleção da informação contextual que será exibida no relatório;
- Escolha da natureza do relatório: gráfico ou textual;
- Oferecer ao usuário, uma vez exibido o relatório, alternativas para personalização do período dos dados mostrados.

(ii) Módulo saRIC

O módulo de acesso ao RIC, saRIC, consiste de um servidor que permanece em execução junto ao serviço de consciência de contexto (EXEHDA-UC), com total acesso ao RIC. Suas funcionalidades estão resumidas a seguir:

- Retornar a lista das informações contextuais gerenciadas pelo Serviço de consciência de contexto;
- Retornar as informações contextuais pertinentes ao contexto de interesse do usuário, para o intervalo de tempo desejado.

(iii) Módulo de Requisição

O Módulo de Requisição tem por objetivo solicitar informações contextuais ao saRIC, a pedido do Módulo de Relatório. O mesmo contempla quatro funções:

- Requisitar a lista de todas as informações contextuais tratados pelo Serviço de Conciência de Contexto;
- Requisitar uma determinada informação contextual desejada pelo usuário no intervalo de tempo especificado;
- Enviar esta solicitação ao saRIC através do barramento ESB do mesmo;
- Receber e interpretar as informações contextuais provenientes do saRIC, disponibilizando as mesmas ao Módulo de Relatório.

3.3.2 Bloco de Tratamento de Alertas pró-ativos

O Bloco de Tratamento de Alertas pró-ativos tem por objetivo sinalizar o usuário quando da ocorrência de eventos de seu interesse através de notificações pró-ativas. O mesmo tem suas funcionalidades organizadas em dois módulos conforme a seguir.

(i) Módulo de Alertas

O Módulo de Alertas executa no dispositivo móvel, disponibilizando ao usuário alertas, empregando para isso o mecanismo nativo de notificação da plataforma móvel. Esta opção tem como característica propiciar ao usuário um gerenciamento integrado da natureza dos alertas praticados pelo seu dispositivo móvel. As suas principais funções são:

- Recuperar no intervalo de tempo especificado pelo usuário os alertas junto ao Módulo sDA;
- Disponibilizar ao usuário estes alertas na área de notificação do dispositivo móvel.

(ii) Módulo sDA

O módulo de Distribuição de Alertas (sDA) executa no mesmo equipamento do Serviço de consciência de contexto em regime de operação ininterrupta. Suas principais funções são:

- Receber os alertas para os dispositivos móveis produzidos pelo tratador de regras do Serviço de consciência de contexto;
- Disponibilizar aos dispositivos móveis os alertas.

Este módulo opera mantendo os alertas produzidos pelas diferentes regras contextuais, liberando o Módulo de Interpretação. Seu acesso pelos dispositivos móveis ocorre através do Módulo de Comunicação o qual através de um barramento ESB propicia acesso as diferentes funcionalidades do Servidor de Contexto.

3.4 Considerações sobre o capítulo

Neste capítulo foram apresentadas a concepção e modelagem do EXEHDA-UC, sendo descrita a modelagem de arquitetura de software, bem como o Servidor de Borda, o Servidor de Contexto com seus respectivos componentes e o Módulo de acesso móvel.

4 TRABALHOS RELACIONADOS

Este capítulo sintetiza o estudo feito dos projetos que constituem um conjunto representativo do que vêm sendo desenvolvido nos últimos anos na direção de infraestruturas para suporte à consciência de contexto na computação ubíqua. As características tratadas abrangem aspectos relacionados à arquitetura de software, bem como às estratégias empregadas pelos projetos para aquisição e processamento dos dados contextuais. O critério de escolha destes trabalhos relacionados foi a sua proximidade com o EXEHDA-UC. A discussão dos projetos e uma análise comparativa entre estes e o EXEHDA-UC também é apresentada.

4.1 *Middleware* CARE

O *middleware* CARE (*Context Aggregation and REasoning*) (BETTINI; MAGGIORINI; RIBONI, 2007) provê suporte para (i) aquisição de dados de contexto de diferentes fontes; (ii) raciocínio sobre o contexto com base em políticas distribuídas e; (iii) tratamento de informações contextuais conflitantes. O objetivo central do CARE é a entrega de uma descrição consistente e precisa do contexto atual, de modo que a reação mais adequada ao mesmo possa ser praticada durante o provisionamento de serviços.

Desse modo, o CARE tem como principais contribuições a proposição de uma arquitetura distribuída para a aquisição e tratamento do contexto, bem como um mecanismo para lidar com inconsistências decorrentes da aquisição de dados de contexto e políticas provenientes de fontes diversas.

A proposta do *middleware* CARE para construção de uma visão integrada dos dados de contexto considera o envolvimento de três entidades: (i) o usuário com seus dispositivos; (ii) o operador de rede com sua infraestrutura e; (iii) o provedor de serviços com infraestrutura própria.

CARE utiliza o termo “perfil” para indicar um conjunto de parâmetros de contexto. A personalização dos parâmetros é feita por regras definidas tanto pelo usuário como pelo provedor de serviços, e gerenciadas pelo correspondente

gerenciador de perfil associado a cada entidade.

O *middleware* CARE utiliza para modelagem do contexto técnicas baseadas em ontologia e linguagem de marcação (CC/PP - *Composite Capabilities/ Preference Profiles*). CC/PP estende o vocabulário RDF (*Resource Description Framework*) para expressar características de dispositivos e preferências de usuários. Para empregar esta abordagem híbrida, CARE considera dois tipos de contexto: (i) contexto simples, representado por CC/PP, que abrange, por exemplo, descrições de recursos; e (ii) contexto complexo, representado por um modelo ontológico, que abrange, por exemplo, atividades e interações do usuário.

CARE utiliza políticas expressas através de regras para definir como os dados de contexto devem ser derivados, indicando a forma de raciocínio a ser utilizada. De acordo com o tipo de contexto (simples ou complexo) pode ser um processo de raciocínio sobre a modelagem baseada em CC/PP e/ou ontologia. Para que seja possível um processo de raciocínio abrangendo tanto atributos CC/PP como classes da ontologia, existe um mapeamento entre estes modelos.

Os módulos do provedor de contexto foram escritos com a perspectiva de otimizar tarefas computacionalmente intensivas, como fusão de perfis e transformação de políticas.

4.2 Plataforma CoCA

CoCA (*Collaborative Context-Aware*) (EJIGU; SCUTURICI; BRUNIE, 2008) é uma plataforma para provimento de serviços de consciência de contexto. CoCA é baseada em uma proposta de gerenciamento do contexto, denominado HCoM (*Hybrid Context Management*). Esta proposta gerencia a forma como os dados contextuais são coletados, organizados, representados, armazenados e disseminados. Os serviços da plataforma CoCA interpretam e agregam os dados brutos de contexto, transformando-os em contextos de alto nível com maior significado. Então, a plataforma executa um processo de raciocínio sobre o contexto, e decide sobre as ações de reação ao contexto que devem ser disparadas. O conhecimento e as decisões correspondentes a atual instância de contexto são armazenadas em uma base de dados de conhecimento.

CoCA emprega um modelo híbrido de contexto, baseado em ontologias e banco de dados relacional, para representação e processamento dos dados contextuais. Os processos de raciocínio e reação ao contexto são executados com base em regras e políticas.

A arquitetura da plataforma CoCA é constituída por quatro grupos: gerenciador de interfaces; fonte de dados; serviços centrais; e serviços complementares.

O grupo “Gerenciador de Interfaces” controla as interfaces com o usuário e com outros módulos específicos para determinados domínios de aplicações. Também, gerencia ações que são disparadas em função do domínio de aplicação específico, no qual a plataforma CoCA é usada.

O grupo “Fonte de Dados” possui um componente heurístico com características híbridas que ajuda a lidar com a seleção e carga de contexto a partir do banco de dados relacional para o esquema semântico, otimizando o processo de raciocínio.

O grupo “Serviços Centrais” fornece serviços de consciência de contexto, após a realização do processo de raciocínio e tomada de decisões. Por sua vez, o grupo “Serviços Complementares” é constituído por serviços relacionados a descoberta de conhecimento, colaboração e segurança.

De forma mais detalhada, os “Serviços Centrais” utilizam como entrada os dados providos pelo modelo HCoM, sendo constituído por um conjunto de serviços para raciocínio, agregação, interpretação, decisão e motor de ações. Estes serviços são responsáveis pelo raciocínio e decisão sobre as ações a serem disparadas. Os raciocinadores desempenham um papel importante para os serviços centrais da CoCA, pois podem derivar conclusões a partir de coleções de dados de contexto, regras e da ontologia.

Uma das contribuições centrais da plataforma CoCA é a característica seletiva de carregar somente dados de contexto relevantes para o Raciocinador. Com isso, minimiza o espaço de busca de raciocínio e reduz a sobrecarga do raciocinador, melhorando a eficiência do serviço. Para tanto, é empregada uma estratégia baseada em heurísticas que levam em consideração informações do usuário, disponibilidade de recursos e políticas institucionais. Como resultado, são minimizadas as limitações de escalabilidade da maioria dos sistemas de raciocínio.

4.3 *Framework* HiCon

HiCon (*Hierarchical Context*)(K. CHO I. HWANG; RHEE., 2008) é um *framework* para monitoramento e composição de contextos dinâmicos e hierárquicos. O *framework* viabiliza a construção de serviços de consciência de contexto com suporte a elevada escalabilidade dos ambientes de computação ubíqua.

Considerando aspectos relacionados à semântica e às estruturas de contextos, serviços e redes, três grandes desafios foram identificados pelos autores para o desenvolvimento do *framework* HiCon. Primeiro, o *framework* deve suportar a composição dinâmica de diversos contextos que abrangem

diferentes escalas como pessoal, local, regional e global. Em segundo lugar, o *framework* deve refletir a natureza hierárquica das redes, mapeando para o ambiente ubíquo as diferentes coberturas em que estas podem ser utilizadas. Por fim, o *framework* deve lidar com a escala massiva de computação necessária para o monitoramento e a composição de contextos.

Para gerenciar estes desafios, o *framework* proposto é composto por três camadas de processamento, correspondentes aos níveis de contexto pessoal, regional e global. HiCon trata um amplo espectro de contextos, abrangendo, em termos de escala, desde o contexto pessoal até o global, e, em termos de complexidade, desde o contexto bruto coletado pelos sensores, até o altamente processado.

Nessa proposta de arquitetura estruturada em níveis, os componentes de software do HiCon atuam tanto na composição de contextos em cada nível (composição horizontal), como cooperam para composição entre os diferentes níveis (composição vertical). Além disso, a concepção da arquitetura do HiCon possibilita a abstração do contexto no nível mais baixo possível da hierarquia, o que contribui para a redução do custo de transmissão de dados para os níveis mais elevados. Assim, para melhorar a escalabilidade, HiCon funde e filtra dados contextuais fornecidos por níveis mais baixos em descritores de contexto mais expressivos e compactos.

4.4 *Middleware* Solar

Solar (G. CHEN; KOTZ., 2008) é um *middleware* centrado em dados para apoiar aplicações ubíquas conscientes do contexto no processo de coleta, agregação e disseminação de dados contextuais. Solar emprega uma abstração de dados baseada em atributos, em que semântica e funcionalidades de raciocínio podem ser adicionadas. Solar fornece um modelo de programação para composição de fluxo de dados, no qual as aplicações podem especificar uma computação modular para inferência sobre o contexto.

O *middleware* Solar utiliza um método de descoberta de recursos consciente de contexto para aplicações conseguirem lidar com a dinâmica do ambiente, e um método de disseminação de dados orientado a política para as aplicações lidarem com fluxos de dados rápidos. No tocante à organização, o Solar consiste de uma arquitetura distribuída construída sobre um roteamento peer-to-peer, escalável e auto-organizada, o que melhora o gerenciamento.

