UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

JOÃO LADISLAU BARBARÁ LOPES

Uma Arquitetura para Provimento de Ciência de Situação Direcionada às Aplicações Ubíquas na Infraestrutura da Internet das Coisas

Tese apresentada como requisito parcial
para a obtenção do grau de
Doutor em Ciência da Computação

Prof. Dr. Cláudio Fernando Resin Geyer
Orientador

Porto Alegre, dezembro de 2016

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Lopes, João Ladislau Barbará

Uma Arquitetura para Provimento de Ciência de Situação Direcionada às Aplicações Ubíquas na Infraestrutura da Internet das Coisas / João Ladislau Barbará Lopes. – Porto Alegre: PPGC da UFRGS, 2016.

123 f.: il.

Tese (doutorado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2016. Orientador: Cláudio Fernando Resin Geyer.

1. Computação ubíqua. 2. Ciência de situação. 3. Internet das coisas. 4. Processamento híbrido do contexto. I. Geyer, Cláudio Fernando Resin. II. Título.

Dedico esta tese à memória de meu pai João Gentil Lopes.

AGRADECIMENTOS

Expresso meu especial agradecimento à minha mãe Maria Barbará Lopes, minha esposa Patrícia Prietsch Lopes e minha filha Vitória Prietsch Lopes. Cada uma, à sua maneira, me apoiou durante o desenvolvimento deste trabalho.

Ao meu orientador Cláudio Geyer por ter aceito a orientação desta tese. Agradeço pelo seu suporte ao longo do desenvolvimento deste trabalho.

Ao meu amigo Adenauer Yamin por me ajudar em todos os momentos da elaboração desta tese. Obrigado pelo seu apoio irrestrito durante várias etapas da minha vida acadêmica, sendo meu professor na graduação e especialização, e meu orientador no mestrado.

Aos grupos de pesquisa GPPD-UFRGS, LUPS-UFPel e G3PD-UCPel por proverem um excelente ambiente para o desenvolvimento desta pesquisa. Aos colegas destes grupos agradeço pelas importantes contribuições ao longo desta tese.

Em particular, agradeço ao colega Rodrigo Souza pela troca de ideias que norteou algumas das decisões de projeto que eram comuns aos nossos trabalhos. Agradeço também aos colegas Alexandre Souza, Douglas Scheunemann, Patrícia Davet e Roger Machado pelo suporte no desenvolvimento dos cenários de uso deste trabalho.

Ao IFSul, especialmente ao CAVG, agradeço pelo apoio institucional a este trabalho.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
AVU	Ambiente Virtual do Usuário
BDA	Base de Dados das Aplicações
CC/PP	Composite Capabilities/Preference Profiles
CIB	Cell Information Base
DoS	Denial of Service
ECA	Evento-Condição-Ação
Embrapa	Empresa Brasileira de Pesquisa Agropecuária
ETB	Estação Experimental Terras Baixas da Embrapa
EXEHDA	Execution Environment for Highly Distributed Applications
FC	Frequência Cardíaca
FIFO	First In, First Out
FFT	Fast Fourier Transform
GPPD	Grupo de Processamento Paralelo e Distribuído
IoT	Internet of Things (Internet das Coisas)
ISAM	Infraestrutura de Suporte às Aplicações Móveis
JSON	JavaScript Object Notation
LabLeite	Laboratório de Qualidade do Leite
LASO	Laboratório de Análise de Sementes Oficial
LUPS	Laboratory of Ubiquitous and Parallel Systems
MAPA	Ministério da Agricultura, Pecuária e Abastecimento
OWL	Web Ontology Language
OX	Objeto eXehda
plenUS	plentiful Ubiquitous Systems
RBQL	Rede Brasileira de Laboratórios de Qualidade do Leite
RDF	Resource Description Framework

RDQL	RDF Data Query Language
REST	REpresentational State Transfer
RHIC	Repositório Híbrido de Informações de Contexto
SAUI	Situation-aware Architecture for Ubiquitous applications in the Internet of things
SISPEL	Sistema de Pesquisa e Desenvolvimento em Pecuária Leiteira
SPARQL	SPARQL Protocol and RDF Query Language
SQL	Structured Query Language
UbiComp	Computação Ubíqua
UFPel	Universidade Federal de Pelotas
UFRGS	Universidade Federal do Rio Grande do Sul
URI	Uniform Resource Identifier
UTI	Unidade de Tratamento Intensivo
HTTP	HyperText Transfer Protocol
XML	eXtensible Markup Language

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

A Computação Ubíqua (UbiComp) foi introduzida na década de 90 como a área de pesquisa que estuda a integração da tecnologia às tarefas cotidianas, com a intenção de proporcionar aos usuários a manutenção do foco em suas atividades, reduzindo a necessidade de seu envolvimento com a gerência da infraestrutura computacional.

Considerando esta perspectiva de integração das tecnologias, exigindo o mínimo de participação do usuário, um dos desafios centrais de pesquisa para contemplar essa proposta da UbiComp é a Ciência de Situação. A construção do suporte à Ciência de Situação para as aplicações ubíquas envolve a representação do contexto, a coleta, o armazenamento e o processamento dos dados contextuais, bem como a identificação e disseminação das situações, de forma autônoma.

Dentre as maneiras que tem sido consideradas para materializar a UbiComp, vem se destacando a Internet das Coisas (*Internet of Things* - IoT). Esta abordagem utiliza a Internet como principal meio para interoperação entre dispositivos computacionais. Nesse sentido, a IoT tem como premissa unir o mundo físico ao digital, preconizando a ideia do "tudo conectado", criando assim uma rede de objetos incorporados ao ambiente de forma ubíqua.

Deste modo, esta tese tem como objetivo conceber uma arquitetura para provimento de Ciência de Situação às aplicações ubíquas, na perspectiva da infraestrutura da Internet das Coisas. A arquitetura, denominada SAUI (*Situation-aware Architecture for Ubiquitous applications in the Internet of things*), é concebida considerando os trabalhos previamente desenvolvidos pelo grupo de pesquisa GPPD/UFRGS, particularmente o *middleware* EXEHDA (*Execution Environment for Highly Distributed Applications*).

Entende-se como contribuições centrais desta tese: (i) a definição de uma abordagem híbrida para representação e processamento do contexto, visando à identificação de situações de interesse das aplicações; e (ii) a concepção de uma arquitetura distribuída, baseada em eventos e regras, visando o suporte à Ciência de Situação das aplicações ubíquas, considerando a infraestrutura provida pela IoT.

As funcionalidades da Arquitetura SAUI são avaliadas através de cenários de uso nas áreas de agropecuária e saúde, sendo caracterizados os protótipos desenvolvidos, as tecnologias empregadas e os testes realizados. Os resultados obtidos corroboram com a abordagem da Arquitetura SAUI de considerar um suporte concomitante para operação distribuída, tratamento autônomo dos dados contextuais baseado em eventos e regras, e processamento híbrido do contexto, visando à identificação de situações de interesse das aplicações.

Palavras-chave: Computação ubíqua, ciência de situação, internet das coisas, processamento híbrido do contexto.

An Architecture for Situation Awareness Targeted to Ubiquitous Applications in the Infrastructure of the Internet of Things

ABSTRACT

The Ubiquitous Computing (UbiComp) was introduced in the 90s as the area of research that studies the integration of technology in the everyday tasks, with the objective to keep the focus of the users in their activities, reducing the need of their involvement in the management of the computational infrastructure.

Considering this perspective of technologies integration, requiring minimal user participation, one of the central research challenges is the situation awareness. The construction of the support to situation awareness for ubiquitous applications involves the representation of context; the acquisition, storage and processing of contextual data; and the identification and dissemination of situations, in autonomous way.

Among the ways that have been considered to materialize UbiComp, has been highlighting the IoT (Internet of Things). This approach uses the Internet as the main means for interoperation between computing devices. In this sense, the IoT has the premise of join the physical and digital worlds, contemplating the idea of "everything connected", thus creating a network of objects embedded in the environment ubiquitously.

Thus, this thesis aims to design an architecture for providing situation awareness to ubiquitous applications, considering the IoT infrastructure. The architecture, called SAUI (Situation-aware Architecture for Ubiquitous applications in the Internet of things), is designed considering the work previously developed by the research group GPPD/UFRGS, particularly EXEHDA middleware (Execution Environment for Highly Distributed Applications).

It is considered as central contributions of this thesis: (i) the definition of a hybrid approach for modeling and processing of context, aiming at identification of situations of interest of the applications; and (ii) the design of a distributed architecture, driven by events and rules, aiming at supporting the situation awareness of ubiquitous applications, considering an IoT infrastructure.

The funcionalities of SAUI architecture are evaluated through usage scenarios in the areas of agriculture and healthcare. It was characterized the developed prototypes, the technologies used, and the tests performed. The results corroborate with the approach of SAUI architecture, which considers a concomitant support for distributed operation, autonomous handling of context based on events and rules, and hybrid processing of context, aiming at identification of situations.

Keywords: Ubiquitous computing, situation awareness, internet of things, hybrid processing of context.

1 INTRODUÇÃO

O clássico artigo de WEISER (1991), considerado precursor da Computação Ubíqua (UbiComp), caracteriza este paradigma computacional como a área de pesquisa que estuda a integração da tecnologia às tarefas cotidianas, com a intenção de proporcionar aos usuários a manutenção do foco em suas atividades, reduzindo a necessidade de seu envolvimento com a gerência da infraestrutura computacional (CACERES; FRIDAY, 2012) (BAUER; NEWMAN; KIENTZ, 2014).

Considerando esta premissa da UbiComp de potencializar o foco do usuário em suas atividades, minimizando a exigência de atenção com a infraestrutura computacional, um dos desafios centrais de pesquisa é o provimento de Ciência de Situação. Nessa perspectiva, uma situação corresponde a uma visão de alto nível e abrangente dos contextos de interesse das aplicações, que pode ser utilizada por estas em seu processo de tomada de decisão para disparo de ações sobre o ambiente. Esta visão é decorrente da identificação de situações, que podem ser compostas a partir de dados contextuais coletados por diferentes sensores distribuídos no ambiente ubíquo (YE; DOBSON; MCKEEVER, 2012) (BIBRI, 2015).

A Ciência de Situação considera que as aplicações, em função de modificações nos seus contextos de interesse, podem ajustar seu comportamento. Essa classe de sistemas computacionais, cientes de situação, abre perspectivas para o desenvolvimento de aplicações mais ricas, elaboradas e complexas, que exploram a natureza dinâmica das infraestruturas computacionais e a mobilidade do usuário (KAKOUSIS; PASPALLIS; PAPADOPOULOS, 2010) (DA; DALMAU; ROOSE, 2011) (KNAPPMEYER et al., 2013).

A revisão de literatura aponta que a construção do suporte à Ciência de Situação, na perspectiva das aplicações ubíquas, apresenta diversos desafios de pesquisa, dentre eles: (i) a definição de um modelo de contexto com elevada expressividade e capacidade de raciocínio; (ii) a coleta do contexto a partir de fontes heterogêneas e distribuídas; (iii) o processamento das informações de contexto adquiridas e a correspondente ação sobre a infraestrutura; e (iv) a disseminação das situações identificadas a consumidores interessados de forma distribuída e no momento oportuno (BETTINI et al., 2010) (BELLAVISTA et al., 2012) (SILVA et al., 2012) (YE et al., 2015).

A proposta da UbiComp vem se concretizando em função do rápido avanço tecnológico dos dispositivos móveis, redes sem fio, redes de sensores, sistemas e dispositivos inteligentes, sistemas distribuídos, sistemas autônomos, computação em nuvem e outras tecnologias relacionadas (BIBRI, 2015).

Nesse cenário, a Internet das Coisas (*Internet of Things* - IoT) tem contribuído para materializar a UbiComp. Esta abordagem utiliza a Internet como principal meio para interoperação entre dispositivos computacionais. A Internet das Coisas caracteriza um

cenário onde qualquer "coisa" pode se comunicar através da Internet, possuindo uma identificação única e sem a obrigatoriedade de uma intervenção humana. É um conceito que une o mundo físico ao digital e que preconiza a ideia do "tudo conectado", criando uma rede de objetos inteligentes incorporados ao ambiente de forma ubíqua. Desta forma, a IoT tem sido considerada como uma abordagem para promover a ubiquidade das soluções computacionais, principalmente por suprir as demandas da UbiComp em relação aos aspectos infraestruturais (PERERA et al., 2014) (WHITMORE; AGARWAL; DA XU, 2015).