No Solar, os aplicativos devem ter um modelo de programação que permita a composição dinâmica, sem exigir que o desenvolvedor ou usuário precise identificar sensores e dispositivos específicos, o que pode otimizar

o desenvolvimento de aplicações para ambientes inteligentes. Também, os aplicativos devem ter acesso aos valores dos dados de contexto para permitir uma reação automática das mesmas, sem instruções explícitas de aplicativos ou operadores humanos em tempo de execução. Esta escolha de projeto objetiva facilitar a implantação de aplicações ubíquas.

Solar emprega um modelo informal, com base na técnica chave-valor, para representação do contexto. Também, não registra dados históricos do contexto e a coleta de dados exige envolvimento direto das aplicações, devendo estas gerenciar sensores e requisições.

4.5 *Middleware* WComp

O *middleware* WComp (N. FERRY V. HOURDIN; TIGLI., 2011) é constituído por (i) uma infraestrutura de software; (ii) uma arquitetura de composição de serviços; e (iii) um mecanismo de reação ao contexto composicional. WComp propõe uma extensão da arquitetura orientada a serviços tradicional, denominada WSOAD (*Web Service Oriented Architecture for Device*) (MACKENZIE et al., 2006).

A arquitetura do *middleware* é empregada no gerenciamento dos aspectos relacionados à dinamicidade e à heterogeneidade de entidades na infraestrutura de software. As aplicações ubíquas gerenciadas pelo WComp são construídas a partir da composição de um conjunto de serviços Web para dispositivos providos pela arquitetura WSOAD, os quais devem interagir uns com os outros. Estes serviços não são editáveis pelos usuários, mas podem ser integrados novos serviços em tempo de execução de uma aplicação para adicionar novas funcionalidades a mesma.

A arquitetura para composição de serviços do *middleware* WComp permite construir aplicações pela composição dinâmica de serviços a partir da infraestrutura de software. Para permitir a reusabilidade de funcionalidades recém-criadas, e para fins de escalabilidade, cada composição pode ser encapsulada como um serviço composto.

A infraestrutura das aplicações ubíquas evolui de forma dinâmica devido à mobilidade dos nodos, falhas ou limitações de energia. Nesse sentido, a composição de serviço deve ser tão relevante quanto possível, de acordo com a infraestrutura de software. O gerenciamento destas composições não deve gerar uma sobrecarga administrativa, mas deve ser autoregulado em tempo de execução das aplicações, de forma transparente.

O mecanismo de adaptação ao contexto do *middleware* WComp é baseado na abordagem denominada *Aspect of Assembly* (AA). Esta abordagem,

direcionada à uma reação composicional às variações de contexto, é considerada particularmente adequada para ajustar um conjunto de serviços em reação a uma variação da infraestrutura ou a uma mudança das preferências dos usuários.

No tratamento do contexto, uma das principais limitações do *middleware* WComp está relacionada a não utilização de um modelo expressivo para representação das informações contextuais (N. FERRY S. LAVIROTTE; RIVEILL., 2009).

4.6 Discussão sobre os Trabalhos Relacionados em relação ao modelo

O estudo dos trabalhos relacionados foi desenvolvido em duas etapas. Em um primeiro momento, foram apresentadas as principais características destes trabalhos, objetivando identificar as tendências no tema da pesquisa, bem como contribuir com a definição das premissas da proposta. Em um segundo momento, nesta seção, os trabalhos relacionados são discutidos, tendo por base aspectos decorrentes das premissas consideradas para concepção do EXEHDA-UC. Estes aspectos, considerados na comparação entre os trabalhos relacionados (vide tabela 2), estão relacionados a seguir:

1. Arquitetura (distribuída ou centralizada)
2. Sensoriamento (suporte a redes de sensores)
3. Aquisição dos dados de contexto
4. Modelagem do contexto (técnica(s) empregada(s))
5. Armazenamento dos dados de contexto
6. Consulta sobre os dados de contexto
7. Suporte ao tratamento de regras
8. Suporte à atuação sobre o meio

Com relação à concepção da arquitetura para consciência de contexto, a maioria dos trabalhos estudados possui arquiteturas distribuídas. Entretanto, estas arquiteturas não mantém o caráter descentralizado para todas as etapas de tratamento dos dados de contexto, não atendendo os requisitos de distribuição em larga escala dos ambientes ubíquos. Por sua vez, o modelo arquitetural do

Tabela 2: Comparação dos Trabalhos Relacionados

	CARE	**CoCA**	**HiCon**	**Solar**	**WComp**
1	distribuída	centralizada	distribuída	distribuída	distribuída
2	não	sim	sim	não	não
3	gerenciada pelas aplicações	separada das aplicações	separada das aplicações	gerenciada pelas aplicações	separada das aplicações
4	ontologia e marcação	ontologia e relacional	-	chave-valor	-
5	não	sim	sim	não	não
6	não	sim (SPARQL)	sim (SQL)	não	não
7	sim	sim	não	sim	sim
8	não	não	não	não	sim

EXEHDA-UC diferencia-se dos trabalhos relacionados por estar estruturado de forma largamente distribuída, o que atende a característica de elevada dispersão de recursos nos ambientes ubíquos, bem como por abranger todas as etapas de tratamento das informações de contexto, desde a aquisição até os procedimentos de atuação sobre o meio.

Os servidores de borda do EXEHDA-UC podem gerenciar redes de sensores e atuadores que se comunicam pelo protocolo 1-Wire [1]. Com isso, pode ser otimizado o gerenciamento tanto da aquisição dos dados de contexto a partir de vários tipos de sensores, usual nos ambientes computacionais para provimento de aplicações ubíquas, como da atuação distribuída sobre o meio físico. Tal característica é encontrada em parte nos projetos CoCA e HiCon, que têm suporte a redes de sensores. O projeto WComp, por sua vez, permite atuação sobre o meio, entretanto, não suporta o gerenciamento de redes de atuadores.

Com exceção dos projetos CARE e Solar, os demais prevêem o emprego de mecanismos específicos para aquisição do contexto. Estes mecanismos adotam uma estratégia de separação entre a obtenção e o uso do contexto. Essa estratégia é um dos aspectos centrais para a concepção de arquiteturas de suporte ao processamento do contexto. O EXEHDA-UC também obtém os dados contextuais de forma separada das aplicações que os utilizam. Por outro lado, diferente dos projetos relacionados, o EXEHDA-UC agrega um caráter autônomo à obtenção dos dados de contexto, visto que estes continuam a ser obtidos pelo mecanismo de aquisição, mesmo que as aplicações interessadas em seu uso estejam inoperantes.

Características, como padronização e expressividade, são consideradas relevantes para o processamento dos dados contextuais, particularmente no que

[1] http://ubiq.inf.ufpel.edu.br/1-wire/

tange ao processo de raciocínio. Assim, as técnicas baseadas em ontologias, que implementam estas características, tem sido utilizadas para modelagem dos dados contextuais nas infraestruturas de suporte à consciência de contexto.

Entretanto, os modelos ontológicos apresentam limitações quanto à escalabilidade, o que pode comprometer o desempenho à medida que aumenta o número de classes e/ou instâncias da ontologia (D'AQUIN; NOY, 2012). Assim, observa-se que alguns projetos estudados, como CARE e CoCA, empregam um modelo híbrido para representação do contexto, aliando a elevada expressividade das ontologias com outra técnica que minimize as limitações de eficiência e escalabilidade. Porém, independente de ser empregada de forma individual ou como parte de um modelo híbrido, o uso de ontologias tende a elevar a complexidade e impor uma sobrecarga ao tratamento dos dados de contexto.

O EXEHDA-UC mantém compatibilidade com o Módulo de Interpretação proposto no EXEHDA-SS (VENECIAN, 2010) garantindo assim suporte a modelos de contexto baseados em ontologias. Porém, para o Projeto AMPLUS (descrito no capítulo 5), diferentemente dos trabalhos relacionados, o EXEHDA-UC explorou a técnica relacional para representação do contexto, descrita por um modelo entidade-relacionamento, implementado em banco de dados (vide RIC, seção 3.2.4). Com isso, é priorizada a eficiência para o processamento e otimização de consultas, bem como a capacidade de manutenção, segurança, escalabilidade e distribuição, além da disponibilização de ferramentas administrativas para o gerenciamento da estrutura do modelo.

A maioria dos projetos estudados possui suporte ao tratamento de regras, porém esta funcionalidade usualmente está restrita a algumas etapas do processamento do contexto, principalmente à interpretação dos dados contextuais. O EXEHDA-UC, diferencia-se destes trabalhos, por sua arquitetura de software ter sido concebida para dar suporte ao tratamento distribuído de regras personalizáveis, as quais podem estar vinculadas aos diferentes níveis de tratamento dos dados contextuais, tanto nos servidores de borda, como nos servidores de contexto.

4.7 Considerações sobre o capítulo

Este capítulo apresentou diversos projetos que propõem infraestruturas de suporte à consciência de contexto. A descrição apresentada para cada projeto buscou caracterizar a arquitetura de software e as estratégias que os mesmos empregam para o tratamento dos dados de contexto. Essa sistematização de

informações da literatura teve o objetivo de identificar tendências no tema de pesquisa, bem como contribuir com a definição das premissas de concepção do EXEHDA-UC.

5 EXEHDA-UC: ESTUDO DE CASO

Neste capítulo estão resumidos os principais aspectos das tecnologias utilizadas na prototipação, bem como o estudo de caso empregado na avaliação das funcionalidades do EXEHDA-UC. O estudo de caso contemplou tarefas referentes ao sensoriamento e a coleta de informações contextuais, processamento, dedução e notificação dos dados de contexto, aos demais serviços do *middleware*.

5.1 Estudo de Caso: Projeto AMPLUS

O Projeto AMPLUS (*Automatic Monitoring and Programable Logging Ubiquitous System*) foi desenvolvido nesta Dissertação com o objetivo promover soluções da Computação Ubíqua para o Laboratório Didático de Análise de Sementes (LDAS) da FAEM/UFPEL.

Dentre os serviços a serem providos, destaca-se a consciência de contexto em que se encontram os equipamentos do LDAS, com o respectivo registro histórico do contexto, e uma atuação pró-ativa quando necessário.

5.1.1 Local de realização - LDAS

O Laboratório Didático de Análise de Sementes (LDAS) ”Flávio Farias Rocha” integra o Programa de Pós Graduação em Ciência e Tecnologia de Sementes, do Departamento de Fitotecnia, da Faculdade de Agronomia Eliseu Maciel, da Universidade Federal de Pelotas. Foi inaugurado em 1973, permitindo a realização de aulas práticas, pesquisas e prestações de serviço.

O Programa de Pós-graduação em Ciência e Tecnologia de Sementes, ao qual o laboratório está vinculado, teve seu inicio na década de 70 com o curso de mestrado. Hoje o programa possui cursos de especialização, mestrado acadêmico, mestrado profissional e doutorado. Ao longo de sua existência, tem contribuído com a capacitação técnica de centenas de especialistas, mestres e doutores, oriundos de vários países americanos, como Argentina,

Uruguai, Paraguai, Chile, Peru, Bolívia, Colômbia, Venezuela, Guiana, Equador, Guatemala, México e Cuba, como também Moçambique e, inclusive, do Oriente Médio.

Conforme exigência legal do Ministério da Agricultura Pecuária e Abastecimento (MAPA), os Laboratórios de Análise de Sementes (LAS) necessitam implementar um sistema de qualidade baseado na norma ABNT NBR ISO/IEC 17025:2005. Esta norma estabelece os critérios para os laboratórios que desejam demonstrar sua competência técnica, que possuem um sistema de qualidade efetivo e que são capazes de produzir resultados tecnicamente válidos. O atendimento desta norma constituiu a motivação central para implementação do Projeto AMPLUS.

O LDAS não presta serviço para terceiros. O objetivo é didático, ou seja, serve de apoio às pesquisas, aulas práticas de graduação e pós-graduação, cursos de treinamento oferecidos pelo Programa de Pós- graduação em Ciência e Tecnologia de Sementes.