O volume de dados de contexto que necessita de algum tratamento é potencialmente muito elevado na infraestrutura de IoT para suporte às aplicações ubíquas. Para que esses dados resultem em informações que possam ser utilizadas por diferentes aplicações é necessário disponibilizá-los em formatos adequados e padronizados, usando linguagens bem definidas. Isso permite que sejam empregados métodos de raciocínio sobre os dados. Nesse sentido, O emprego de abordagens híbridas para representação e processamento do contexto vêm se constituindo em uma área de pesquisa desafiadora na Computação Ubíqua. Esta abordagem visa integrar diferentes modelos de representação e técnicas de raciocínio a fim de obter abstrações de contexto de alto nível (BIBRI, 2015) (YE et al., 2015) (RAZZAQUE et al., 2016).

Considerando o cenário exposto, o tema central desta tese é a Ciência de Situação para aplicações ubíquas, na perspectiva da infraestrutura da Internet das Coisas. De forma específica, a tese aborda a representação dos dados contextuais e os procedimentos relacionados ao tratamento dos contextos de interesse das aplicações, bem como a consequente identificação de situações que podem gerar mudanças no comportamento das aplicações.

Desta forma, considerando a premissa central da UbiComp, bem como o cenário da IoT, tem-se como objetivo conceber uma arquitetura para provimento de Ciência de Situação, com organização distribuída, tratamento autônomo do contexto baseado em eventos e regras, com suporte a processamento híbrido do contexto, visando à identificação de situações. Essa arquitetura, denominada SAUI (*Situation-aware Architecture for Ubiquitous applications in the Internet of things*), é concebida considerando os trabalhos previamente desenvolvidos pelo grupo de pesquisa GPPD/UFRGS (Grupo de Processamento Paralelo e Distribuído da Universidade Federal do Rio Grande do Sul), particularmente o *middleware* EXEHDA (*Execution Environment for Highly Distributed Applications*) (YAMIN, 2004) (LOPES et al., 2014).

Este capítulo tem por objetivo contextualizar a tese, apresentando suas motivações, problema de pesquisa e objetivos, bem como aspectos da organização do texto.

1.1 Motivação e Problema de Pesquisa

A UbiComp vem sendo gradativamente concretizada em função do avanço tecnológico de diferentes paradigmas computacionais, como a computação distribuída, computação autonômica, computação em grade, computação em nuvem. Além disso, as recentes evoluções nas tecnologias associadas a dispositivos móveis e redes sem fio, bem como à infraestrutura de Internet das Coisas, como redes de sensores e atuadores, sistemas e dispositivos inteligentes, têm contribuído para tornar realidade a proposta da Computação Ubíqua (BIBRI, 2015).

Essas tecnologias que viabilizam a infraestrutura para a UbiComp são convergentes e se integram sinergicamente no cotidiano dos usuários. Tendo em vista a dinamicidade

decorrente desta integração, a Ciência de Situação, caracterizada pela capacidade de identificar situações a partir do processamento dos contextos de interesse das aplicações, torna-se um aspecto central para a consecução das premissas da UbiComp relacionadas a um gerenciamento autônomo da infraestrutura computacional, minimizando o envolvimento do usuário (YE; DOBSON; MCKEEVER, 2012).

A infraestrutura computacional para UbiComp tende a ser distribuída, heterogênea e dinâmica. A distribuição é caracterizada pelo fato da infraestrutura computacional ter seus recursos dispersos em diferentes locais. As aplicações ubíquas devem ser capazes de executar em ambientes com elevada heterogeneidade, constituídos por diferentes tipos de dispositivos, com vários sistemas operacionais e interfaces com o usuário. A característica de dinamicidade se deve ao fato dos elementos de contexto, dispositivos e componentes de software mudarem com frequência seu status no ambiente ubíquo (COSTA et al., 2010) (RAYCHOUDHURY et al., 2013).

Nesse cenário, a Internet das Coisas contribui para materializar as premissas da Computação Ubíqua, no que tange ao fornecimento de serviços computacionais de forma o mais transparente possível, independente de infraestrutura e tecnologias empregadas, não necessitando que o usuário tenha conhecimento de todo o sistema e/ou elementos deste ambiente computacional (DELICATO; PIRES; BATISTA, 2013) (GUBBI et al., 2013a).

Ainda, a revisão da literatura aponta problemas relacionados ao uso de estruturas de dados *ad-hoc* para modelagem do contexto. Muitos projetos também empregam modelos informais para representar o contexto, elevando a complexidade no desenvolvimento de aplicações, bem como reduzindo a possibilidade de reuso e de manutenção dos componentes de software. O uso de modelos informais ainda reduz a capacidade de raciocínio e compartilhamento de conhecimento entre as aplicações (BIBRI, 2015) (ALEGRE; AUGUSTO; CLARK, 2016).

Na perspectiva da UbiComp, aplicações que tenham a capacidade de ajustar-se de forma autônoma às mudanças no ambiente podem tornar as atividades cotidianas do usuário mais confortáveis e, por consequência, melhorar sua qualidade de vida, podendo fazer com que suas necessidades sejam consideradas, muitas vezes, sem uma solicitação explícita por parte do mesmo. Alguns dos principais problemas relacionados ao desenvolvimento de aplicações com essa característica de ubiquidade dizem respeito à necessidade de uma infraestrutura para suporte à Ciência de Situação que possua capacidade autônoma de tratamento dos dados contextuais. Nesse sentido, um suporte à Ciência de Situação contribui para reduzir o esforço de desenvolvimento, monitorando proativamente as condições de contexto e fazendo com que o desenvolvedor fique desobrigado de gerenciar aspectos como coleta, armazenamento, processamento e raciocínio de dados contextuais (HUEBSCHER; MCCANN; HOSKINS, 2007) (KNAPPMEYER et al., 2013).

Portanto, considerando os diferentes contextos de interesse das aplicações ubíquas e o decorrente aumento do esforço computacional no seu tratamento, prover uma arquitetura para suporte às aplicações cientes de situação que considere a infraestrutura provida pela IoT, constitui um desafio de pesquisa para a área de UbiComp. Nesta perspectiva, esta tese investiga como problema central de pesquisa:

Como deve ser uma arquitetura para provimento de Ciência de Situação direcionada às aplicações ubíquas, considerando a infraestrutura da Internet das Coisas?

1.2 Objetivos

Considerando o problema de pesquisa, entende-se como objetivo central desta tese a concepção de uma arquitetura direcionada ao provimento de Ciência de Situação para as aplicações ubíquas na infraestrutura da Internet das Coisas.

No que diz respeito a operação da arquitetura, é empregada uma abordagem distribuída, baseada em eventos e regras para as diferentes etapas de tratamento do contexto. Nesse sentido, a arquitetura deve prover mecanismos para coletar dados contextuais, considerando a característica de distribuição das infraestruturas da IoT, bem como processar os contextos de interesse das aplicações e as decorrentes situações identificadas.

No que tange à identificação de situações, é concebida uma abordagem híbrida para representação e processamento do contexto.

Na representação do contexto, a abordagem concebida permite combinar modelos semânticos e sintáticos, explorando de forma sinérgica suas características e minimizando suas limitações. Com isso, busca-se potencializar o processamento do contexto, através da construção de modelos que se aproximem do domínio das aplicações.

Com relação ao processamento do contexto, a abordagem híbrida permite processar os dados armazenados conforme o modelo de representação do contexto, combinando técnicas para identificação de situações baseadas tanto em especificação de regras como em aprendizagem. Com isso, pretende-se viabilizar a construção de mecanismos para raciocínio sobre o contexto, qualificando o processo de identificação de situações.

Considerando o objetivo central da tese, as seguintes metas específicas foram contempladas:

- revisar, na perspectiva da Computação Ubíqua, os principais conceitos e o estado da arte nos temas relacionados à Ciência de Contexto, Ciência de Situação e Internet das Coisas;
- definir uma abordagem híbrida para representação e processamento do contexto que permita a identificação de situações;
- modelar uma arquitetura para suporte à Ciência de Situação das aplicações ubíquas em diferentes domínios, considerando a infraestrutura computacional provida pela IoT;
- especificar os módulos a serem providos pela arquitetura, considerando os aspectos de distribuição e o tratamento autônomo do contexto através de eventos e regras;
- prototipar a arquitetura no âmbito do escopo de pesquisa do GPPD/UFRGS, particularmente do projeto ISAM/EXEHDA;
- validar as funcionalidades da arquitetura através de cenários de uso.

Considerando os objetivos, entende-se como contribuições centrais desta tese: (i) a definição de uma abordagem híbrida para representação do contexto que permite combinar modelos semânticos e sintáticos, em função da necessidade do domínio de problema que está sendo tratado, explorando de forma sinérgica suas características e minimizando suas limitações; (ii) a construção de uma abordagem híbrida para processamento do contexto que viabilize a combinação de técnicas baseadas em especificação e aprendizagem visando à identificação de situações; e (iii) a concepção de

uma arquitetura para suporte à Ciência de Situação das aplicações ubíquas na perspectiva da infraestrutura de Internet das Coisas, cujo tratamento dos dados de contexto ocorre de forma autônoma, combinando a sinergia de uma arquitetura distribuída com uma abordagem baseada em eventos e regras.

Na perspectiva do *middleware* EXEHDA, a concepção da Arquitetura SAUI pretende contribuir com o Subsistema de Reconhecimento de Contexto e Adaptação, adicionando recursos relacionados à Ciência de Situação.

1.3 Estrutura do Texto

O texto desta tese é composto por seis capítulos. No Capítulo 1 foram apresentadas as motivações, o problema de pesquisa e os objetivos desta tese. O conteúdo dos demais capítulos está resumido a seguir.

O Capítulo 2 apresenta uma revisão bibliográfica, incluindo conceitos sobre Computação Ubíqua, Ciência de Contexto e de Situação e Internet das Coisas. Ainda, são descritas as características e funcionalidades do *middleware* EXEHDA.

O Capítulo 3 sintetiza a revisão de literatura que buscou identificar os trabalhos relacionados a esta tese, sendo apresentada uma discussão dos mesmos, considerando os objetivos e premissas de concepção da Arquitetura SAUI.

No Capítulo 4 é descrita a concepção da Arquitetura SAUI, sendo caracterizada a abordagem híbrida, baseada em eventos e regras para processamento do contexto, bem como a organização e funcionalidades da arquitetura.

A avaliação das funcionalidades da Arquitetura SAUI é apresentada no Capítulo 5, através de cenários de uso relacionados às áreas de agropecuária e saúde.

Por fim, no Capítulo 6 são apresentadas as conclusões da tese, sendo destacadas as principais contribuições e publicações realizadas, bem como os trabalhos futuros que podem ser desenvolvidos na continuidade desta pesquisa.

2 FUNDAMENTOS CONCEITUAIS

A sistematização de conceitos apresentada neste Capítulo foi feita considerando o tema central de pesquisa desta tese. Nesse sentido, são apresentados conceitos sobre Computação Ubíqua, Ciência de Contexto e de Situação e Internet das Coisas. Também, considerando a proposta de instanciar da Arquitetura SAUI no âmbito do escopo de pesquisa do *middleware* EXEHDA, são descritas as principais características e funcionalidades do EXEHDA.

2.1 Computação Ubíqua

A UbiComp contempla uma visão de futuro na qual o acesso do usuário aos recursos do ambiente computacional deve ocorrer de forma transparente e calma (COSTA; YAMIN; GEYER, 2008). Nessa perspectiva, o sistema ubíquo deve preservar o foco do usuário em suas atividades cotidianas e não no computador. Estes aspectos constituem uma das motivações centrais da pesquisa em UbiComp, sendo uma premissa central que relaciona o acesso do usuário ao ambiente computacional quando e onde quiser e com qualquer dispositivo.

A Computação Ubíqua idealiza a existência de ambientes carregados de recursos computacionais que se integram ao cotidiano dos usuários. Considerando a escala de recursos desses ambientes, a premissa é automatizar a coleta de informações contextuais, reagindo a possível ocorrência de situações. Nesse sentido, a Ciência de Situação constitui um dos principais desafios de pesquisa na UbiComp (KRUMM, 2010).

A automatização das operações relacionadas à Ciência de Situação se baseia em informações coletadas na infraestrutura computacional e no ambiente em que os sistemas ubíquos têm interesse. Estas informações são tratadas através de mecanismos que têm suporte à Ciência de Contexto (KHARGHARIA; HARIRI; YOUSIF, 2008).

Neste sentido, a noção de tecnologia calma (*Calm Technology*) (KRUMM, 2010) corresponde ao modo de operação através do qual o sistema pode ser composto por muitos e diferentes recursos computacionais que ajustam a operação para melhor atender o usuário, a aplicação ou metas exigidas dele, tendo como base a Ciência de Situação.

A infraestrutura computacional para UbiComp pode ser constituída por conjuntos heterogêneos de dispositivos computacionais, como microcomputadores, *smartphones*, e combinações de sensores e atuadores, que monitoram e disparam ações sobre um determinado ambiente. Estes dispositivos podem usar uma combinação de redes cabeadas e sem fio, consumindo energia a partir de fontes usuais de fornecimento de eletricidade ou baterias. Para fornecer um ambiente de computação calma, o usuário não estará interessado em como essa arquitetura heterogênea está instalada, interage, ou é mantida (LALANDA; MCCANN; DIACONESCU, 2013).

Nesse cenário, a Internet das Coisas tem sido considerada como uma abordagem para promover a ubiquidade das soluções computacionais. A IoT aponta para um cenário no qual objetos, como sensores e atuadores, devem possuir a capacidade de se comunicar e transferir dados através da Internet com o mínimo de intervenção humana (PERERA et al., 2014).

A gradativa integração da Computação Ubíqua e da Internet das Coisas às atividades cotidianas tem gerado um crescente número de dados de contexto. A utilização deste grande volume de dados contextuais pode propiciar o desenvolvimento de novas aplicações que exploram a Ciência de Situação. Estas aplicações, que tem a capacidade de ajustar seu comportamento a mudanças no ambiente com a mínima intervenção humana, introduzem novos desafios para os desenvolvedores, principalmente relacionados a como implementar uma solução eficiente para otimizar o uso das informações de contexto para identificação de situações (LI et al., 2015).

Nesse sentido, a revisão bibliográfica realizada aponta a existência de três abordagens típicas para o desenvolvimento de aplicações ubíquas cientes de situação: (i) cada aplicação obtém, processa e usa as informações de contexto de seu interesse; (ii) algumas bibliotecas e ferramentas que visam à aquisição e processamento do contexto podem ser adicionadas e reusadas para construção das aplicações; e (iii) as aplicações são construídas com base em *middlewares* que proveem suporte à coleta, processamento e disseminação das informações de contexto (LI et al., 2015) (ALEGRE; AUGUSTO; CLARK, 2016).

A abordagem baseada em *middleware* pode reduzir a complexidade do desenvolvimento de aplicações ubíquas cientes de situação, por prover suporte para aquisição, modelagem, armazenamento e processamento do contexto, dentre outros aspectos. Assim, o emprego de um *middleware* pode liberar os desenvolvedores de preocupações relacionadas ao tratamento do contexto, possibilitando que estes mantenham o foco na definição das regras de negócio e no desenvolvimento das funcionalidades específicas das aplicações (PERERA et al., 2014) (LI et al., 2015).

2.2 Ciência de Contexto

Contexto é um conceito originalmente estudado pelas áreas de filosofia e linguística. Na Ciência da Computação as pesquisas envolvendo contexto e Ciência de Contexto têm sido desenvolvidas por pesquisadores que atuam em diferentes áreas.

Particularmente, na área da Computação Ubíqua os sistemas cientes de contexto envolvem uma grande quantidade de informações contextuais dinâmicas que necessitam ser adquiridas, interpretadas, processadas, disseminadas para consumidores interessados e mantidas atualizadas em diferentes repositórios. Assim, devem ser adotados mecanismos de tratamento do contexto, bem como especificados modelos para representação do contexto que sejam interoperáveis, dinâmicos e escaláveis (BIBRI, 2015).

2.2.1 Definição de Contexto e de Ciência do Contexto

Uma das primeiras definições de contexto é apresentada por SCHILIT; THEIMER (1994). Para estes autores, o contexto é descrito como a localização, as identidades de pessoas próximas, os objetos e as mudanças que ocorrem nesses objetos.

No trabalho de RYAN; PASCOE; MORSE (1997) o contexto é definido como a localização do usuário, o ambiente, a identidade e o tempo. Por sua vez, DEY (1998) define contexto como o estado emocional do usuário, o foco de atenção, a localização e a

orientação, a data e o tempo, os objetos e as pessoas no ambiente do usuário. Já HULL; NEAVES; BEDFORD-ROBERTS (1997) descrevem contexto como aspectos da situação corrente.

CHEN; KOTZ (2000) definem contexto como o conjunto de estados e configurações do ambiente que determina um comportamento da aplicação, no qual um evento ocorre e é interessante para o usuário.

O contexto é definido por YAMIN et al. (2003) como toda a informação relevante para a aplicação que pode ser obtida da infraestrutura computacional, cuja alteração em seu estado dispara um processo de adaptação na aplicação. Nessa visão, o contexto permite enfocar os aspectos relevantes para uma situação particular e ignorar outros. A aplicação explicitamente identifica e define as entidades que caracterizam seu contexto.

Embora várias definições tenham sido propostas e discutidas, uma das mais aceitas e utilizadas por pesquisadores da área é a encontrada em DEY (2001). Segundo o autor entende-se por contexto "Qualquer informação que possa ser utilizada para caracterizar a situação de entidades (pessoa, lugar ou objeto) que sejam consideradas relevantes para a interação entre um usuário e uma aplicação, incluindo o usuário e a aplicação".

Pode-se observar que a definição de DEY (2001) é abrangente quanto aos tipos de dados que podem ser considerados como contextos, sendo suficientemente ampla para incluir as diversas necessidades específicas de cada aplicação. Também, não há restrição quanto às fontes de contextos possíveis de serem utilizadas, permitindo que os dados reflitam a situação de qualquer entidade relevante para cada caso em particular (LOUREIRO et al., 2009).

O conceito de contexto pode ser aplicado em diferentes tipos de ambientes no domínio da Ciência da Computação, tais como: (i) ambiente computacional: poder de processamento, capacidade de rede, custo computacional; (ii) ambiente do usuário: localização, grupo de pessoas, estrutura social; e (iii) ambiente físico: temperatura, pressão, umidade.

Na área de Ciência da Computação, o termo Ciência de Contexto também foi cunhado por SCHILIT; THEIMER (1994), tendo sido definido por estes autores como a capacidade das aplicações coletarem informações contextuais e se ajustarem às mudanças que ocorrem no estado do contexto.

Atualmente, a Ciência de Contexto tem sido definida como a capacidade da infraestrutura computacional fornecer informações relevantes para as aplicações ajustarem seu comportamento, a partir dos dados de contexto coletados. Desta forma, a Ciência de Contexto caracteriza-se por possuir duas grandes frentes: (i) a aquisição e o tratamento dos dados que expressam informações relevantes sobre o contexto; e (ii) o ajuste do comportamento das aplicações às alterações de contexto (KNAPPMEYER et al., 2013).

2.2.2 Modelagem do Contexto

A definição do contexto possibilita a especificação do domínio do problema que está sendo tratado. Porém, definir contexto pode não ser suficiente para viabilizar o processamento computacional dos dados contextuais. Assim, a modelagem do contexto visa representar o mesmo em um formato que seja processável por máquina.

A modelagem do contexto pode contemplar diversos aspectos que são considerados no processo de construção de software, tais como: facilidade de desenvolvimento, manutenção e evolução das aplicações, separação entre código e modelo, reuso e compartilhamento.

Assim, com base nos artigos de PERTTUNEN; RIEKKI; LASSILA (2009); BETTINI et al. (2010); KNAPPMEYER et al. (2013); BIBRI (2015) e ALEGRE; AUGUSTO; CLARK (2016), esta seção descreve os requisitos para representação do contexto, apresentando as técnicas de modelagem, com suas características e limitações.

Requisitos de Modelagem

A revisão bibliográfica realizada caracteriza um conjunto de requisitos que podem ser considerados no processo de modelagem do contexto. Estes requisitos estão descritos a seguir.

Heterogeneidade e mobilidade: os modelos devem ser capazes de manipular uma grande quantidade de informações, obtidas de dispositivos heterogêneos, as quais diferem em aspectos como a taxa de atualização e a semântica dos dados. Os diferentes sensores podem obter dados do ambiente e disponibilizá-los quase instantaneamente, sem seguir uma padronização. Assim, esses dados precisam ser interpretados e preparados antes de serem disponibilizados para as aplicações. Como os dados de contexto podem ser obtidos tanto de fontes estáticas como dinâmicas, ou ainda serem gerados a partir de outros dados, o modelo deve ser capaz de representar diferentes tipos de dados de contexto e viabilizar o tratamento dos mesmos. Pelo fato de muitas aplicações serem executadas em dispositivos móveis ou utilizarem dados de fontes móveis, o modelo deve considerar a localização onde as informações foram geradas.

Relacionamento e dependência: podem existir vários relacionamentos entre as informações de contexto que são importantes para as aplicações. Assim, o modelo deve lidar com estes relacionamentos, bem como expressar a dependência que um evento associado a uma entidade pode gerar em outra.

Temporariedade: as aplicações podem precisar de acesso a dados históricos, na previsão de estados contextuais. Assim, o histórico do contexto é uma característica que deve ser representada pelos modelos. Quando a geração dos dados ocorre com uma frequência muito alta, talvez não seja viável armazenar todos os dados históricos. Nesse caso, pode ser utilizada alguma técnica que promova a agregação ou sumarização dos dados.

Raciocínio: os modelos devem ter suporte para derivar informações de contexto de maior abstração a partir de dados armazenados. As técnicas empregadas no processo de raciocínio devem ter desempenho computacionalmente viável.

Usabilidade e formalismo: os modelos devem possibilitar a representação, de forma fácil e formal, das informações de contexto a serem usadas pelas aplicações.

Provisionamento eficiente de contexto: o acesso às informações de contexto pode ser um requisito difícil de ser alcançado em modelos com grande quantidade de objetos. O modelo deve disponibilizar formas que agilizem o acesso aos objetos relevantes.

Técnicas de Modelagem

A técnica **Chave-valor** foi uma das primeiras propostas para modelagem do contexto. Esta técnica caracteriza-se por representar o contexto através de uma lista de atributos e seus correspondentes valores. Por sua vez, a técnica de **Marcação** utiliza uma estrutura de dados hierárquica, empregando *tags* com atributos e conteúdo. Esta técnica abrange linguagens de marcação amplamente utilizadas como XML e outras para domínios de problemas específicos como CC/PP (*Composite Capabilities/Preference Profiles*) que descreve perfis. Tanto a técnica **Chave-valor** como a **Marcação** possuem diversas limitações que reduzem a capacidade de processamento do contexto. As principais

limitações estão relacionadas à caracterização de relacionamentos, à verificação de consistência, e ao suporte para raciocínio sobre o contexto.

A **Orientação a Objetos** é uma técnica para representação do contexto que apresenta características típicas deste paradigma computacional, tais como: encapsulamento, reuso e herança. Esta técnica possui limitações quanto ao formalismo, já que o encapsulamento pode tornar "invisível" partes do modelo, o que não permite o completo conhecimento sobre o conteúdo do mesmo, logo a descrição do contexto pode levar a alguma dualidade de interpretação o que não é adequado para o formalismo.

A técnica **Relacional** para representação do contexto pode ser descrita por modelos Entidade-Relacionamento, sendo implementada através de bancos de dados relacionais. Esta técnica apresenta diversas características desejáveis para o processamento das informações contextuais, tais como: robustez, escalabilidade, distribuição, otimização de consultas, segurança de acesso, e disponibilidade de diversas ferramentas administrativas. Entretanto, uma limitação importante diz respeito ao fato desta técnica ter suporte limitado para raciocínio sobre dados contextuais. O processo de raciocínio pressupõe a dedução de novas informações a partir das existentes no modelo (informações implícitas), porém na técnica relacional apenas o que está representado no banco de dados é considerado válido, não sendo, portanto possível inferir outras informações.