5.1.2 Motivação para realização

Um laboratório de análise de sementes pode ser considerado um centro de controle de qualidade, que tem como principal função fornecer apoio para padronização dos processos, com base em técnicas reconhecidas, ao sistema de produção de sementes em todas as suas fases: semeadura, colheita, processamento, tratamento, armazenamento até a sua comercialização (CARVALHO; NAKAGAWA, 2000).

A principal finalidade da análise de sementes é a de determinar a qualidade de um lote de sementes e, consequentemente, o seu valor para a semeadura. Seus resultados são utilizados para a emissão de certificado, que acompanha a embalagem de sementes para a fiscalização do comércio e a normatização da produção. Essas são as bases para o beneficiamento, a comercialização, o armazenamento e a distribuição das sementes. A análise é, ainda, utilizada em trabalhos de pesquisa e na identificação de problemas de qualidade e suas causas. Assim, para a obtenção de sementes com um nível de qualidade proposto, é importante manter a produção sob controle e, dessa forma, a análise se constitui em instrumento imprescindível (NOVEMBRE, 2001)

Para que os objetivos das análises sejam atingidos é necessário que os laboratórios possuam equipamentos adequados e sigam métodos e procedimentos uniformes. As Regras para Análise de Sementes (RAS) descrevem os procedimentos de análise visando à padronização e resultados uniformes e comparáveis entre diferentes laboratórios e diferentes analistas,

dentro de certa tolerância, permitindo obter dados precisos que forneçam confiança aos usuários.

Determinados procedimentos exigem precisão dos equipamentos estabelecendo variações específicas de temperatura. Por exemplo, no teste de germinação a temperatura não deve ser maior do que +/- 2°C, em cada período de 24 horas (BRASIL, 2009) (ISTA, 2012) e ser capaz de manter uma temperatura uniforme em toda a câmara e a temperatura específica ao nível da prateleira. Para determinação do grau de umidade é permitido uma oscilação de temperatura nas estufas de $\pm$ 3°C (BRASIL, 2009) e $\pm$ 2°C (ISTA, 2012). No teste de Envelhecimento acelerado (ISTA, 2012) a variação aceita é de $\pm$ 0,3°C.

De acordo com (MARCOS FILHO, 1999) os resultados do teste de envelhecimento acelerado sofrem influência de vários fatores: período de exposição das sementes, grau de umidade das sementes, abertura da câmara durante o teste, tratamento com fungicida, tamanho da amostra, genótipo e temperatura. Sobre a temperatura o autor detalha a necessidade de atenção especial ao monitoramento da mesma durante o teste, para que sejam obtidos resultados mais seguros.

Para a condução dos testes de vigor das sementes, ferramenta imprescindível para o controle de qualidade, a precisão é fator fundamental para a reprodutibilidade e padronização. Portanto, é necessário compra de equipamentos adequados, treinamento de analistas, segmento rígido de todos os procedimentos dos testes, monitoramento de temperatura e umidade das sementes e tempo.

Os testes de vigor são uma ferramenta útil no controle interno de qualidade dentro de uma empresa, mas na comparação interlaboratorial os resultados não têm mostrado reprodutibilidade, provavelmente devido às oscilações de temperatura que ocorrem nos equipamentos.

Por sua vez, no teste de envelhecimento acelerado, quando se utiliza o método da câmara, que é necessário verificar se a câmara externa está devidamente regulada para atingir a temperatura desejada e se a variação não supera 0,3°C, acima ou abaixo do valor de calibração (40 a 45°C), a avaliação da temperatura deve ser efetuada através de termômetros situados no interior da câmara externa. E, se esta não apresentar condições para manter a umidade relativa elevada em seu interior, a calibração da temperatura é realizada após a colocação de recipiente com água no interior da câmara (MARCOS FILHO, 1999).

O Projeto AMPLUS foi concebido para permitir um registro dos estados contextuais dos equipamentos envolvidos, ao longo de todo o tempo de realização dos vários testes.

Um experimento pode durar várias semanas, e os parâmetros para a realização deste registro e o correspondente disparo de mensagens de alerta podem ser definidos pelo usuário em função do teste em questão.

5.1.3 Infraestrutura de Hardware e Software envolvidas

O estudo de caso compreendeu a instalação de um Servidor de Contexto para atender as demandas da UFPel como um todo, e em particular do LDAS; de um Servidor de Borda no ambiente físico do LDAS; e de uma rede de sensores padrão 1-wire neste ambiente.

O Servidor de Contexto foi instalado em um hardware com processador Intel E3400-2.6GHz de dois núcleos, com memória de 4Gb, sobre o Sistema Operacional Ubuntu Server. A escolha da distribuição Linux Ubuntu foi decorrente da adoção da mesma pelo LUPS para os seus equipamentos servidores.

O código do Servidor de Contexto está escrito na linguagem Phyton, sendo empregado XML-RPC (*Extensible Markup Language - Remote Procedure Call*) para implementação do barramento ESB utilizado para interoperabilidade. O RIC emprega o gerenciador PostGreSQL para implementação das bases de dados. A regras utilizadas no projeto AMPLUS para tratamento dos Contextos de Interesse dos usuários são escritas em Phyton. Informações sobre os aspectos de implementação dos servidores empregados no EXEHDA-UC estão disponíveis em http://ubiq.inf.ufpel.edu.br/exehda-uc.

Com o intuito de maximizar a disponibilidade do Servidor de Contexto, o mesmo foi instalado nas dependências do Centro de Informática da UFPel, o qual, além de um sistema de *no-breaks* de longa duração, conta com um sistema gerador que é ativado automaticamente quando da interrupção do fornecimento de energia elétrica.

O Servidor de Borda, por sua vez, foi instalado em um hardware com processador AMD Athlon XP2000-1.66GHz, com memória de 500Mb, sobre o Sistema Operacional Ubuntu Server. Diferentemente do Servidor de Contexto as exigências de poder computacional para ativação das funcionalidades do Servidor de Borda são bastante reduzidas.

O código do Servidor de Borda também é escrito em Phyton, sendo empregado XML-RPC para implementação do barramento ESB utilizado para recebimento dos comandos de atuação. A coleta de dados emprega o relógio do sistema, através da Crontab do Linux, para disparo dos eventos de leitura dos sensores.

A leitura dos sensores é efetivada por *drivers* (softwares) específicos por tipo de sensor, os quais são acionados pelo Componente Escalonador do Servidor

de Borda. As regras de contingência para supervisão dos atuadores são escritas em Phyton. A escolha da linguagem e a concepção das mesmas aconteceu considerando as demandas de tratamento de contextos de interesse do LDAS.

O Servidor de Borda foi instalado no LDAS com suporte de *no-break* com autonomia de 12h.

Como um desdobramento desta dissertação de mestrado, foi desenvolvida uma solução embarcada para Servidor de Borda, de baixo consumo energético e robustez operacional. Uma versão funcional do produto está em uso, sendo mantidos os esforços de otimização da plataforma de hardware e software disponibilizada para a comunidade usuária. O mesmo denomina-se O.S. MagicBox e informações estão disponíveis no Apêndice A. A O.S. MagicBox foi concebida tendo por base uma cooperação entre a O.S.Systems e a UFPel, a qual foi implementada através de um Projeto de Graduação de um colaborador da empresa, também aluno do Bacharelado em Ciência da Computação do CDTec/UFPel (LISBOA, 2012).

No decorrer desta dissertação, para atender as funcionalidades previstas para este estudo de caso, foi necessária a criação de alguns dispositivos para operarem na rede de sensores 1-wire concebida para o LDAS. Dentre os dispositivos desenvolvidos destacam-se: interface serial 1-wire, HUB, sensor de temperatura, grid de sensores de temperatura, sensor de umidade, módulo embarcado para aquisição de dados. Suas especificações estão disponíveis no Anexo A.

Foi priorizado, juntamente com os usuários, a instalação de uma infraestrutura de rede 1-wire nos 10 BODs (*Biological Oxigen Demand*) utilizados principalmente em testes de vigor e envelhecimento de sementes. Ao total foram instalados 15 sensores e um atuador.

O atuador disponibilizado para o LDAS consiste de um alerta luminoso que é ativado sempre que uma regra de processamento de contexto de interesse identificar a necessidade da atenção dos laboratoristas para com um dos equipamentos. Uma vez ativado o alerta o laboratorista deverá utilizar um dispositivo móvel ou fixo para averiguar a situação propriamente dita.

5.1.4 Funcionalidades disponibilizadas

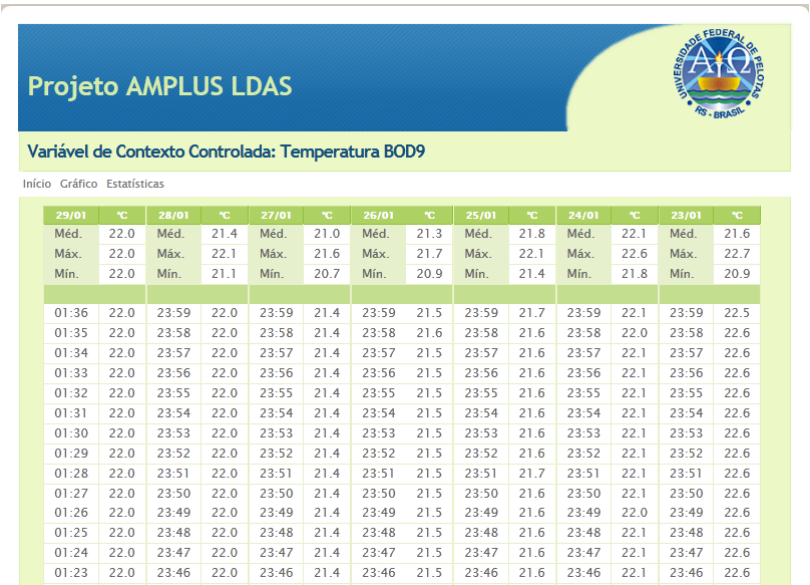
Após um ciclo de reuniões com os diferentes usuários do LDAS (professores, pesquisadores, mestrandos, doutorandos e laboratoristas), foram identificadas duas abordagens para desenvolvimento de aplicações que atendessem as funcionalidades necessárias para o LDAS, as quais seriam (i) direcionadas para interface Web, (ii) direcionadas para dispositivo móvel.

A interface Web para as aplicações inicia disponibilizando um menu para

Figura 11: Projeto AMPLUS - Seleção do contexto de interesse

seleção do contexto de interesse a ser exibido, como mostra a Figura 11.

Uma vez selecionado o contexto de interesse pelo usuário é oferecido um relatório textual com os dados coletados pertinentes a última semana. Juntamente com este relatório é disponibilizado um menu que permite selecionar uma visualização gráfica dos dados, bem como a geração de relatórios personalizados, apresentado na Figura 12.