Por fim, a técnica baseada em **Ontologia** tem sido considerada oportuna para modelagem e processamento do contexto. As suas características de expressividade, extensibilidade, padronização, interoperabilidade, e capacidade de raciocínio são altamente desejáveis. Porém, as ontologias apresentam limitação quanto à escalabilidade, ou seja, um crescente número de classes e/ou instâncias implica em uma elevação do tempo de processamento, comprometendo o desempenho. Essa limitação pode ser bastante restritiva, em função das características dos ambientes ubíquos.

Considerando esse cenário, tem-se observado propostas para integração de diferentes técnicas de modelagem do contexto, buscando aproveitar as melhores características e minimizar as limitações. Com isso, tem-se uma abordagem híbrida para o processamento do contexto, decorrente de uma combinação sinérgica das características das diferentes técnicas.

Nesse sentido, a Tabela 2.1 mostra um resumo das características e limitações das técnicas discutidas anteriormente, buscando organizar as possibilidades de combinação com maior sinergia para a obtenção de modelos híbridos. Um exemplo seria a combinação entre as técnicas **Relacional** e **Ontologia**. A primeira tem como característica positiva um bom suporte à escalabilidade, porém não viabiliza raciocínio sobre o contexto. Já uma ontologia possibilita elevada capacidade de raciocínio, mas tem seu desempenho reduzido à medida que cresce o número de classes/instâncias no modelo.

Tabela 2.1: Características e Limitações das Técnicas de Modelagem do Contexto

Técnicas	**Características**	**Limitações**
Chave-valor	Simplicidade, facilidade de uso	Suporte limitado para raciocínio
Marcação	Padronização	Suporte limitado para raciocínio
Orientação a Objetos	Encapsulamento, reuso, herança	Pouco formalismo
Relacional	Otimização de consultas, escalabilidade	Suporte limitado para raciocínio
Ontologia	Padronização, expressividade, formalismo e raciocínio	Reduzida escalabilidade

2.2.3 Aquisição do Contexto

A aquisição diz respeito a forma de obtenção das informações contextuais, a qual pode ocorrer de três formas: (i) Sensoriada, a informação é adquirida por meio de sensores, por exemplo: temperatura, nível de ruído, localização; (ii) Derivada, a informação é obtida através do tratamento dos dados brutos, em tempo de execução da aplicação. Por exemplo: média da temperatura em um determinado ambiente; (iii) Provida, a informação é explicitamente disponibilizada à aplicação, por exemplo: fornecer os dados de usuário por meio de um formulário.

O processo de aquisição do contexto pode tornar-se complexo quando envolve informação sensoriada. Isso se deve à grande variedade de sensores que podem estar envolvidos, bem como à natureza dinâmica da informação contextual.

Em função do modo como os dados são capturados, os sensores podem ser classificado em três grupos (BALDAUF; DUSTDAR; ROSENBERG, 2007) (ALEGRE; AUGUSTO; CLARK, 2016):

Sensores Físicos: correspondem aos sensores de hardware, os quais são capazes de capturar praticamente qualquer dado físico, por exemplo: sensores de temperatura, sensores de fumaça, câmeras, microfones, sistema de posicionamento global (GPS).

Sensores Virtuais: nesse grupo, a origem das informações de contexto é um software. Isso significa que é possível determinar, por exemplo, a localização de uma pessoa não somente através do uso de sistemas de localização com sensores físicos, mas também através de sensores virtuais que geram informações de localização, tais como: sistemas de agenda, reservas de viagens, correio eletrônico, mensagens instantâneas.

Sensores Lógicos: esses sensores fazem uso de um conjunto de fontes de informação, combinando sensores físicos e virtuais com informação adicional obtida em bases de dados. Por exemplo, um sensor lógico pode ser construído para detectar a posição atual de uma pessoa através da análise dos *logins* em microcomputadores e de um banco de dados mapeando os dispositivos fixos.

Desta forma, a camada de aquisição visa abstrair das aplicações a complexidade da coleta de dados, além de possibilitar a reutilização de sensores e a separação entre obtenção e utilização das informações de contexto (ALEGRE; AUGUSTO; CLARK, 2016).

2.2.4 Processamento do Contexto

O processamento do contexto abrange aspectos relacionados à interpretação, agregação, armazenamento, consulta e inferência das informações contextuais obtidas por uma etapa de aquisição, considerando o modelo do contexto empregado.

O propósito central do processamento do contexto é viabilizar a compreensão dos contextos de interesse das aplicações, apoiando o processo de identificação de situações, e a consequente tomada de decisões para ajustes no comportamento das aplicações.

O processamento do contexto deve ser implementado separadamente do código da aplicação. Assim, pode-se considerar o uso de um modelo arquitetural em camadas para dar suporte às aplicações. As camadas descritas nas próximas seções tem como referência os trabalhos de KAKOUSIS; PASPALLIS; PAPADOPOULOS (2010); BELLAVISTA et al. (2012); KNAPPMEYER et al. (2013); e BAUER; NEWMAN; KIENTZ (2014).

Interpretação

A camada de Interpretação é a primeira etapa de refinamento dos dados de contexto, elevando a possibilidade de utilização destes dados pelos componentes de software do *middleware* e/ou aplicações.

Essa camada produz um refinamento das informações contextuais de um ambiente, elevando o nível de abstração das mesmas. Também, gera informações mais elaboradas a partir de outras mais primitivas, permitindo o processamento computacional destas informações. Uma preocupação central nesse processo diz respeito ao modelo utilizado para representação do contexto. A estratégia a ser utilizada para converter os dados sensoriados em informações contextuais é definida em função da modelagem empregada para a representação do contexto.

A Interpretação trata aspectos de formatação do dado de contexto coletado, usando o modelo de contexto como base para conversão. Atua no dado coletado, ajustando seu formato, convertendo em outra escala/unidade, sem correlacionar com outros dados de contexto. Nesse sentido, a Interpretação pode ser entendida como o processo de abstração, mapeamento e manipulação das informações contextuais.

Os dados de contexto coletados são propensos a erros de leitura e ruídos de transmissão, assim para identificar essas falhas podem ser considerados critérios, como por exemplo, repetição de leituras e redundância no sensoriamento.

Agregação

A Agregação corresponde ao processo de composição de informações de contexto individuais relacionadas a uma entidade específica. Essa camada identifica diferentes origens de informações e combina contextos para produzir um resultado mais preciso e com melhor usabilidade.

As formas de Agregação abrangem: (i) utilizar identificadores para determinar quais dados devem ser correlacionados; (ii) combinar todos os dados referentes a um tipo de contexto; ou (iii) usar regras definidas pelo usuário que determinam quais contextos devem ser agregados.

O processo de Agregação também pode ser utilizado para verificar a consistência do contexto. Tendo em vista a complexidade dos contextos de interesse das aplicações é possível que os diferentes sensores envolvidos em determinada situação produzam dados com algum nível de conflito. Assim, quando dados contextuais são combinados podem ser geradas inconsistências, então este aspecto deve ser considerado e tratado nessa etapa. O tratamento dessas inconsistências é contemplado em uma área específica de estudo, a Qualidade de Contexto (QoC), a qual explora diferentes parâmetros, cuja quantificação indica a qualidade da informação empregada para caracterizar o contexto (NAZARIO et al., 2014).

Armazenamento

A camada de Armazenamento está relacionada com a necessidade de manter o histórico das informações de contexto. Um histórico de contexto pode ser utilizado para estabelecer tendências e prever situações futuras de interesse das aplicações.

O método de armazenamento é uma característica central relacionada a essa camada. O emprego de bancos de dados viabiliza a persistência e o uso de uma linguagem padrão para consulta e gerenciamento dos dados. Outras opções incluem a possibilidade de manter as informações de contexto em arquivos com estrutura

e gerenciamento específicos, bem como em espaços de tuplas que proporcionam persistência e sincronização.

Consulta e Inferência

As camadas de Consulta e Inferência podem abranger desde mecanismos simples para consulta de dados de contexto armazenados até sofisticados mecanismos de inferência. A presença de mecanismos de inferência tem influência direta na maneira como as informações de contexto são geradas, ou seja, permitem a produção de contextos implícitos e não apenas contextos explícitos.

A consulta produz contextos explícitos, ou seja, aqueles diretamente representados e instanciados no modelo, sendo obtidos a partir de filtragens das informações de contexto já existentes. Por sua vez, a inferência produz contextos implícitos, ou seja, aqueles que não estão representados ou instanciados através modelo e são obtidos por processos de dedução a partir das informações de contexto já existentes.

A linguagem empregada pelos mecanismos de consulta pode variar de uma linguagem específica até linguagens padrões para consultas a dados, como as baseadas em SQL (*Structured Query Language*). As características associadas aos motores de inferência dizem respeito à construção e armazenamento das regras de inferência, bem como a capacidade de inferir novos contextos.

A inferência pode ser baseada na estrutura, propriedades e relações das entidades de contexto, o que depende do modelo de contexto, ou em regras construídas com linguagens específicas que são executadas sobre o modelo de contexto.

2.3 Ciência de Situação

O avanço das tecnologias de sensoriamento possibilitou progressos significativos na concepção de sensores de tamanhos menores, mais leves, com menor custo e maior autonomia de suas baterias. Estes sensores coletam dados no ambiente, permitindo que o sistema computacional possa fornecer serviços personalizados, de acordo com o contexto de interesse das aplicações. Esses dados podem ser referentes a informações do usuário como localização, velocidade de deslocamento e sinais vitais; bem como do ambiente, por exemplo, temperatura, umidade e luminosidade (KRUMM, 2010).

Devido à complexidade dos dados de contexto sensoriados, na maioria das vezes, as aplicações devem ajustar seu comportamento considerando informações de mais alto nível, abstraídas dos dados individualmente sensoriados pelos sensores, por exemplo, temperatura, frequência cardíaca ou pressão arterial de um paciente.

Nesse sentido, a Figura 2.1 mostra um exemplo de sensoriamento de um paciente, no qual o sinal vital frequência cardíaca se considerado isoladamente não permitiria identificar a situação do paciente. Desse modo, no exemplo, uma abstração envolvendo dados de localização, atividade e frequência cardíaca, possibilitaria diferenciar se o paciente está tendo uma arritmia cardíaca ou fazendo exercícios físicos, caracterizando ou não uma situação de risco para saúde. Este exemplo é discutido no cenário de uso apresentado na Seção 5.1.3.

Esta abstração em alto nível de dados de contexto que define um estado específico de uma entidade é denominada de situação. Portanto, uma situação decorre da agregação de significado aos dados de contexto, correspondendo a uma abstração de eventos do mundo real, derivada da percepção de contextos e da compreensão de como estes contextos podem ser correlacionados. No exemplo mostrado na Figura 2.1, ao processar e agregar

significado através de mecanismos de raciocínio aos dados de contexto coletados sobre um determinado paciente, torna-se possível identificar uma situação de risco à saúde do mesmo.

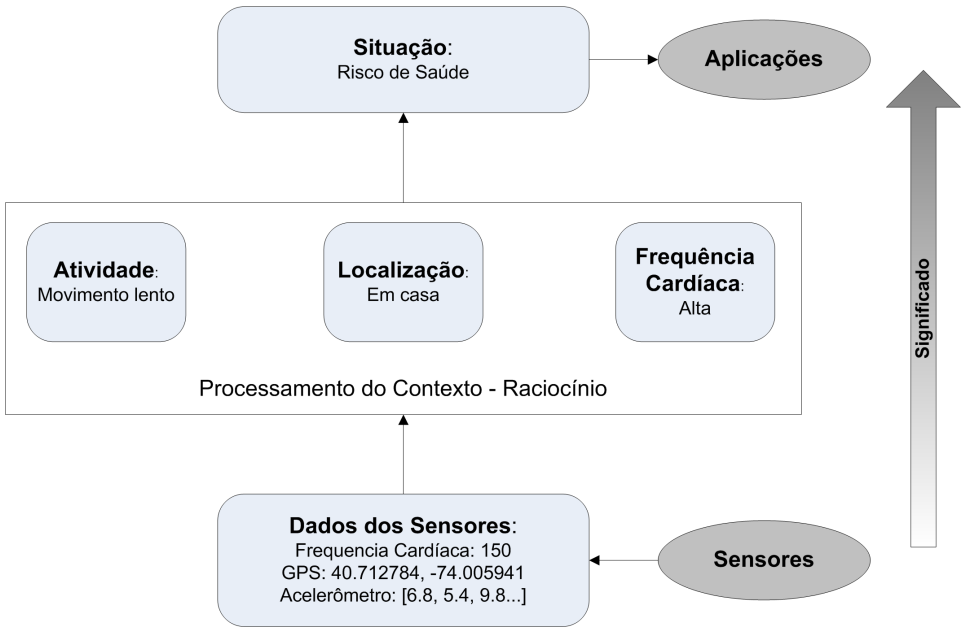


Figura 2.1: Exemplo de Identificação de Situações

Desse modo, a identificação de situações é um dos aspectos que pode prover transparência ao usuário na interação com o sistema computacional, resguardando seu envolvimento para aspectos nos quais sua intervenção é necessária. No entanto, a identificação de situações e o decorrente ajuste do comportamento das aplicações às diversas alterações que podem ocorrem no ambiente é uma atividade complexa, sendo preciso considerar a relação entre os diferentes contextos de interesse, bem como possíveis inconsistências entre os mesmos. Outro aspecto que deve ser considerado diz respeito às limitações dos mecanismos de inferência. Ainda, verifica-se que a necessidade do emprego de sensores de diferentes naturezas contribui para elevar a complexidade na interpretação das informações de contexto (PEREIRA; COSTA; ALMEIDA, 2013).

O processo de aquisição de dados sensoriados envolve diversos fatores que podem comprometer a interpretação do contexto pelos mecanismos de Ciência de Situação, tais como: a validade temporal (perda da relevância da informação coletada), as limitações técnicas da eletrônica envolvida, as eventuais desconexões da rede de sensores, a ocorrência de defeitos. Todos estes fatores geram dificuldades para transformar os dados sensoriados em informações compreensíveis e úteis para as aplicações (COSTA et al., 2012).

Uma situação pode corresponder ao estado de uma entidade ou ainda pode ser formada a partir de outras situações com granulosidade mais fina. Nesse sentido, destacam-se algumas relações que as situações podem possuir:

- Generalização: uma situação pode ser considerada como mais geral do que outra, se a sua ocorrência tem implicação em outra mais específica.

- Composição: uma situação pode ser decomposta em um conjunto situações menores.

- Dependência: uma situação depende de outra situação se a ocorrência da primeira é determinada pela ocorrência da segunda.
- Contradição: duas situações podem ser consideradas como mutuamente exclusivas se elas não podem ocorrer ao mesmo tempo, no mesmo lugar e com a mesma entidade.
- Sequência temporal: uma situação pode ocorrer antes ou depois de outra situação, ou ainda intercalar com outra situação.

2.3.1 Eventos e Regras na Ciência de Situação

Na perspectiva da Ciência de Situação, os eventos são responsáveis por indicar mudanças no estado do contexto de interesse, que estejam ocorrendo no momento (BOUZEGHOUB; DO; LECOCQ, 2007).

Em um ambiente ubíquo, diferentes tipos de eventos podem ser definidos e, dependendo do evento ocorrido em um dado instante de tempo, uma nova situação pode ser caracterizada. O que determina as condições para ocorrência de um evento são as ações executadas sobre o ambiente (FLEISCHMANN, 2012).

A literatura apresenta diversas definições para evento. Uma definição bastante referenciada caracteriza evento como uma ocorrência dentro de um determinado sistema ou domínio. Ainda, evento é considerado como uma ocorrência única dentro de um ambiente, envolvendo uma mudança no estado do contexto. Inclui normalmente a noção de tempo, a ocorrência, e os detalhes que pertencem explicitamente ao evento ou ambiente que podem ajudar a explicar ou compreender as causas ou efeitos do evento (ETZION; NIBLETT, 2010).

Assim, os eventos podem ser utilizados para indicar mudanças em estados de contexto que ocorrem em ambientes sensoriados e sejam de interesse de aplicações. Por exemplo, uma regra pode especificar que um valor de temperatura entre 20ºC e 30ºC é normal para um determinado ambiente que está sendo sensoriado. Por outro lado, se a temperatura coletada deste mesmo sensor estiver acima de 30°C representa uma mudança no estado do contexto (temperatura) que o deixa com valor inadequado para o ambiente. Caso as leituras seguintes deste sensor permaneçam com valores acima de 30ºC isso pode caracterizar um padrão de eventos (temperatura em elevação) que pode constituir uma situação (temperatura alta), válida por um determinado período de tempo, que seja de interesse de algum componente de software de uma aplicação.

Desta forma, observa-se que os estados de contexto capturados de sensores não necessariamente estão associados a eventos de interesse, ou seja, não representam algo relevante naquele determinado contexto. Por outro lado, outros dados coletados são relevantes e podem produzir uma mudança na situação atual.

O emprego de regras no processamento dos dados contextuais para identificação de situações pode se constituir em uma estratégia direcionada tanto para o raciocínio sobre o contexto, como para o tratamento de eventos que geram mudanças no estado do contexto. As regras possibilitam a geração de informações contextuais de alto nível a partir de dados contextuais de baixo nível, apresentando um grande potencial de utilização pelas aplicações cientes de situação no cenário da IoT, pois são uma maneira usualmente mais simples de representar o pensamento humano, viabilizando o processamento computacional do mesmo (PERERA et al., 2014).

Os sistemas baseados em regras podem adotar basicamente dois tipos de regras: regras ativas ou regras de produção (ETZION; NIBLETT, 2010).

As regras ativas (regras ECA - Evento-Condição-Ação) têm uma sintaxe com o seguinte formato:

```
ON <evento>, IF <condição>, DO <ação>
```

Isto significa que toda vez que a ocorrência do evento descrito na cláusula `ON` for detectada, e se a condição presente na cláusula IF (geralmente impondo restrições sobre os diferentes aspectos dos eventos considerados na cláusula `ON`) é verdadeira, a ação especificada na cláusula `DO` é executada pelo sistema. Quando a cláusula `ON` for satisfeita a regra é "acionada" (*triggered*) e se, também a cláusula `IF` for satisfeita, então a regra é "disparada" (*fired*).

As regras de produção, por sua vez, têm por princípio a reação às mudanças de estado das variáveis de contexto. Toda vez que uma condição avaliada é satisfeita uma ação é executada. Sintaticamente elas obedecem a sentença do tipo IF <condição> THEN <ação> e, ao contrário das regras ECA, não necessitam nenhum evento de ativação, sendo executadas sequencialmente.

2.3.2 Identificação de Situações

Nesta seção são apresentadas algumas técnicas para identificação de situações, organizadas em dois tipos: baseadas em especificação e baseadas em aprendizagem. A sistematização de conceitos apresentada nesta seção tem como referência os trabalhos de CHEN et al. (2009); LOKE (2010); e YE; DOBSON; MCKEEVER (2012).

Técnicas Baseadas em Especificação

As técnicas baseadas em especificação são usualmente aplicadas em sistemas que possuem um número reduzido de sensores, para os quais os dados e relações podem ser interpretados através de regras elaboradas por especialistas.

A identificação de situações com base em especificação consiste em representar o conhecimento através de regras lógicas e aplicar mecanismos de raciocínio para identificar situações a partir dos dados coletados pelos sensores. Nos próximos parágrafos estão descritas algumas destas técnicas.

Lógica Formal

A utilização de lógica formal para identificação de situações depende da capacidade de modularização das situações e discretização de variáveis de contexto. Assim, a lógica formal assume que o conhecimento a respeito de situações pode ser modularizado ou discretizado. O principal objetivo é proporcionar a fundamentação teórica para construção de sistemas que sejam capazes de: (i) formalmente representar as especificações lógicas das situações; (ii) verificar a integridade e a consistência das especificações das situações em uma base de regras; e (iii) estender os sistemas existentes para incorporar dados de mais sensores e reconhecer mais situações.

A lógica formal é constituída por um conjunto de operações matemáticas definidas principalmente em teoria de conjuntos e álgebra booleana. Como ferramenta de programação para lógica formal, pode ser utilizada a linguagem Prolog.

No código apresentado na Figura 2.2 pode ser visto um exemplo de modelagem de situação, aplicando lógica formal modelada em Prolog. Neste exemplo é feita a configuração do tipo de toque de um telefone celular baseada na situação do usuário. Para as situações "reunião" (*meeting*) e "palestra" (*lecture*), o toque é configurado para

silencioso. Quando o usuário estiver em um restaurante (*restaurant*), os sons de alerta do celular são habilitados.

```
required_phone_mode(quiet) :- my_current_user(E), in_meeting_now*>E.
required_phone_mode(quiet) :- my_current_user(E), in_lecture*>E.
required_phone_mode(noisy) :- my_current_user(E), restaurant*>E.
```

Figura 2.2: Identificação de Situações com Lógica Formal

Ontologia

As ontologias permitem representar os dados de contexto de forma que estes podem ser entendidos e compartilhados por humanos e computadores. As ontologias podem ser utilizadas para detecção de inconsistências, derivação de novos conhecimentos e também no reconhecimento de atividades.

Modelos baseados em ontologias tiram proveito da capacidade destas em expressar relações complexas. A validade de dados é geralmente expressa através da imposição de restrições, centrando-se sobre as relações entre as entidades. As ontologias são adequadas para mapear o conhecimento dentro de uma estrutura de dados facilmente utilizada e interpretada. Além disso, a adoção de ontologias permite a reutilização e a criação de vocabulários de domínio comuns e compartilhados.

Na Figura 2.3 pode ser visto um exemplo de inferência para situação de uma entidade baseado na modelagem por ontologia. A situação da instância da classe "Humano", definida como "João", possui a variável de contexto "temperatura" com valor igual a 39,0ºC. Como o valor da temperatura é maior do que o definido na relação de dependência, a propriedade inferida "Febre" recebe o estado "Sim". Logo, pode-se afirmar: "João está com febre".

Lógica Temporal e Espacial

A perspectiva de tempo e espaço relacionada a contextos e situações pode ser representada e processada através da lógica temporal e espacial. Essa abordagem permite a detecção de situações relacionadas a pessoas, ambientes e/ou objetos, considerando a relação entre os eventos associados e a sua duração.

A lógica temporal e espacial pode ser aplicada na representação e raciocínio sobre as características e restrições de contexto e situações. A proposta da Lógica Temporal e Espacial indica que a representação de tempo e espaço por intervalos pode ser mais interessante do que a representação por pontos específicos.

Lógica Fuzzy

A lógica fuzzy é utilizada para lidar com incertezas através do uso de funções de pertinência. Na Computação Ubíqua pode ser empregada para mapear os dados de sensores para variáveis linguísticas que tenham significado humano e sejam capazes de serem avaliadas. É utilizado um limiar, definido por especialistas ou pelas experiências dos usuários, que estabelece a qual faixa de valores os dados dos sensores pertencem. Ela permite que conhecimento impreciso seja tratado, podendo ser aplicada no raciocínio de situações com incerteza.