29/01	℃	28/01	℃	27/01	℃	26/01	℃	25/01	℃	24/01	℃	23/01	℃
Méd.	22.0	Méd.	21.4	Méd.	21.0	Méd.	21.3	Méd.	21.8	Méd.	22.1	Méd.	21.6
Máx.	22.0	Máx.	22.1	Máx.	21.6	Máx.	21.7	Máx.	22.1	Máx.	22.6	Máx.	22.7
Mín.	22.0	Mín.	21.1	Mín.	20.7	Mín.	20.9	Mín.	21.4	Mín.	21.8	Mín.	20.9
01:36	22.0	23:59	22.0	23:59	21.4	23:59	21.5	23:59	21.7	23:59	22.1	23:59	22.5
01:35	22.0	23:58	22.0	23:58	21.4	23:58	21.6	23:58	21.6	23:58	22.0	23:58	22.6
01:34	22.0	23:57	22.0	23:57	21.4	23:57	21.5	23:57	21.6	23:57	22.1	23:57	22.6
01:33	22.0	23:56	22.0	23:56	21.4	23:56	21.5	23:56	21.6	23:56	22.1	23:56	22.6
01:32	22.0	23:55	22.0	23:55	21.4	23:55	21.5	23:55	21.6	23:55	22.1	23:55	22.6
01:31	22.0	23:54	22.0	23:54	21.4	23:54	21.5	23:54	21.6	23:54	22.1	23:54	22.6
01:30	22.0	23:53	22.0	23:53	21.4	23:53	21.5	23:53	21.6	23:53	22.1	23:53	22.6
01:29	22.0	23:52	22.0	23:52	21.4	23:52	21.5	23:52	21.6	23:52	22.1	23:52	22.6
01:28	22.0	23:51	22.0	23:51	21.4	23:51	21.5	23:51	21.7	23:51	22.1	23:51	22.6
01:27	22.0	23:50	22.0	23:50	21.4	23:50	21.5	23:50	21.6	23:50	22.1	23:50	22.6
01:26	22.0	23:49	22.0	23:49	21.4	23:49	21.5	23:49	21.6	23:49	22.0	23:49	22.6
01:25	22.0	23:48	22.0	23:48	21.4	23:48	21.5	23:48	21.6	23:48	22.1	23:48	22.6
01:24	22.0	23:47	22.0	23:47	21.4	23:47	21.5	23:47	21.6	23:47	22.1	23:47	22.6
01:23	22.0	23:46	22.0	23:46	21.4	23:46	21.5	23:46	21.6	23:46	22.1	23:46	22.6

Figura 12: Projeto AMPLUS - Relatório textual

O relatório gráfico oferecido pelo sistema permite a visualização simultânea das informações de vários sensores. A seleção dos sensores é feita a partir de um menu com suporte a múltipla seleção, conforme a Figura 13. Também é disponibilizado um recurso de inspeção que permite a comparação dos valores em um determinado instante do tempo (vide Figura 14).

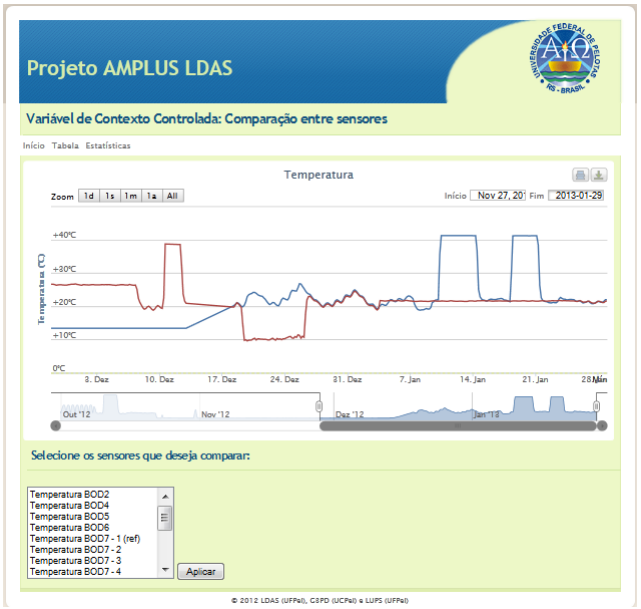


Figura 13: Projeto AMPLUS - Relatório gráfico

A janela de tempo dos dados que estão sendo visualizados pode ser definida pelo usuário através da mesma interface gráfica que exibe os valores sensorados.

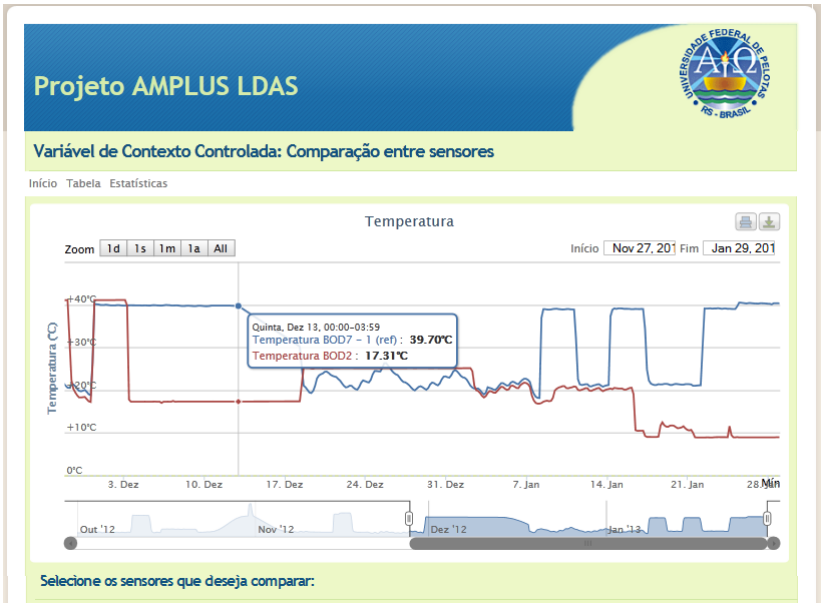


Figura 14: Projeto AMPLUS - Relatório gráfico com detalhes

Estas três características do relatório gráfico foram priorizadas a partir de: reuniões com usuários; disponibilização de versões para avaliação; e os decorrentes refinamentos nas funcionalidades oferecidas.

Para melhor suportar as necessidades de dados para as pesquisas realizadas no LDAS foi concebida uma funcionalidade que realiza o cruzamento de dados contextuais envolvendo múltiplos sensores a partir de diferentes regras. Toda

a manipulação é feita através da interface Web com facilidades para adição, remoção e edição de regras e seus parâmetros, como mostra a Figura 15.

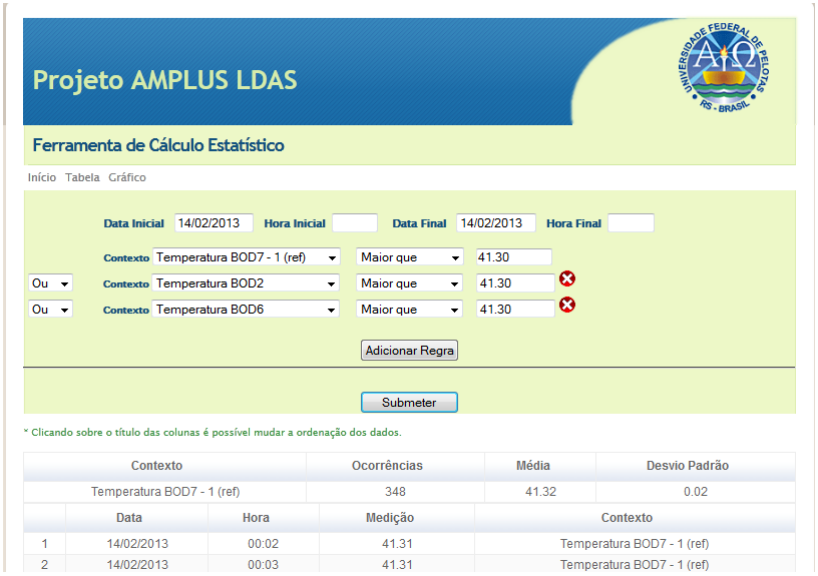


Figura 15: Projeto AMPLUS - Informações estatísticas

Com o intuito de promover a pro-atividade do Projeto AMPLUS junto a comunidade usuária, foram desenvolvidas interfaces para dois serviços públicos de comunicação: e-mail Internet e SMS para rede celular (vide Figura 16).

Figura 16: Projeto AMPLUS - Mensagem de alerta (a) por e-mail (b) por SMS

Estas mensagens são produzidas a partir do processamento das regras contextuais de forma autônoma pelo Servidor de Contexto do Projeto AMPLUS.

Considerando o caráter de uso compartilhado do LDAS, os endereços de e-mail e números de telefone para envio das mensagens são obtidos da funcionalidade de Agendamento (vide Anexo B).

Para atender o fato da rotina de trabalho dos laboratoristas do LDAS implicar em uma mobilidade nos diversos recintos do laboratório, foi desenvolvida uma interface de alerta visual (vide Figura 17) a qual será ativada sempre que

um dispositivo estiver em um estado contextual que exija atenção. A partir deste alerta, detalhes podem ser inferidos através da interface computacional do Projeto AMPLUS.

Figura 17: Projeto AMPLUS - Alerta Visual

A prototipação do módulo de Acesso Móvel foi direcionada para a plataforma Android SDK (GOOGLE, 2012). A Interface de abertura pode ser vista na Figura 18 e, através da mesma, é possível selecionar o sensor a ser exibido, seja através de um relatório gráfico ou textual.

Figura 18: Projeto AMPLUS - Acesso móvel - Interface de abertura

Os relatórios gráficos e textuais oferecem a opção do usuário especificar intervalo de visualização (hora, dia, semana), sendo o ajuste do eixo vertical feito de forma automatica, minimizando o emprego de rolagem de tela (vide Figuras 19 e 20).

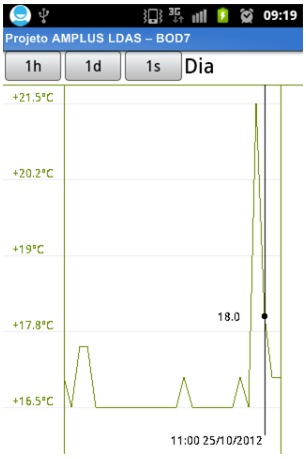


Figura 19: Projeto AMPLUS - Acesso móvel - Relatório Gráfico

Figura 20: Projeto AMPLUS - Acesso móvel - Relatório Textual

Para exibição dos alertas foi explorada a interface disponibilizada pela plataforma Android para esta finalidade, este aspecto potencializa a integração do mecanismo de alertas às funcionalidades do *smartphone* do usuário (vide Figura 21).

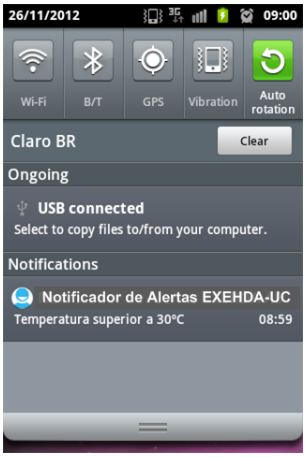


Figura 21: Projeto AMPLUS - Acesso móvel - Mensagem de Alerta

5.2 Considerações sobre o capítulo

Este capítulo apresentou um estudo de caso com a finalidade de validar a modelagem do EXEHDA-UC. O Projeto AMPLUS foi descrito, bem como seu local de realização, o LDAS, e as motivações para a realização do mesmo. Apresentou-se a infraestrutura de hardware e software, assim como as funcionalidades disponibilizadas aos usuários do laboratório.

6 CONSIDERAÇÕES FINAIS

Neste capítulo são resumidas as principais conclusões do que foi realizado ao longo do esforço de concepção e avaliação do EXEHDA-UC, enquanto um serviço para consciência de contexto na Ubicomp. São apresentados também os trabalhos futuros previstos e as publicações realizadas junto ao grupo de pesquisa.

6.1 Principais conclusões

A Computação Ubíqua contempla a criação de ambientes carregados de dispositivos computacionais que se integram a vida das pessoas de forma o mais transparente possivel, cujos componentes de software que compõem as aplicações reagem automaticamente de acordo com a consciência de contexto de seu interesse.

Constatou-se que para a execução de aplicações úbiquas conscientes de contexto, uma série de funcionalidades que devem ser providas, envolvendo desde a aquisição de informações contextuais, a partir do conjunto de fontes heterogêneas e distribuídas, seu processamento, armazenamento, e a realização de inferências para seu uso em tomadas de decisão. Estas funcionalidades cada vez mais são providas por *middlewares* de provisão de contexto, removendo da aplicação final do usuário o custo de gerenciamento da complexidade das mesmas.