As aplicações ubíquas, especialmente no cenário da IoT, lidam com uma grande quantidade de sensores, os quais possuem também elevada heterogeneidade. Nesse

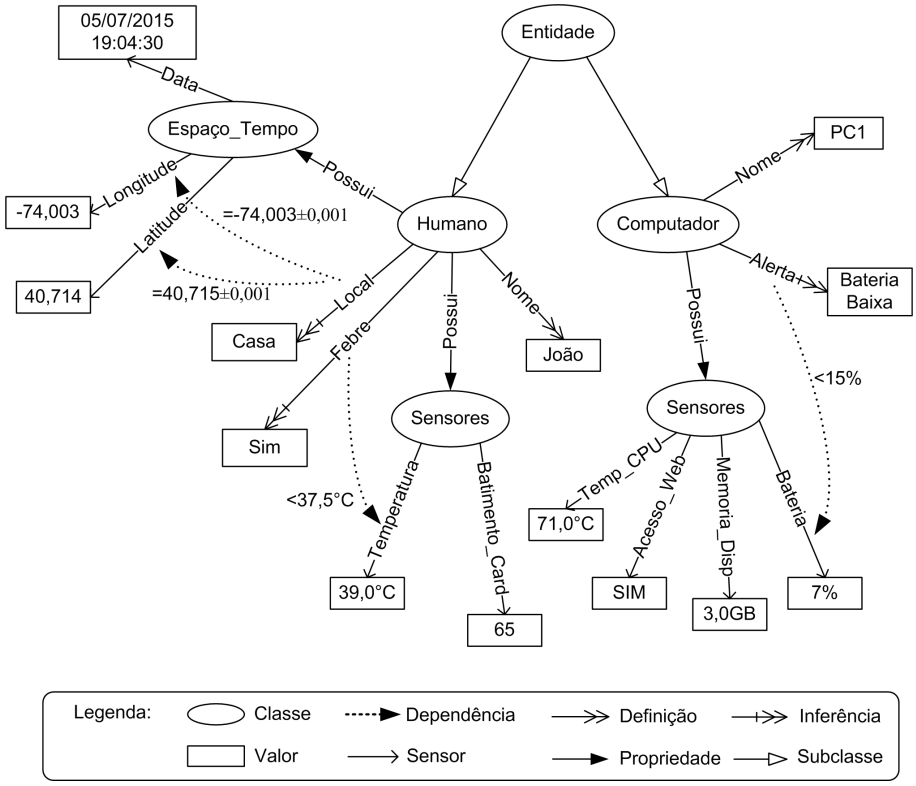


Figura 2.3: Identificação de Situações com Ontologias

cenário, a incerteza de disponibilidade e diferentes faixas de precisão de sensores agregam incertezas ao sistema. Além disso, na modelagem de contexto e de situação, é necessário gerar inferências de estados a partir de dados de sensores. Em muitos casos, essa classificação não está relacionada com um valor especifico, como falso ou verdadeiro, mas sim com uma faixa de valores.

Assim, nesse tipo de cenário, a lógica fuzzy se mostra uma técnica adequada para tratar a incerteza de dados e mapear saídas consistentes para ao sistema.

A lógica fuzzy, ao contrário da lógica booleana, permite respostas intermediárias entre falso e verdadeiro, ou seja, assume a existência de infinitos valores ente 0 e 1. Assim, conceitos intrínsecos da linguagem humana para representar situações de forma aproximada podem ser modelados. Por exemplo, as afirmações: “está pouco frio” ou “está muito quente”, de acordo com o contexto, podem transmitir de maneira satisfatória a informação entre os interlocutores, sem a necessidade de tratar valores numéricos exatos ou escalas de medidas.

Na Computação Ubíqua, a lógica fuzzy pode ser utilizada como ferramenta na identificação de situações através do mapeamento do grau de pertinência de uma variável para a ocorrência de uma situação. Isso é feito através de variáveis linguísticas, as quais podem assumir valores de um determinado conjunto de termos. A definição dos conjuntos para as variáveis linguísticas pode ser feita com o auxílio de uma ferramenta para ontologia.

Na Figura 2.4 pode ser visto um exemplo do mapeamento de pertinência da variável temperatura para a variável linguística “febre”, a qual pode assumir os valores: “Baixa”, “Média” e “Alta”. Como trata-se de uma técnica baseada em especificação, as faixas de valores são estabelecidas a partir do conhecimento de especialistas. Porém, inferências como “febre um pouco alta” para 38,5^{o}C e “febre muita alta” para 40^{o}C, podem ser

facilmente geradas.

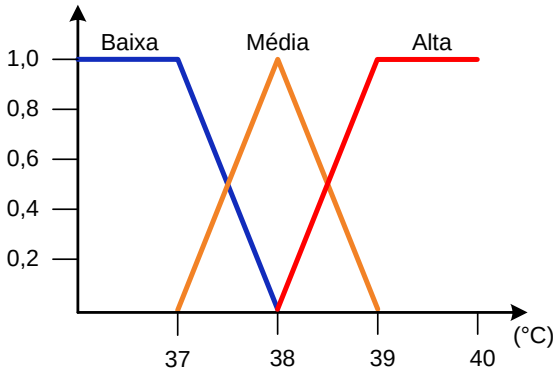


Figura 2.4: Identificação de Situações com Lógica Fuzzy

Quando uma dada situação depender de mais de uma variável, a pertinência final pode ser dada pelo somatório da pertinência de cada variável multiplicada por um peso correspondente à sua contribuição global na situação, conforme expresso na equação 2.1.

$$P = \sum_{i=1}^{N} w_i.\mu(x_i) \tag{2.1}$$

Onde: x_i representa uma dada variável contínua ou discreta, $\mu(x_i)$ representa a função de pertinência individual da variável, e w_i representa o peso da variável na situação final.

Técnicas Baseadas em Aprendizagem

Com as técnicas baseadas em aprendizagem de máquina é possível que o sistema aprenda associações complexas entre situações e dados de muitos sensores. Apesar dessas técnicas apresentarem bons resultados na identificação de situações, elas precisam de uma grande quantidade de dados para treinamento com o intuito de estimar os parâmetros do modelo a ser utilizado.

As técnicas baseadas em aprendizagem permitem que o sistema gere inferências de forma automática, estabelecendo relações entre dados de sensores e estados anteriores do sistema através de um processo de aprendizagem de máquina. Essas técnicas são aplicadas em sistemas em que existem muitos dados de entradas e sensores de forma que a inferência de situação baseada em especificação torna-se inviável. Algumas destas técnicas, aplicadas na área de processamento contextual e identificação de situações são descritas a seguir.

Redes Bayesianas

As redes bayesianas fornecem um método para entender como a probabilidade da ocorrência de um evento está condicionada a um outro evento. Esta abordagem é mais adequada para aplicações onde não há necessidade de representar a falta de conhecimento, e as dependências são fáceis de serem obtidas através da representação probabilística, bem como as probabilidades anteriores estão disponíveis. O desempenho pode ser baixo quando há pouca disponibilidade de dados de probabilidade.

Uma rede bayesiana é um gráfico acíclico dirigido, em que cada nó representa uma variável que pode ser discreta ou contínua, e cada arco é a relação causal entre os nós. Se

houver um arco a partir de um nó A para outro nó B, então A é chamado um pai de B, o que implica que a variável B é considerado como diretamente dependente de A.

Na Figura 2.5 tem-se um exemplo de rede de classificação bayesiana, a partir da qual pode ser calculada a probabilidade da variável "C" em assumir um dado estado "j" em função das probabilidades condicionais P($X_i|C$), conforme equação 2.2.

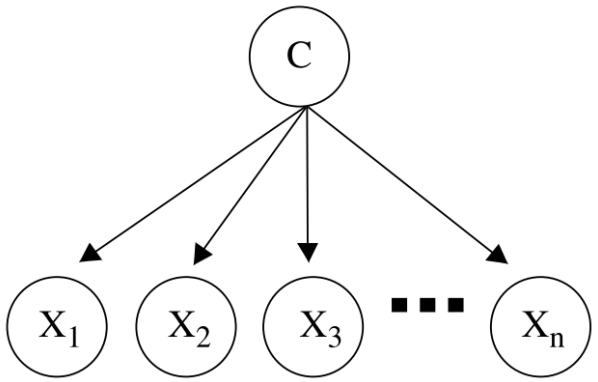


Figura 2.5: Estrutura de uma Rede de Classificação Bayesiana

$$P(C_j|X_1...X_n) = \prod_{i=1}^{n} P(X_i|C) \tag{2.2}$$

O processo de aprendizagem de uma rede Bayesiana consiste em determinar o mapa de probabilidades condicionais $P(X_i|C)$ para cada variável da rede, possibilitando assim que, a partir dos valores de um dado conjunto de variáveis, a probabilidade condicional de outra possa ser determinada seguindo a equação 2.2.

Árvores de Decisão

Uma árvore de decisão é um modelo preditivo onde cada folha representa uma classificação e cada ramo representa uma conjunção de características que levam à classificação alvo.

Uma árvore de decisão é construída sobre a entropia da informação, sendo escolhida uma variável em cada etapa. Uma das principais vantagens de usar uma árvore de decisão é que ela pode gerar regras de classificação que são fáceis de entender e explicar.

Árvores de decisão tradicionais são eficientes no raciocínio com conjuntos de dados relativamente pequenos. No entanto, a eficiência de sua construção depende do tamanho dos dados de treino, uma vez que a construção destas árvores requer que os dados de treinamento residam na memória. Portanto, quando elas são usadas em aplicações para mineração de um conjunto muito grande de dados, elas se tornam ineficientes devido à troca de dados de treinamento entre a memória principal e a *cache*.

A literatura aponta vários métodos que podem ser empregados na modelagem e treinamento de árvores de decisão, um método amplamente utilizado é o C4.5. Por exemplo, aplicando este método sob um arquivo de treinamento como mostrado na Figura 2.6 é obtida a árvore de decisão mostrada na Figura 2.7. Neste exemplo, está modelada a probabilidade para ocorrência de um jogo de tênis baseado em um histórico de eventos relacionados com condições climáticas.

```
@relation jogo.simbolico

@attribute tempo {ensolarado, nublado, chuvoso}
@attribute temperatura {quente, média, fria}
@attribute umidade {alta, normal}
@attribute vento {sim, não}
@attribute jogar {sim, não}

@data
ensolarado,quente,alta,não,não
ensolarado,quente,alta,sim,não
nublado,quente,alta,não,sim
chuvoso,média,alta,não,sim
chuvoso,fria,normal,não,sim
chuvoso,fria,normal,sim,não
nublado,fria,normal,sim,sim
ensolarado,média,alta,não,não
ensolarado,fria,normal,não,sim
chuvoso,média,normal,não,sim
ensolarado,média,normal,sim,sim
nublado,média,alta,sim,sim
nublado,quente,normal,não,sim
chuvoso,média,alta,sim,não
```

Figura 2.6: Dados para Treinamento de uma Árvore de Decisão

Redes Neurais

As redes neurais artificiais podem ser uma boa alternativa para o reconhecimento de situações, proporcionado o raciocínio e a predição de dados, uma vez que possuem a característica de se moldarem ao problema que estão sendo aplicadas.

Uma rede neural é composta de neurônios artificiais, os quais estão ligados entre si de acordo com uma arquitetura de rede específica. São essas arquiteturas que caracterizam os diferentes tipos de redes, sendo que algumas são mais específicas para um determinado tipo de aplicação que outras.

As redes neurais podem aprender automaticamente e mapear complexas relações não-lineares. A Figura 2.8 mostra um exemplo de classificador baseado em uma rede neural artificial.

O desempenho de redes neurais é afetado pela quantidade de dados de treinamento. Uma rede neural é considerada uma boa opção se há abundância de dados para treinamento e se o domínio do problema não pode ser adequadamente compreendido para derivar um modelo aproximado.

2.4 Internet das Coisas

A Internet das Coisas vem ganhando destaque como um paradigma de evolução da Internet (PERERA et al., 2014). A IoT preconiza a ideia do tudo conectado, ou seja, qualquer "coisa", como utensílios, sistemas de transportes, redes de energia, equipamentos pessoais, sistemas agrícolas, o corpo humano, pode possuir sensores capazes de gerar e compartilhar informações tendo a Internet como principal meio de interoperação (LI; XU; ZHAO, 2015).