Nesta perspectiva de reduzir o esforço de desenvolvimento de aplicações ubíquas vem sendo desenvolvido o *middleware* EXEHDA. O EXEHDA tem sua organização baseada em um núcleo mínimo e em serviços carregados sob demanda. Os serviços fornecidos estão organizados em quatro subsistemas, sendo um destes o Subsistema responsável pela adaptação e reconhecimento de contexto com o qual esta dissertação contribui diretamente.

O fato de, no EXEHDA, as condições de contexto serem pró-ativamente monitoradas pelo EXEHDA-UC, o desenvolvedor fica desobrigado de gerenciar

aspectos como coleta, processamento e armazenamento de dados contextuais.

No modelo computacional do EXEHDA tanto o suporte da execução do *middleware* como a aplicação podem se valer destes dados contextuais quando da gerência de aspectos funcionais e não-funcionais do processamento da aplicação.

No EXEHDA-UC é empregada uma estratégia colaborativa entre aplicação e ambiente de execução, através da qual é facultado ao programador individualizar regras para reger o comportamento de componentes que constituem o software da aplicação.

O EXEHDA-UC, quando comparado com outros trabalhos da área, segue a tendência de prover uma abordagem distribuída na aquisição de contexto e/ou atuação, com o adicional de aproximar da rede de sensores uma infraestrutura com capacidade autônoma de tratamento.

Potencializando a abordagem distribuída quando do tratamento contextual, o EXEHDA-UC suporta o conceito de rede de sensores e atuadores, o que além de contribuir com aspectos de modularidade do desenvolvimento do software necessário, também contribui para a organização dos procedimentos de criação e manutenção das redes envolvidas. Dentre os trabalhos relacionados, nem todos suportam a gerência de redes de sensores e atuadores.

Outra característica relevante do EXEHDA-UC quando da comparação com os trabalhos relacionados é que o mesmo gerencia a aquisição dos dados de contexto, seu processamento e armazenamento de forma independente das aplicações, em uma perspectiva autonômica baseada em regras. O emprego de regras em diferentes pontos da arquitetura distribuída do EXEHDA-UC constitui um diferencial significativo. Este comportamento, mesmo que parcialmente, é suportado apenas por parte dos trabalhos relacionados.

No que diz respeito ao modelo do EXEHDA-UC, foi concebida uma arquitetura de software distribuída, bem como os respectivos artefatos de hardware, na perspectiva de prover consciência de contexto na UbiComp atendendo as seguintes premissas:

(i) manter a compatibilidade do EXEHDA-UC com os recursos para processamento de regras previstos no trabalho (VENECIAN, 2010);

(ii) disponibilizar para a comunidade usuária um gerenciador que ofereça facilidades para especificação de contextos de interesse e sua associação a sensores/atuadores, regras e equipamentos;

(iii) dotar o EXEHDA de um padrão de mercado direcionado a interoperabilidade. Este padrão foi utilizado tanto entre os serviços distribuídos, como para comunicação com outros *middlewares*.

(iv) atender as demandas do Projeto AMPLUS, dentre as quais destaca-se:

- Aquisição, processamento e armazenamento de informações contextuais 24h por dia, 7 dias por semana;
- Suportar contextos de interesse formados por diversas fontes, e tratados por regras individualizadas nos diferentes servidores da arquitetura;
- Prover elevada autonomia quanto a operação ininterrupta do Servidor de Borda, o qual interopera com o meio (coleta e atuação);
- Disponibilizar as aplicações do AMPLUS para o usuário final do LDAS sob duas perspectivas: através de equipamentos de mesa, ou dispositivos móveis. No caso dos dispositivos móveis destaca-se a disponibilização de alertas de forma pró-ativa;
- Empregar para a coleta das informações contextuais de natureza física (temperatura, umidade, etc.) uma tecnologia escalável e de elevada robustez operacional. Neste sentido, é importante registrar que a rede 1-Wire empregada aliou robustez, facilidade de instalação e baixo custo.

No intuito de potencializar a robustez operacional do Servidor de Borda foi desenvolvida uma solução industrial junto a Empresa O.S.Systems [1]. O trabalho de sintetizar o modelo do Servidor de Borda do EXEHDA-UC enquanto uma arquitetura de hardware e software embarcada foi desenvolvido pelo Projeto de Graduação do BCC/UFPel (LISBOA, 2012). O produto concebido denomina-se O.S. MagicBox e alia baixo consumo de energia, facilidade de configuração pelo usuário final e possibilidade de ser utilizada em ambientes externos por não necessitar de mecanismos de resfriamento forçado. Informações sobre a O.S. MagicBox estão disponíveis no Apêndice A.

6.2 Trabalhos Futuros

Dentre os aspectos levantados para continuidade do trabalho destaca-se:

- Ampliação da rede de sensores e atuadores do LDAS da FAEM/UFPel atendendo outros equipamentos, tais como germinadores e estufas;
- Concepção de uma interface de acesso expresso, a ser disponibilizada no ambiente cujo contexto está sendo processado. Esta interface será baseada em um tablet o qual seria fixado em um local de fácil acesso pela

[1] http://ossystems.com.br/

comunidade usuária. Este tipo de solução, particularmente no caso do LDAS, atenderia o aspecto de elevada mobilidade presente nas atividades dos laboratoristas;

- Explorar estudos de caso em que as regras de processamento contextual utilizem outros mecanismos de inferência de mais alto nível, explorando recursos de inferência sobre os dados coletados;
- Utilizar a arquitetura do EXEHDA-UC para provimento de Consciência de Situação na qual são antecipadas possiveis situações futuras dos diferentes contextos do ambiente ubíquo.

6.3 Publicações

No desenvolvimento dos trabalhos desta Dissertação de Mestrado foram efetuadas algumas publicações relacionadas a mesma. Algumas publicações tratam da concepção do EXEHDA-UC e da seleção das tecnologias envolvidas no mesmo, outras contemplam a projeção do EXEHDA-UC nos estudos e pesquisas do LUPS na área de Consciência e Adaptação ao Contexto.

- GUSMÃO, M. Z.; YAMIN, A. C. EXEHDA-UC: Gerenciamento de Servidores de Contexto Distribuídos direcionado à Computação Ubíqua. In: II Seminário de Pesquisa em Computação da UFPel, 2011, PPGC - CDTEC. (SPC 2011)
- GUSMÃO, M. Z.; DUARTE, C.; LOPES, J. L.; YAMIN, A. C. EXEHDA-UC: Gerenciamento de Servidores de Contexto Distribuídos direcionado à Computação Ubíqua. In: 12ª Escola Regional de Alto Desempenho, 2012, Erechim, RS. (ERAD 2012)
- LOPES, J. L.; SOUZA, R. S.; GADOTTI, G. I.; GUSMÃO, M. Z.; YAMIN, A. C; GEYER, C. R. Uma Proposta para Consciência de Contextos Dinâmicos na Ubicomp. In: XXXIX Seminário Integrado de Software e Hardware, 2012, Curitiba, PR. (SEMISH 2012)
- LOPES, J. L.; SOUZA, R. S.; GEYER, C. R.; COSTA, C. A.; BARBOSA, J. V.; GUSMÃO, M. Z.; YAMIN, A. C. A model for context awareness in Ubicomp. In: BRAZILIAN SYMPOSIUM ON MULTIMEDIA AND THE WEB, 18., 2012, New York, NY, USA. Proceedings... ACM, 2012. p.161-168. (WebMedia 2012)
- LOPES, J. L.; SOUZA, R. S.; GEYER, C. R.; COSTA, C. A.; BARBOSA, J. V.; GUSMÃO, M. Z.; YAMIN, A. C. A middleware for context-aware

adaptation in Ubicomp. In: BRAZILIAN SYMPOSIUM ON MULTIMEDIA AND THE WEB, 18., 2012, New York, NY, USA. Proceedings... ACM, 2012. (WebMedia 2012) BEST PAPER

- LOPES, J. L.; SOUZA, R. S.; GUSMÃO, M. Z.; COSTA, C. A.; BARBOSA, J. V.; YAMIN, A. C; GEYER, C. R. Managing Adaptation in Ubicomp. In: XXXVIII Conferencia Latinoamericana en Informática, 2012, Medellín, Colômbia.(CLEI 2012)

- LOPES, J. L.; SOUZA, R. S.; GADOTTI, G. I.; GUSMÃO, M. Z.; COSTA, C. A.; BARBOSA, J. V.; YAMIN, A. C; GEYER, C. R. Uma Arquitetura Distribuída para Composição de Contextos Dinâmicos na Ubicomp. In: XXXVIII Conferencia Latinoamericana en Informática, 2012, Medellín, Colômbia.(CLEI 2012)

- GUSMÃO, M. Z.; LOPES, J. L.; YAMIN, A. C. EXEHDA-UC - Uma Arquitetura de Software direcionada à consciência de contexto na UbiComp. In: XIV Encontro de Pós-Graduação da UFPel, 2012, PRPPG - UFPel. (ENPOS 2012)

- GUSMÃO, M. Z.; YAMIN, A. C. EXEHDA-UC: Uma Arquitetura de Software direcionada à consciência de contexto na UbiComp. In: III Seminário de Pesquisa em Computação da UFPel, 2012, PPGC - CDTEC. (SPC 2012)

- LOPES, J. L. ; SOUZA, R. ; GUSMAO, M. Z. ; YAMIN, A. C.; GEYER, C. R. . Processamento de Contexto na Ubicomp: Uma Revisão Orientada a Aspectos Semânticos. Revista do CCEI, v. 16, p. 254-273, 2012. (CCEI 2012)

- JACCOLLET, D. P.; SANTOS, A. V.; VENECIAN, L. R. LOPES, J. L.; GUSMÃO, M. Z.; YAMIN, A. C. Uma Abordagem para Acesso Móvel ao EXEHDA-SS. In: 13ª Escola Regional de Alto Desempenho, 2013, Porto Alegre, RS. (ERAD 2013)

Um registro dos trabalhos desenvolvidos ao longo do Mestrado está disponível em <http://ubiq.inf.ufpel.edu.br/marciazg/>.

REFERÊNCIAS

ABOWD, G. D.; MYNATT, E. D. Charting past, present, and future research in ubiquitous computing. **ACM Transactions on Computer-Human Interaction**, [S.l.], v.7, n.1, p.29–58, Mar. 2000.

AWTREY, D. **Understanding 1-Wire Series**. [S.l.]: SPRINGBOK DIGITRONICS, 2004.

BELLAVISTA, P.; CORRADI, A.; FANELLI, M.; FOSCHINI, L. A Survey of Context Data Distribution for Mobile Ubiquitous Systems. **ACM Computing Surveys**, [S.l.], v.45, n.1, p.1–49, 2012.

BELOTTI, R.; DECURTINS, C.; GROSSNIKLAUS, M.; NORRIE, M. C.; PALINGINIS, A. **Modelling Context for Information Environments**. [S.l.]: In: Workshop on Ubiquitous Mobile Information and Collaboration Systems (UMICS), 2004.

BETTINI, C.; MAGGIORINI, D.; RIBONI, D. Distributed Context Monitoring for the Adaptation of Continuous Services. **DICo - University of Milan, Italy**, [S.l.], 2007.

BIZTALK. **Microsoft BizTalk Server**. Disponível em: http://www.microsoft.com/biztalk.Acesso em: dezembro de 2011.

BRASIL. **Regras para Análise de Sementes**. Brasília: Mapa/ACS, Ministério da Agricultura, Pecuária e Abastecimento. Secretaria de Defesa Agropecuária.

CAMEL. **Apache Camel - Open Source Integration Framework.** Disponível em: http://camel.apache.org. Acesso em: dezembro de 2011.

CARVALHO, N.; NAKAGAWA, J. **Sementes**: ciência, tecnologia e produção. Jaboticabal: Funep, 4. ed.

COSTA, C. A.; YAMIN, A. C.; GEYER, C. F. R. Toward a General Software Infrastructure for Ubiquitous Computing. **IEEE Pervasive Computing**, Los Alamitos, CA, USA, v.7, n.1, p.64–73, 2008.

D'AQUIN, M.; NOY, N. F. Where to publish and find ontologies? A survey of ontology libraries. **Journal of Web Semantics**, [S.l.], n.11, p.96–111, 2012.

DEY, A.; ABOWD, G.; SALBER, D. **A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications**. [S.l.]: Human-Computer Interaction, 2001. 97-166p.

EJIGU, D.; SCUTURICI, M.; BRUNIE, L. Hybrid Approch to Collaborative Context-Aware Service Plataform for Pervasive Computing. **Journal of Computers**, [S.l.], 2008.

FRANTZ, R. **"Integración de Aplicaciones**: Un Linguaje Específico de Dominio para el Diseño de Soluciones de Integración". [S.l.]: Sevilla, Spain, 2008.

G. CHEN, M. L.; KOTZ., D. Data-centric middleware for context-aware pervasive computing. **Pervasive Mob. Comput.**, [S.l.], 2008.

GAUVIN, M.; BOURRY-BRISSET, A. C.; AUGER, A. **Context, Ontology and Portfolio**: Key Concepts for a Situational Awareness Knowledge Portal. [S.l.]: In: Proceedings of the 37th Hawaii International Conference on System Sciences, 2004.

GOOGLE. **Android Developers**. Disponível em http://developer.android.com.

GREENBERG, S. **Context as a Dynamic Construct**. [S.l.]: Human Computer Interaction, 2001. 257-268p.

GRIMM, R.; BERSHAD, B. N. **System Support for Pervasive Applications**. [S.l.]: Springer, 2003. 212–217p. (Lecture Notes in Computer Science, v.2584).

GUTWIN, C.; GREENBERG, S. A Descriptive Framework of Workspace Awareness for Real-Time Groupware. **Computer Supported Cooperative Work**, [S.l.], v.11, n.3-4, p.411–446, 2002.

HENRICKSEN, K.; INDULSKA, J.; RAKOTONIRAINY, A. **Modeling context information in pervasive computing systems**. Zurich, Switzerland: Proceedings of the First International Conference on Pervasive Computing, 2003. 167-180p.

ISAM. **InfraEstrutura de Suporte às Aplicações Móveis**. Disponível em: $<$ http://www.inf.ufrgs.br/ isam/index.html$>$. Acesso em março de 2012.

ISTA. **Seed Testing International**. $< http://www.seedtest.org/en/seed-testing-international-_content---1--1085.html >$, No. 144.

JACCOTTET, D. P. **EXEHDA-AD**: Acesso Móvel ao Serviço de Consciência do Contexto do EXEHDA. [S.l.]: UCPel, 2012. Projeto de Graduação.

K. CHO I. HWANG, S. K. B. K. J. L. S. L. S. P. J. S.; RHEE., Y. Hicon: a hierarchical context monitoring and composition framework for next-generation context-aware services. **IEEE**, [S.l.], 2008.

KORPIPAA, P.; MäNTYJäRVI, J.; KELA, J.; KERäNEN, H.; MALM, E. **Managing Context Information in Mobile Devices**. [S.l.]: IIEEE Pervasive Computing, 2003.

KRUMM, J. (Ed.). **Ubiquitous Computing Fundamentals**. [S.l.]: Chapman and Hall/CRC, 2010.

LISBOA, M. S. A. **Uma Solução Embarcada para o Tratamento de Sensores e Atuadores na Ubicomp**. [S.l.]: BCC/UFPel, 2012. Projeto de Graduação.

LIU, W.; LI, X.; HUANG, D. A Survey on Context Awareness. **IEEE**, [S.l.], 2011. Disponível: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=05972040.

LOPES, J. L.; SOUZA, R. S.; GEYER, C. R.; COSTA, C. A.; BARBOSA, J. V.; GUSMãO, M. Z.; YAMIN, A. C. A model for context awareness in Ubicomp. In: BRAZILIAN SYMPOSIUM ON MULTIMEDIA AND THE WEB, 18., 2012, New York, NY, USA. **Proceedings...** ACM, 2012. p.161–168. (WebMedia '12).

LOPES, J. L.; SOUZA, R. S.; GEYER, C. R.; COSTA, C. A.; BARBOSA, J. V.; GUSMãO, M. Z.; YAMIN, A. C. A middleware for context-aware adaptation in Ubicomp. In: BRAZILIAN SYMPOSIUM ON MULTIMEDIA AND THE WEB, 18., 2012, New York, NY, USA. **Proceedings...** ACM, 2012. (WebMedia '12).

LOPES, J. L.; SOUZA, R. S.; GEYER, C. R.; GADOTTI, G. I.; GUSMãO, M. Z.; YAMIN, A. C. Uma proposta para Consciência de Contexto Dinâmicos na Ubicomp. In: XXXIX SEMINáRIO INTEGRADO DE SOFTWARE E HARDWARE, 2012, Curitiba/PR. **Anais...** [S.l.: s.n.], 2012. (SEMISH 2012).

MACIEL, R. S. P.; ASSIS, S. R. **Middleware**: Uma solução para o desenvolvimento de aplicações distribuídas. [S.l.]: In: Científico - Ano IV, 2004.

MACKENZIE, M.; LASKEY, K.; MCCABE, F.; BROWN, P.; METZ, R. **Reference model for service oriented architecture 1.0**. Disponível em: $< http : //www.oasis - open.org/committees/tc_home.php?wg_abbrev = soa - rm >$.

MARCOS FILHO, J. **Teste de envelhecimento acelerado**. In: KRYZANOWSKI, F.C.; VIEIRA, R.D.; FRANÇA NETO, J.B. ABRATES, Comitê de vigor de sementes.

MAXIM INTEGRATED PRODUCTS, I. **DS18B20 Programmable Resolution 1-Wire Digital Thermometer**. Disponível em :<http://datasheets.maximintegrated.com/en/ds/DS18B20.pdf>. Acesso em janeiro de 2013.

MENGE, F. Enterprise Service Bus. **Free and Open Source Software Conference**, [S.l.], 2007.

MICROSOFT. **Microsoft ESB Guidance for BizTalk Server 2006 R2.** Disponível em: http://msdn.microsoft.com/en-us/library/ff647678.aspx.

MILLER, F.; VANDOME, A.; MCBREWSTER, J. **1-Wire**. [S.l.]: Alphascript Publishing, 2010.

MOSTEFAOUI, G. K.; ROCHA, J. P.; BREZILLON, P. **Context-Aware Computing**: A Guide for the Pervasive Computing Community. Beirute, Libano: Proceedings of the 2004 IEEE/ACS International Conference on Pervasive Services, 2004.

N. FERRY S. LAVIROTTE, J.-Y. T. G. R.; RIVEILL., M. Context adaptative systems based on horizontal architecture for ubiquitous computing. **Proceedings of the 6th International Conference on Mobile Technology, Application and Systems**, [S.l.], 2009.

N. FERRY V. HOURDIN, S. L. G. R. M. R.; TIGLI., J.-Y. Wcomp, a middleware for ubiquitous computing. **InTech**, [S.l.], v.1, n.8, p.171–176, 2011.

NOVEMBRE, A. **Avaliação da qualidade de sementes**. Seednews, v.5 n.3.

PERNAS, A. M. **Sensibilidade à Situação em Sistemas Educacionais na Web**. 2012. Tese (Doutorado em Ciência da Computação) — UFRGS.

POSLAD, S. **Ubiquitous Computing**: Smart Devices, Environments and Interactions. [S.l.]: Wiley, 2009.

ROSA, M.; BORGES, M.; SANTORO, F. **A Conceptual Framework for Analyzing the Use of Context in Groupware**. [S.l.]: In: Proc. of CRIWG'03, v. LNCS 2806, pp. 300-313, Springer Verlag Berlin, 2003.

RUSSELL S., N. P. **Artificial Intelligence**. [S.l.]: A Modern Approach, 2003.

PIMENTEL M. E FUKS, H. o. (Ed.). **Sistemas Colaborativos**. [S.l.]: Elsevier, 2011.

SCHILIT, B. **A Context-Aware Systems Architecture for Mobile Distributed Computing**. Columbia University: Ph.D. Thesis, 1995.

SCHILIT, B.; ADAMS, N.; WANT, R. **Context-aware computing applications**. Santa Cruz, California: In Proceedings of IEEE Workshop on Mobile Computing Systems and Applications, 1994. 85-90p.

SILVA, T. H.; CELES, C. S. F. d. S.; MOTA, V. F. S.; LOUREIRO, A. A. F. A picture of actual Ubicomp research exploring publications from important events in the field. In: SIMPóSIO BRASILEIRO DE COMPUTAçãO UBíQUA E PERVASIVA, 2012. **Anais...** [S.l.: s.n.], 2012. (SBCUP 2012).

SPRING. **Spring Integration**. Disponível em: http://www.springsource.org/spring-integration. Acesso em: dezembro de 2011.

STRANG, T.; LINNHOFF-POPIEN, C. **A context modeling survey**. Nottingham, England: Proceedings of the I International Conference on Ubiquitous Computing, 2004. 34-41p.

VENECIAN, L. **Um Mecanismo para Sensibilidade ao Contexto com Suporte Semântico na UbiComp**. 2010. Tese de Mestrado em Ciência da Computação — PPGINF/Centro Politécnico/UCPEL, Pelotas-RS. Disponível: http://olaria.ucpel.tche.br/luthiano/lib/exe/fetch.php?media=lvti.pdf.

WANG, X.; GU, T.; ZHANG, D.; PUNG, H. **Ontology based context modeling and reasoning using OWL**. [S.l.]: In: Workshop on Context Modeling and Reasoningat II IEEE International Conference on Pervasive Computing and Communication, 2004.

WARKEN, N. **Uma Proposta de Controle da Adaptação Dinâmica ao Contexto na Computação Ubíqua**. 2010. Tese de Mestrado em Ciência da Computação — PPGINF/Centro Politécnico/UCPEL, Pelotas-RS. Disponível: http://olaria.ucpel.tche.br/nelsiw/.

WEISER, M. The computer for the 21st century. **Scientific American**, [S.l.], v.265, n.3, p.94–104, 1991.

WEISER, M. Ubiquitous Computing. **IEEE Computer**, [S.l.], v.26, n.10, p.71–72, Oct. 1993.

YAMIN, A. C. **Arquitetura para um Ambiente de Grade Computacional Direcionada às Aplicações Distribuídas, Móveis e Conscientes de Contexto**

da Computação Pervasiva. 2004. Tese de Doutorado em Ciência da Computação — Instituto de Informática/UFRGS, Porto Alegre-RS.

YAML. **The Official YAML Web Site**. Disponível em: < http://www.yaml.org >. Acesso em outubro de 2012.

APÊNDICE A EXEHDA-UC: PRODUTO INDUSTRIAL ASSOCIADO

Durante a realização do Mestrado aconteceu uma cooperação entre a O.S.Systems e a UFPel a qual foi articulada através de um Projeto de Graduação de um funcionário da empresa, também aluno do Bacharelado em Ciência da Computação do CDTec/UFPel. O produto desenvolvido consiste de uma solução embarcada, de baixo consumo energético e robustez operacional. Uma versão funcional do produto está em uso, sendo mantido esforço de otimização da plataforma de hardware e software disponibilizada para a comunidade usuária.

A O.S. Systems[1] é uma empresa brasileira especializada no desenvolvimento de soluções baseadas em código aberto. A empresa foi criada e tem crescido desenvolvendo software e personalizando sistemas operacionais para seus clientes.

O O.S. MagicBox - Modelo E2-1W - (Figura 22) é um equipamento de pequenas dimensões e elevada robustez operacional, não exigindo sistema de dissipação térmica forçada. O mesmo é voltado para controle de sensores e atuadores. Utiliza a tecnologia 1-wire, oferecendo suporte a mais de 100 tipos de dispositivos. O O.S. MagicBox oferece uma interface web, através da qual é possível configurá-lo para executar diferentes ações em função de condições especificadas pelo usuário.