O compartilhamento destas informações disponibilizadas pelo cenário da IoT ocorre automaticamente com outros sistemas computacionais externos, o que introduz o conceito

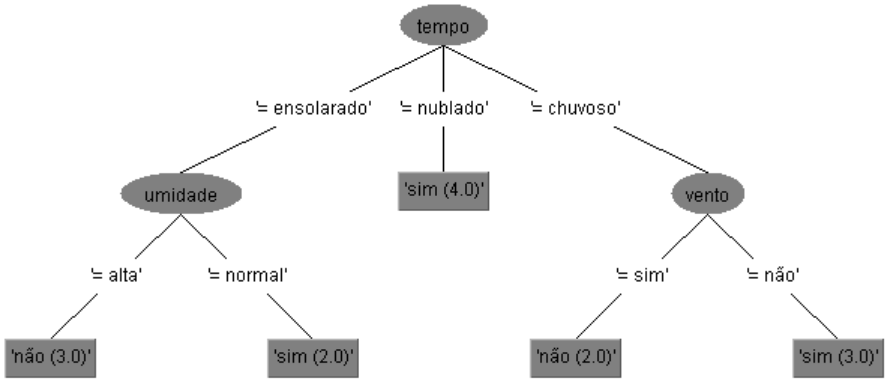


Figura 2.7: Identificação de Situações com Árvore de Decisão

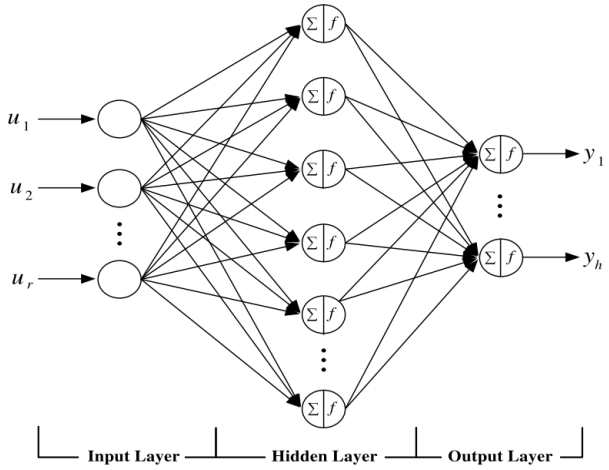


Figura 2.8: Exemplo de Classificador Baseado em uma Rede Neural

de objetos inteligentes. Estes objetos inteligentes ou "coisas" devem ser capazes de detectar fenômenos (físicos ou lógicos) e/ou disparar ações sobre o ambiente, bem como possuir alguma capacidade computacional e de comunicação, além de um identificador único que os capacitem a interoperarem de forma o mais autônoma possível (MIORANDI et al., 2012).

A IoT tem despertado interesse pelo crescente número de dispositivos que vem sendo interconectados à Internet, gerando novas informações que podem agregar conhecimento para os mais diferentes tipos de aplicações. Por ser uma área relativamente nova, ainda possui problemas de pesquisa em aberto, o que estimula o envolvimento da comunidade científica no tema (WHITMORE; AGARWAL; DA XU, 2015).

A IoT também tem sido considerada como uma abordagem para promover a ubiquidade das soluções computacionais, suprindo as demandas das mesmas em relação aos aspectos infraestruturais. Com essa característica, a IoT vem se constituindo em um dos elementos centrais para consolidar a integração entre os sistemas computacionais e o ambiente físico, com potencial para produzir informações contextuais em larga escala (PIRES et al., 2014).

2.4.1 Visões das Frentes de Estudo em IoT

Segundo ATZORI; IERA; MORABITO (2010) e (GUBBI et al., 2013b), a Internet das Coisas apresenta três visões de pesquisa que podem levar a focos distintos:

- Visão orientada à Internet: esta visão de pesquisa abrange o ponto de vista de redes. As pesquisas direcionadas a esta visão procuram criar modelos e técnicas para a interoperabilidade dos dispositivos em rede.

- Visão orientada às Coisas: trata da integração dos objetos presentes em um cenário de IoT. As pesquisas nesta visão procuram apresentar propostas que garantam o melhor aproveitamento dos recursos dos dispositivos e sua comunicação.

- Visão orientada à Semântica: esta visão parte de uma análise semântica da expressão "Internet das Coisas", tendo foco na comunicação e troca de informações entre dispositivos distintos. Os trabalhos nesta perspectiva apresentam propostas relacionadas a representação, armazenamento, interconexão, pesquisa e organização da informação gerada na IoT, buscando soluções para a modelagem das descrições que permitam um tratamento adequado para os dados gerados pelos objetos.

2.4.2 Definição de IoT

Na literatura não existe uma definição única para Internet das Coisas. Segundo PERERA et al. (2014), isto ocorre principalmente por se tratar de área de pesquisa recente e bastante ampla. A seguir estão resumidas duas definições de IoT encontradas na literatura da área.

- A IoT é um paradigma onde objetos possuem identificação, funcionalidades de sensoriamento e conectividade, bem como capacidade de processamento que permitem a comunicação entre eles e com outros dispositivos e serviços através da Internet (WHITMORE; AGARWAL; DA XU, 2015).

- A Internet das Coisas permite que objetos possam se conectar a qualquer momento, em qualquer lugar, com qualquer coisa, de preferência usando qualquer caminho ou rede e qualquer serviço (FRIESS, 2011).

Dentre as definições apresentadas, a segunda é que se mostra mais alinhada com a premissa da UbiComp de prover computação de forma o mais transparente possível, através de diferentes dispositivos e redes, em qualquer lugar e a qualquer tempo, onde não apenas computadores, mas qualquer coisa pode se comunicar e transferir informações através da Internet.

Nesse cenário, o conceito de "coisa" é fundamental na perspectiva da UbiComp. Assim, a definição dada por MIORANDI et al. (2012) se mostra adequada, pois caracteriza "coisas" como objetos que devem ter um identificador único, devem ser capazes de detectar aspectos físicos ou lógicos e/ou disparar ações sobre o ambiente, bem como devem possuir alguma capacidade computacional e de comunicação.

2.4.3 Visão Geral de uma Arquitetura de IoT para Sistemas Cientes de Situação

Uma arquitetura de IoT é composta por rede de sensores e atuadores, bem como por camadas de software que podem ser instalados em dispositivos computacionais fixos, móveis ou em nuvem, e realizam a intermediação entre os dados de coleta e/ou atuação e as aplicações e/ou serviços. Uma rede de sensores e atuadores compreende o hardware, firmware e uma camada de software. Em um sistema ciente de situação, os sensores são responsáveis por coletar as informações de contexto e os atuadores por executar ações decorrentes do processamento dos dados contextuais coletados (PERERA et al., 2014).

As redes de sensores em um cenário de IoT usualmente tem um propósito geral, sendo os sensores implantados com o objetivo de atender diferentes domínios de aplicação. Por exemplo, em uma ponte recém construída podem ser instalados sensores de pressão para monitorar sua estrutura, no entanto estes sensores podem se conectar a muitos outros sensores e serem reutilizados para controlar o tráfego em um estágio posterior. Portanto, soluções de *middleware*, *frameworks* e APIs são projetados para fornecer serviços genéricos e funcionalidades, como inteligência, interoperabilidade, semântica, Ciência de Situação, que são necessários para executar a comunicação entre sensores e atuadores (LI; XU; ZHAO, 2015).

Devido as diferentes naturezas das aplicações no cenário da IoT, o mesmo pode necessitar ser tratado por diferentes abordagens em uma arquitetura (PERERA et al., 2014), por exemplo, abordagem orientada por eventos e abordagem orientada por tempo. Nesse caso, alguns sensores produziriam dados quando da ocorrência de um evento (por exemplo, sensor de abertura de porta) e os outros produziriam dados continuamente, com base em períodos de tempo especificados (por exemplo, sensor de temperatura).

2.5 *Middleware* EXEHDA

Considerando a proposta de instanciação da Arquitetura SAUI no âmbito do *middleware* EXEHDA, esta seção registra a revisão feita sobre os aspectos arquiteturais e funcionais do EXEHDA.

O *middleware* EXEHDA (YAMIN, 2004) (LOPES et al., 2014) tem como objetivo definir a arquitetura para um ambiente de execução destinado às aplicações da Computação Ubíqua, no qual as condições de contexto são proativamente monitoradas e o suporte à execução deve permitir que tanto a aplicação como o *middleware* utilizem estas informações na gerência da adaptação de seus aspectos funcionais e não-funcionais. Nesse sentido, entende-se por adaptação não-funcional a capacidade do sistema em atuar sobre a localização física dos componentes das aplicações, e por adaptação funcional a capacidade do sistema em atuar sobre a seleção da implementação do componente a ser utilizado em um determinado contexto de execução.

As aplicações do ambiente gerenciado pelo EXEHDA são cientes do contexto, estando disponíveis a partir de qualquer lugar, todo o tempo. Os serviços fornecidos estão organizados em subsistemas relacionados a acesso ubíquo, comunicação, execução distribuída, reconhecimento do contexto e adaptação (LOPES et al., 2014).

2.5.1 Arquitetura de Software

A arquitetura de software do *middleware* EXEHDA é apresentada na Figura 2.9. 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 das 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.

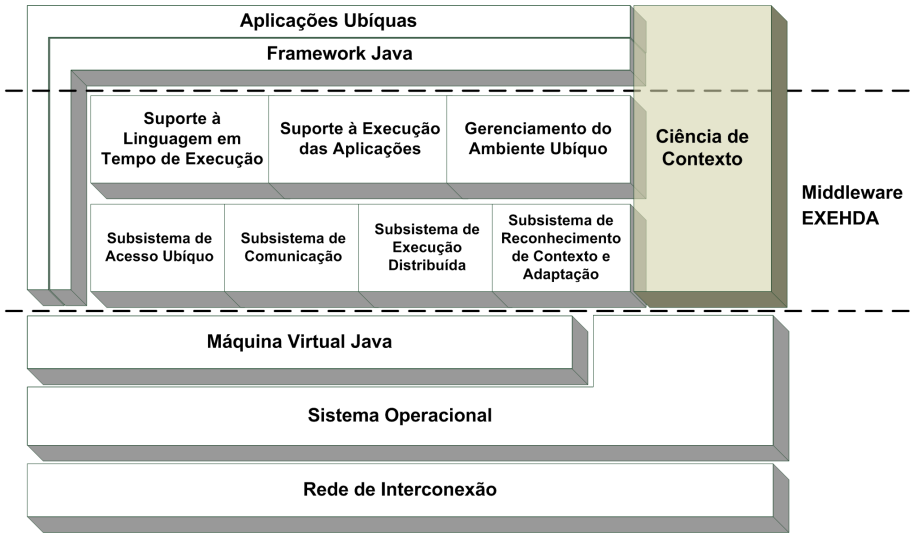


Figura 2.9: Arquitetura de Software do EXEHDA
(LOPES et al., 2014)

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. Este ambiente é organizado em células de execução e a sua composição envolve tanto os dispositivos dos usuários, como os equipamentos da infraestrutura de suporte, todos instanciados pelo seu respectivo perfil de execução do *middleware*.

O meio físico sobre o qual o ambiente ubíquo é definido constitui-se 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:

- EXEHDAcels: indica a área de atuação de uma EXEHDAbase, sendo 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.

2.5.2 Subsistemas do EXEHDA

O *middleware* EXEHDA é formado por quatro subsistemas, conforme mostra a Figura 2.10: Subsistema de Execução Distribuída, Subsistema de Comunicação, Subsistema de Acesso Ubíquo e Subsistema de Reconhecimento de Contexto e Adaptação.

Os subsistemas do EXEHDA são descritos nas próximas seções, sendo destacado o Subsistema de Reconhecimento de Contexto e Adaptação, para o qual a Arquitetura SAUI contribui diretamente, ampliando suas funcionalidades pela inclusão do suporte à Ciência de Situação (vide Capítulo 4).

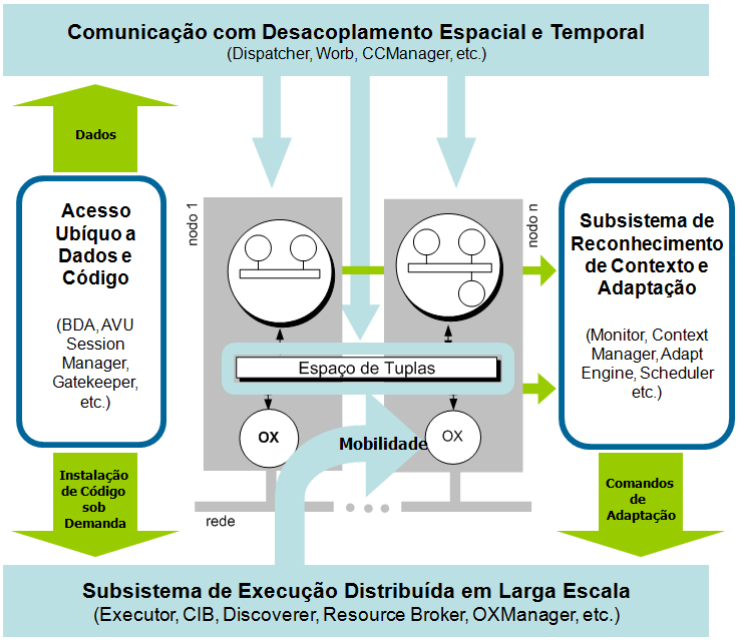


Figura 2.10: Subsistemas do EXEHDA
(LOPES et al., 2014)

Subsistema de Execução Distribuída

O Subsistema de Execução Distribuída é responsável pelo suporte ao processamento distribuído no EXEHDA. No intuito de promover uma execução efetivamente ubíqua, este subsistema interage com outros subsistemas do EXEHDA. Em específico, interage com o Subsistema de Reconhecimento de Contexto e Adaptação, de forma a prover comportamento distribuído e adaptativo às aplicações da Computação Ubíqua. Este subsistema é constituído pelos seguintes componentes:

Executor: serviço que acumula as funções de disparo de aplicações, e de criação e migração de seus objetos. Na implementação destas funções é empregada a instalação de código sob demanda. Para tal, interage com os serviços *CIB* e *BDA*, que podem ser vistos no itens seguintes.