Figura 22: O.S. MagicBox

[1] http://ossystems.com.br/

Especificação de hardware

Processador: Freescale i.MX233 - ARM926EJ-S 454MHz
Memória: 64MB RAM
Armazenamento: 64MB NAND

Principais funcionalidades:

- Dois barramentos 1-wire
- Alarme e dashboard de informações
- Atribuição de valores do dispositivo
- Envio de dados através de HTTP (para integração com aplicações de terceiros)
- Coleta de dados persistente em caso de perda de conexão com a Internet
- Equipamento que utiliza premissas da computação embarcada e verde;
- Gerencia uma rede de sensores e atuadores padrão de rede 1-Wire;
- Emprega rede Ethernet para comunicação com os outros equipamentos do ambiente ubíquo;
- Configurável através de uma interface web ou comandos via TCP/IP;
- Interopera com o Servidor de Contexto repassando dados sensorados e/ou recebendo comandos de atuação.

ANEXO A HARDWARE DESENVOLVIDO

Rede 1-wire

O sistema 1-wire (MILLER; VANDOME; MCBREWSTER, 2010) é uma rede de transmissão de dados, também conhecida como MicroLAN, que possibilita a comunicação digital entre um computador ou microcontrolador, atuando como mestre, e dispositivos da série 1-wire tais como sensores, atuando como escravos. Por mestre, entende-se o elemento capaz de controlar e gerenciar a transmissão de dados. Por escravo, entende-se o dispositivo endereçado e gerenciado pelo mestre.

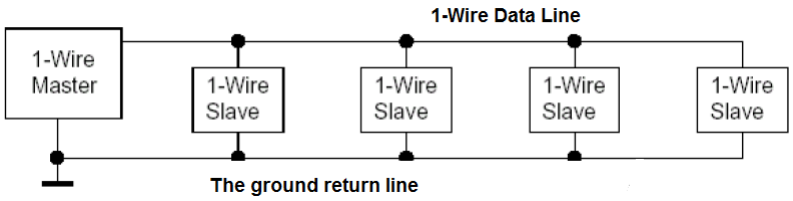


Figura 23: Rede 1-Wire (Fonte: (AWTREY, 2004))

Na rede 1-wire de transmissão de dados um único mestre pode ser conectado a múltiplos escravos em diversos tipos de topologia. Esta arquitetura confere ao sistema 1-wire versatilidade e simplicidade (vide Figura 23).

Todas as comunicações de 1-wire ocorrem digitalmente (estados lógicos on ou off), ao longo de um cabo (tipicamente par trançado). O dispositivo mestre inicia e controla todas as atividades no barramento de rede, atuando como uma interface entre a rede 1-Wire e a infraestrutura computacional.

Tanto o mestre do barramento como todos os dispositivos escravos atuam internamente como transceptores, enviando e recebendo dados através de uma única linha de dados.

No protocolo 1-Wire a linha de comunicações única, onde o tráfego se reveza, primeiro indo para um lado, depois o outro. Assim, os dados podem fluir nas duas direções, mas apenas em uma direção de cada vez (operação half duplex).

Interface Serial 1-Wire

Com o intuito de implementar a comunicação do Servidor de Borda com os dispositivos de uma rede 1-Wire, construiu-se uma interface para uso com a porta serial ou serial via USB (Figura 24). Este equipamento uma vez conectado a um computador, permite a leitura de informações de diversos sensores e o acionamento de atuadores através de *drivers* específicos.

Figura 24: Interface serial 1-Wire

A concepção da interface atendeu as especificações fornecidas pela Dallas Semiconductor, fabricante líder de dispositivos para a tecnologia 1-Wire. O diagrama esquemático desenvolvido está apresentado na Figura 25.

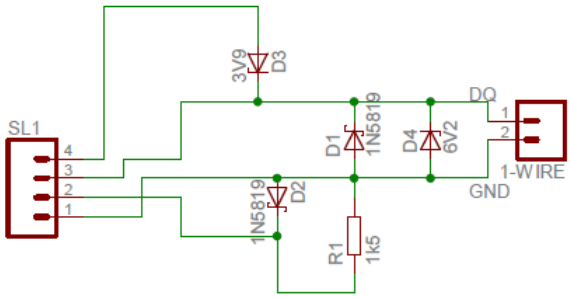


Figura 25: Interface serial 1-Wire - Diagrama Esquemático

HUB 1-Wire

Para atender uma proposta de cabeamento estruturado quando da implementação da rede 1-Wire no ambiente ubíquo, foi desenvolvido um HUB 1-Wire empregando conectores padrão RJ45. Com o suporte de conexões tipo RJ45 facilmente podem ser incluidos, excluidos e/ou trocados dispositivos 1-Wire (vide Figura 26).

Figura 26: HUB

Além das conexões necessárias para troca de dados, o HUB também disponibiliza as tensões necessárias para os sensores ou atuadores que necessitem de alimentação. Seu diagrama esquemático é apresentado na Figura 27.

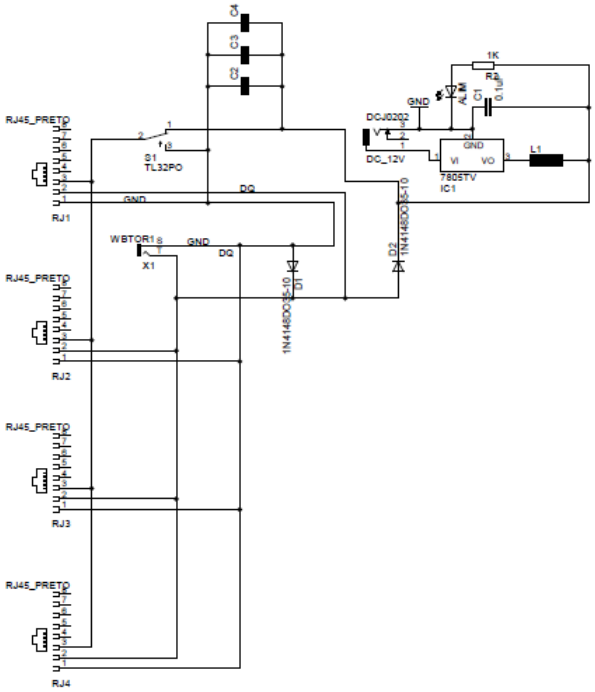


Figura 27: HUB - Diagrama Esquemático

Sensor de temperatura

O sensor de temperatura Dallas DS18B20 (Figura 28) trabalha com precisão de 9 a 12 bits, e sua comunicação acontece através de um barramento 1-Wire, que, por definição, requer apenas uma linha de dados (e terra) para comunicação com o equipamento mestre (Servidor de Borda).

A faixa de temperatura que o mesmo opera vai de -55°C a +125°C, e possui uma precisão de ±0,5°C ao longo de -10°C a +85°C. Além disso, o DS18B20 pode derivar energia diretamente da linha de dados (”modo parasita”), eliminando a necessidade de uma fonte de alimentação. Em situações nas quais o ambiente onde será instalada a rede 1-Wire estiver submetido a ruidos elétricos (interferências) o uso do modo parasita não é recomendado.

O código de identificação utilizado pelo equipamento mestre para acesso aos DS18B20 presentes em uma mesma rede 1-Wire é de 64 bits.

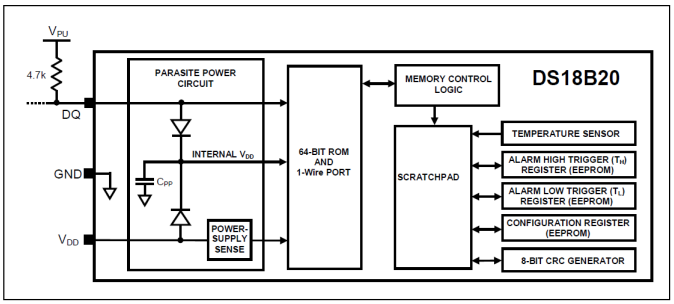


Figura 28: Diagrama de blocos do DS18B20
(MAXIM INTEGRATED PRODUCTS, 2012)

O diagrama esquemático do sensor de temperatura é apresentado na Figura 28.

Figura 29: Sensor de temperatura encapsulado

O sensor de temperatura contempla diversos acoplamentos mecânicos: (i) inércia térmica com o objetivo de evitar falsos positivos decorrentes de flutuações transitórias de temperatura no meio (vide Figura 29); (ii) encapsulamento metálico, com vedação resistente a temperaturas elevadas (150ºC) (vide Figura 30); (iii) grid de sensores cujo objetivo é permitir o acompanhamento da evolução da temperatura em diferentes pontos de um determinado equipamento (vide Figura 31).

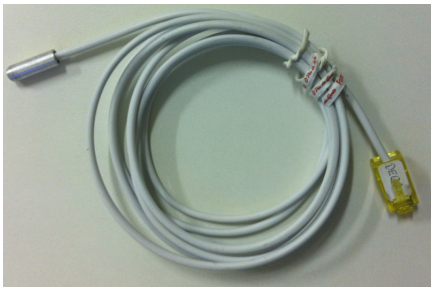


Figura 30: Sensor de temperatura

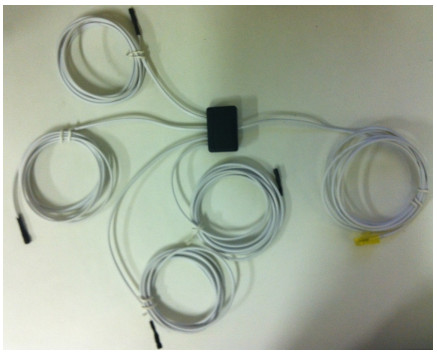


Figura 31: Grid de Sensores de temperatura

Sensor de Umidade

O sensor de umidade relativa desenvolvido opera na faixa de 0-100% com uma precisão de 3%. O encapsulamento projetado para o mesmo previne a incidência de luz direta e/ou água (vide Figura 32).

Figura 32: Sensor de umidade

A parte eletrônica do sensor de umidade tem por base dois semicondutores: o Honeywell HIH4000 que converte percentual de umidade em níveis de voltagem, e o DS2438 que dentre outras funções converte níveis de voltagem analógicos para o padrão digital e disponibiliza os mesmos através do protocolo 1-Wire.

O Diagrama esquemático do sensor de umidade é apresentado na Figura 33.

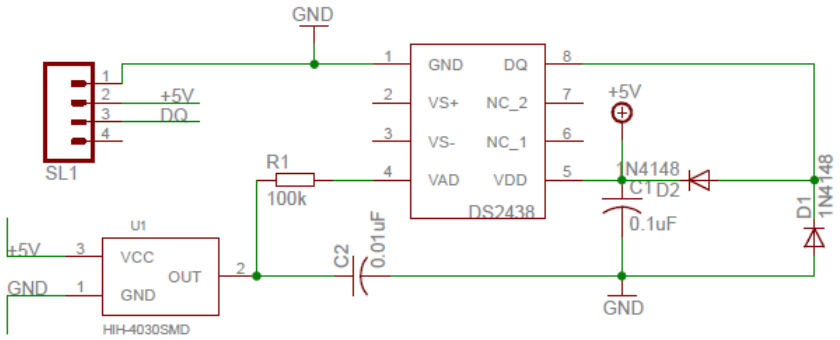


Figura 33: Sensor de Umidade - Diagrama Esquemático

Módulo para Aquisição de Dados

Em parceria com a empresa O.S.Systems[1] foi desenvolvida uma primeira solução para aquisição de dados baseada em computação embarcada, esta solução foi base para o desenvolvimento de produto comercial da mesma empresa (vide Apêndice A).

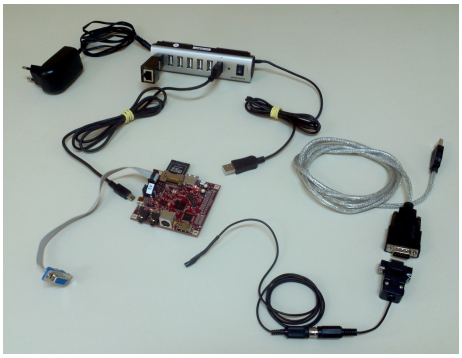


Figura 34: Solução embarcada - Aquisição de dados

Na Figura 34 é apresentada a solução baseada em uma placa da empresa Beagleboard. Esta solução realiza tanto coleta dados do meio, como atua sobre o mesmo (ativa ou desativa dispositivos) através da rede 1-Wire.

Estão em andamento estudos para customizar em vários aspectos o hardware embarcado. Entre os vários aspectos sendo otimizados, temos: custo, desempenho, consumo de energia e novas funcionalidades.

Atuador - Alerta Visual

O Alerta Visual consiste em uma dispositivo emissor de luz de elevada intensidade, visivel mesmo sob condições de iluminação diurnas e a distâncias que chegam a dezenas de metros. O dispositivo foi encapsulado em uma caixa

[1] http://ossystems.com.br/

com lente difusora de luz para facilitar sua visibilidade.

Figura 35: Alerta visual

Na Figura 35 é possível ver o arranjo mecânico desenvolvido no qual estão integrados os recursos para interoperabilidade com a tecnologia 1-Wire com aqueles necessários para estabilidade mecânica e ótica da solução. Na Figura 36 é apresentado o diagrama esquemático.

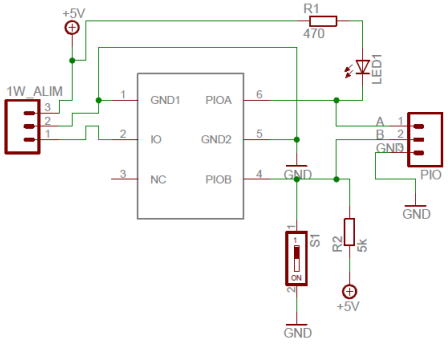


Figura 36: Alerta visual - Diagrama esquemático

ANEXO B MÓDULO DE GERENCIAMENTO

Este módulo tem por objetivo permitir ao usuário um gerenciamento confortável das configurações do Servidor de Contexto. O mesmo provê facilidades para que sejam especificados os diferentes aspectos dos sensores e atuadores, bem como informações dos equipamentos cujo contexto está sendo aquisitado.

As principais funcionalidades deste módulo podem ser resumidas como a seguir:

Procedimento de Login

No que diz respeito ao seu gerenciamento o EXEHDA-UC contempla dois tipos de usuários: (i) usuário administrador, que tem acesso aos procedimentos de cadastramento em geral (sensores, atuadores, equipamentos, usuários regulares, etc.); e (ii) usuário regular, o qual cadastra interesses de uso indicando equipamentos, regras, sensores, atuadores e o correspondente agendamento no tempo (Figura 37).

Figura 37: EXEHDA-UC: Mod. Gerenciamento - Login

A validação de acesso é feita a partir da especificação do usuário e sua senha, sendo provido para aumento da segurança de acesso um recurso de CAPTCHA [1]. É também oferecido ao usuário um recurso de recuperação de senhas.

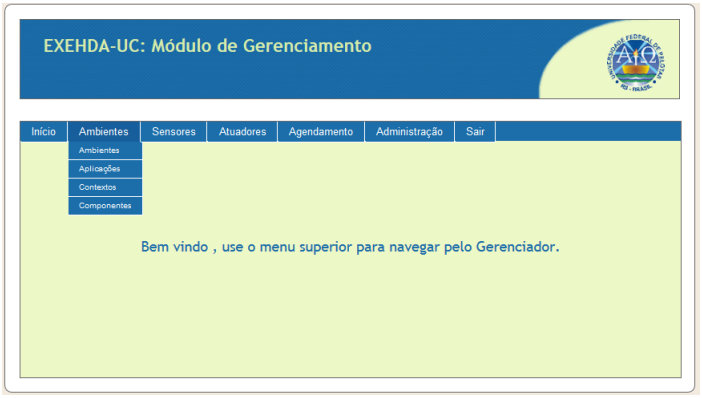


Figura 38: EXEHDA-UC: Mod. Gerenciamento - Ambientes

Procedimento de Cadastro de Ambientes

Através deste procedimento o usuário especifica os vários ambientes que compõem a infraestrutura de computação ubíqua, associando aos mesmos contextos de interesse. A Figura 38 apresenta essas opções na interface do Gerenciador.

O emprego de ambientes tem por objetivo auxiliar na localização espacial dos sensores e atuadores ativos. Considerando o número de dispositivos envolvidos, por definição do usuário, um ambiente pode ser um prédio, uma sala e/ou um equipamento.

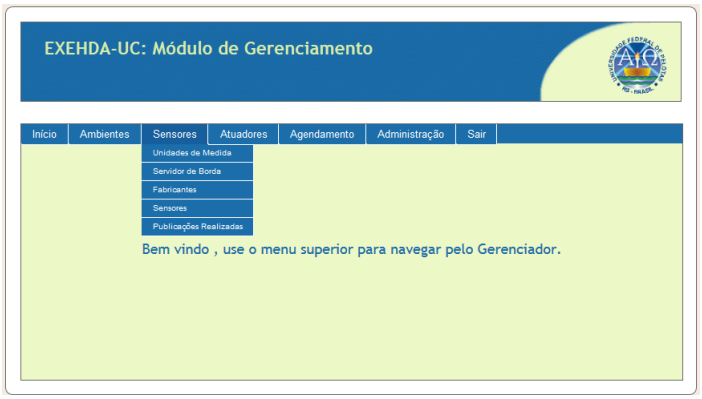


Figura 39: EXEHDA-UC: Mod. Gerenciamento - Sensores

Um contexto de interesse é formado por um ou mais sensores cujos valores são manipulados por uma regra de processamento contextual. Esta regra é

[1] http://www.captcha.net/

disparada sempre que um dos sensores envolvidos tiver seu valor publicado pelo Servidor de Borda. A critério do usuário o disparo da regra pode ficar atrelado a um subgrupo do total de sensores envolvidos no contexto de interesse.

Procedimento de Cadastro de Sensores

Empregando este procedimento o usuário pode definir os aspectos relacionados aos sensores ativos no seguimento do ambiente ubíquo sob responsabilidade do Servidor de Contexto. Estes aspectos tem tanto finalidade documentacional, bem como de especificação dos procedimentos de coleta de informações contextuais realizados pelos Servidores de Borda junto aos sensores (vide Figura 39).

Nesta perspectiva, podem ser especificadas as unidades de medida pertinentes aos diversos sensores, por exemplo graus centígrados, umidade relativa, etc.

Os Servidores de Borda associados ao Servidor de Contexto também são registrados por este procedimento.

É possível identificar os fabricantes dos sensores, bem como as características operacionais previstas para os mesmos em função das aplicações.

Por fim, através deste procedimento também é disponibilizado acesso as informações armazenadas no RIC, sem nenhum tratamento.

Figura 40: EXEHDA-UC: Mod. Gerenciamento - Atuadores

Procedimento de Cadastro de Atuadores

Este procedimento, por sua vez, oferece ao usuário a opção de cadastrar os atuadores ativos. De forma análoga aos sensores são identificados os servidores de borda responsáveis pelos atuadores, os fabricantes, os dados dos atuadores propriamente ditos, e um acesso ao registro das atuações realizadas, conforme Figura 40.

Procedimento de Agendamento

Considerando uma política de uso não exclusivo dos equipamentos por parte dos usuários, este procedimento disponibiliza mecanismos para agendamento através do qual é feito um planejamento de uso e providas informações para o Módulo de Comunicação, de tal modo que as mensagens de alerta, caso necessárias, sejam enviadas para o usuário corrente, como mostra a Figura 41.

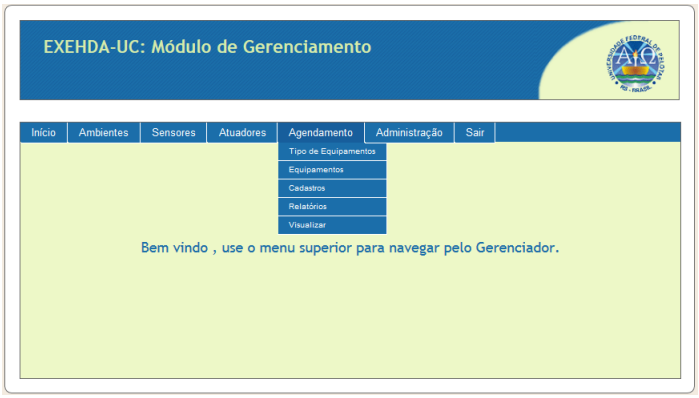


Figura 41: EXEHDA-UC: Mod. Gerenciamento - Agendamento

Neste procedimento são oferecidas interfaces para visualização dos agendamentos realizados, sua criação e/ou edição através de interfaces baseadas em calendário (vide Figuras 42 e 43).

Figura 42: EXEHDA-UC: Mod. Gerenciamento - Agendamento - Relatório

Figura 43: EXEHDA-UC: Mod. Gerenciamento - Agendamento - Mapa

Procedimento Administrativo

O Procedimento Administrativo é de uso exclusivo do usuário Administrador, sendo possível através do mesmo especificar o layout dos menus da interface do gerenciador, cadastrar perfis de acesso aos menus e usuários (vide Figura 44).

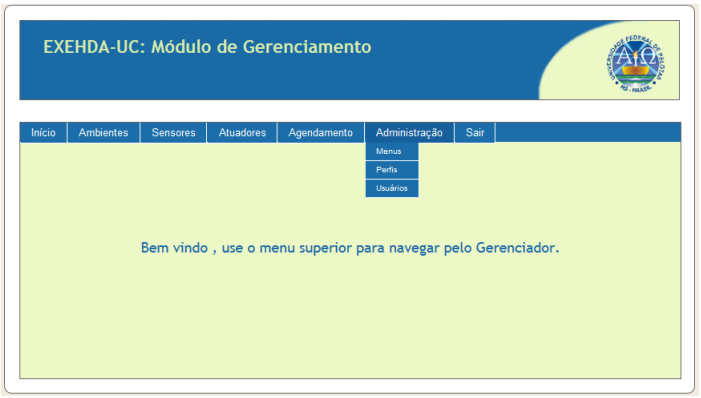


Figura 44: EXEHDA-UC: Mod. Gerenciamento - Administração

Considerando o caráter eclético da comunidade usuária do LDAS formada por estudantes, pesquisadores, professores, técnicos administrativos, este procedimento se mostra oportuno no gerenciamento das diversas possibilidades de acesso às funcionalidade do Gerenciador do EXEHDA-UC. Por exemplo, a Figura 45, apresenta a interface típica de um usuário regular, ao qual é oferecida a opção de Agendamento.

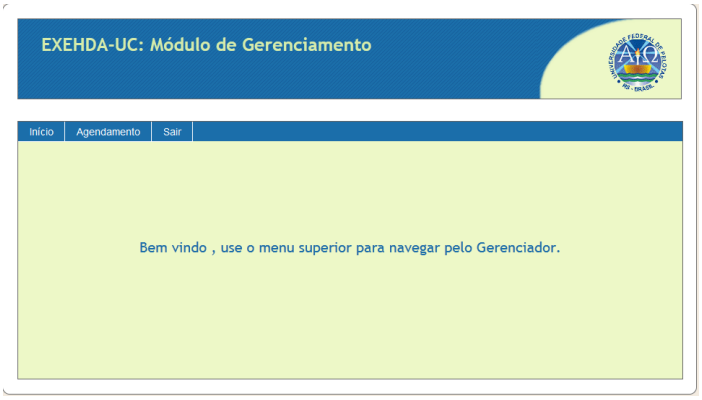


Figura 45: EXEHDA-UC: Mod. Gerenciamento - Administração - Usuário Regular