Cell Information Base - CIB: serviço que implementa a base de informações da célula. Sua principal funcionalidade está relacionada à manutenção da infraestrutura

distribuída que forma o ambiente ubíquo.

OXManager: a abstração OX (Objeto eXehda), provida pelo *middleware* às aplicações, consiste em uma instância de objeto, criada por intermédio do serviço *Executor*, a qual pode ser associada meta-informação em tempo de execução. No caso da abstração OX, esta meta-informação ocorre na forma de atributos, i.e., pares <nome, valor>, sendo que "nome" é uma cadeia de caracteres ASCII que descreve o atributo, podendo "valor" ser um conteúdo genérico, inclusive de natureza binária. A gerência e manutenção da meta-informação associada a um OX é atribuição do serviço *OXManager*. Este serviço confere às operações de consulta e atualização dos atributos do OX o necessário caráter ubíquo, permitindo que estes sejam acessados a partir de qualquer nodo do ambiente ubíquo.

Discoverer: o serviço de descoberta de recursos é responsável pela localização de recursos especializados no ambiente ubíquo a partir de especificações abstratas dos mesmos. As especificações caracterizam o recurso a ser descoberto por meio de atributos e seus respectivos valores.

ResourceBroker: o controle da alocação de recursos às aplicações no EXEHDA é desempenhado pelo serviço *ResourceBroker*, o qual atende tanto requisições originárias da própria EXEHDAcel quanto oriundas de outras células do ambiente ubíquo.

Gateway: serviço que faz a intermediação das comunicações entre os nodos externos à célula e os recursos internos a ela. Da sua ação integrada com o *ResourceBroker* decorre o controle de acesso aos recursos de uma EXEHDAcel.

StdStreams: serviço que provê o suporte ao redirecionamento dos *streams* padrões de entrada, saída e erro. Sua funcionalidade ocorre por aplicação, sem a necessidade de modificação no código da mesma. Para isto, define que cada aplicação em execução em um determinado EXEHDAnodo possui um componente individualizado, o qual agrupa os três *streams* padrões.

Logger: o serviço Logger provê a funcionalidade de registro de rastro de execução (logging). Esta funcionalidade pode ser empregada para registro de operações importantes e/ou críticas realizadas, facilitando a identificação de situações de intrusão, ou de uso indevido do sistema.

Dynamic Configurator - DC: serviço que tem como objetivo realizar a configuração do perfil de execução do *middleware* em um determinado EXEHDAnodo de forma automatizada.

Subsistema de Comunicação

A natureza da mobilidade do hardware e, na maioria das vezes, também a do software, não garante a interação contínua entre os componentes da aplicação distribuída. As desconexões são comuns, não somente devido à existência de alguns links sem fio, mas sobretudo como uma estratégia para economia de energia nos dispositivos móveis. O subsistema de comunicação do EXEHDA disponibiliza mecanismos que atendem estes aspectos da Computação Ubíqua. Integram este subsistema os seguintes serviços:

Dispatcher: serviço que disponibiliza o modelo de comunicação mais elementar do EXEHDA, ou seja, troca de mensagens ponto-a-ponto com garantia de entrega e ordenamento das mensagens, o qual é especializado para operação no ambiente ubíquo. As mensagens trafegam entre instâncias do *Dispatcher* localizadas em nodos diferentes através de estruturas denominadas canais. Nesse sentido, quando de sua inicialização, o *Dispatcher* atualiza a informação do EXEHDAnodo na CIB (Cell Information Base), em específico o atributo *contactAddress*, provendo uma lista de protocolos e endereços que

podem ser utilizados para alcançar aquele EXEHDAnodo.

WORB: serviço que tem o objetivo de simplificar a construção de serviços distribuídos, permitindo que os programadores focalizem esforços no refinamento da semântica distribuída associada ao serviço em desenvolvimento, abstraindo aspectos de baixo nível relativos ao tratamento das comunicações em rede. Para tanto, oferece um modelo de comunicação baseado em invocações remotas de método, similar ao RMI, porém sem exigir a manutenção da conexão durante toda a execução da chamada remota.

CCManager: considerando a premissa da mobilidade lógica dos componentes que constituem as aplicações ubíquas, o suporte a um mecanismo de comunicação com característica de desacoplamento temporal e espacial é particularmente oportuno, à medida que este simplifica a construção de tais aplicações. Este serviço vem atender a esta demanda, disponibilizando um mecanismo baseado na abstração espaço de tuplas, o qual prescinde da coexistência temporal de emissor e receptor. Outro aspecto oportuno desta abstração é a facilidade com que podem ser implementados outros padrões de comunicação, além do ponto-a-ponto.

Subsistema de Acesso Ubíquo

A premissa de acesso em qualquer lugar, todo o tempo, a dados e código da Computação Ubíqua, requer um suporte do *middleware*. Os serviços que compõem este subsistema no EXEHDA são:

BDA - Base de Dados das Aplicações: o serviço BDA contempla métodos para a recuperação do código integral de uma aplicação ou de componentes específicos e suas dependências. Ainda, este serviço inclui métodos para a gerência das aplicações instaladas. O padrão de utilização concebido para o serviço BDA define que as requisições geradas nos EXEHDAnodos de uma determinada célula são sempre direcionadas à instância celular do serviço BDA da mesma EXEHDAcel onde foi gerada a requisição. Esta, se necessário, realiza o acesso a instâncias remotas (BDAs de outras células) para satisfazer uma dada requisição.

AVU - Ambiente Virtual do Usuário: a premissa *siga-me* reflete-se não só nas aplicações que um usuário atualmente executa, mas no seu ambiente computacional como um todo. Este engloba, além das aplicações em execução, as informações de personalização das aplicações definidas pelo usuário, o conjunto de aplicações instaladas, como também seus arquivos privados. É atribuição do serviço AVU a manutenção do acesso ubíquo a este ambiente virtual, da forma mais eficiente possível.

SessionManager: serviço responsável pela gerência da sessão de trabalho do usuário, sendo definida pelo conjunto de aplicações correntemente em execução para aquele usuário. A informação que descreve o estado da sessão de trabalho é armazenada no AVU, estando portanto disponível de forma ubíqua.

Gatekeeper: serviço responsável por intermediar acessos entre as entidades externas ao EXEHDA e os serviços do *middleware*, conduzindo os procedimentos de autenticação necessários.

Subsistema de Reconhecimento de Contexto e Adaptação

Este subsistema inclui serviços que tratam desde a extração da informação bruta sobre as características dinâmicas e estáticas dos recursos que compõem o ambiente ubíquo, passando pela identificação em alto nível dos elementos de contexto, até o disparo das ações de adaptação em reação a modificações no estado de tais elementos de contexto. Integram este subsistema os seguintes serviços:

Collector: responsável pela extração da informação bruta (diretamente dos recursos envolvidos) que, posteriormente refinada, dará origem aos elementos de contexto. Para isto, o serviço *Collector* aglutina a informação oriunda de vários componentes monitores e as repassa aos consumidores registrados. Um componente de monitoramento gerencia um conjunto de sensores parametrizáveis.

Deflector: disponibiliza abstração de canais de multicast para uso na disseminação das informações monitoradas. A presença do serviço *Deflector* é decorrência da busca de escalabilidade para a arquitetura de monitoração do EXEHDA.

ContextManager: responsável pelo tratamento da informação bruta produzida pela monitoração para produção de informações abstratas referentes aos elementos de contexto (valores contextualizados). É recebida como parâmetro uma descrição (XML) de como o dado referente àquele elemento de contexto deve ser produzido a partir da informação proveniente da monitoração.

AdaptEngine: responsável pelo controle das adaptações de cunho funcional, este serviço provê facilidades para definição e gerência de comportamentos adaptativos por parte das aplicações. Deste modo, libera o programador de gerenciar os aspectos de mais baixo nível envolvidos na definição e liberação dos elementos de contexto junto ao *ContextManager*. Outra função do serviço *AdaptEngine* é prover um mecanismo de carga de código contextualizado. Desta forma, com base na informação do estado do elemento de contexto fornecido pelo serviço *ContextManager*, seleciona e carrega o código correspondente dentre alternativas de código definidas na política de adaptação funcional vigente. Este serviço disponibiliza ainda métodos para a ativação de ações quando determinado estado de um elemento de contexto tornar-se ativo.

Scheduler: serviço central na gerência das adaptações de cunho não-funcional no EXEHDA, isto é, que não implicam alteração de código. Nesse sentido, o *Scheduler* emprega a informação de monitoração, obtida junto ao serviço *Collector*, para orientar operações de mapeamento. Essas operações decorrem de instanciações remotas ou migrações realizadas pelo serviço *Executor*, ou quando de chamadas de re-escalonamento, originadas do estado atual de um recurso não satisfazer mais as necessidades de um objeto anteriormente a ele alocado.

2.6 Considerações do Capítulo

Este capítulo apresentou os esforços realizados durante a revisão de conceitos associados a esta tese, destacando aspectos dos principais temas que constituem o foco de pesquisa, especificamente, a Computação Ubíqua, a Ciência de Contexto e de Situação e a Internet das Coisas. Ainda, foram descritos aspectos arquiteturais e funcionais do *middleware* EXEHDA, os quais são considerados na concepção da Arquitetura SAUI.

A revisão de conceitos indicou que uma das premissas centrais da UbiComp diz respeito ao acesso do usuário ao seu ambiente computacional quando e onde quiser e com qualquer dispositivo. Para atingir esse grau de ubiquidade e transparência das aplicações ubíquas ficou caracterizado que estas devem ser cientes de situação. Nesse sentido, a revisão apontou as situações como uma visão de alto nível e abrangente dos contextos de interesse das aplicações, que permite as mesmas, em função de modificações nos seus contextos de interesse, ajustarem seu comportamento.

Também, ficou constatado que para prover Ciência de Situação às aplicações é necessário considerar aspectos relacionados à infraestrutura para suporte à coleta e processamento dos dados contextuais, bem como para identificação das situações

relacionadas aos contextos de interesse das aplicações. Nessa perspectiva, a Internet das Coisas tem sido considerada como uma abordagem capaz de suprir a carência da UbiComp em relação aos aspectos infraestruturais, promovendo suporte ao provimento de Ciência de Situação da forma mais transparente possível.

Considerando este cenário, a revisão de conceitos apresentada neste capítulo apontou que a representação e o processamento do contexto para identificação de situações vêm se constituindo em uma área de pesquisa desafiadora no cenário da Ciência de Situação. Nessa perspectiva, o emprego de abordagens híbridas tem se destacado como uma estratégia promissora, a qual busca integrar diferentes técnicas de representação dos dados contextuais e de raciocínio sobre o contexto a fim de identificar situações.

Tendo em vista que o esforço de concepção da Arquitetura SAUI considera os aspectos arquiteturais do *middleware* EXEHDA, bem como os seus princípios operacionais, este capítulo apresentou um resumo das principais características e funcionalidades do EXEHDA. Foram destacados os subsistemas que compõem o EXEHDA, visto que, a arquitetura proposta por esta tese contribui com o Subsistema de Reconhecimento de Contexto e Adaptação do *middleware* EXEHDA, ampliando suas funcionalidades de forma a atender demandas inerentes à Ciência de Situação.

O próximo capítulo apresenta a revisão de literatura realizada, na qual estão contemplados os projetos mais próximos à proposta da Arquitetura SAUI, dentre os diversos trabalhos identificados como relacionados, tendo como objetivo caracterizar os principais aspectos que diferenciam a Arquitetura SAUI dos projetos estudados.

3 REVISÃO DE LITERATURA

Este capítulo apresenta uma revisão de literatura que buscou identificar os projetos relacionados à Arquitetura SAUI. Nessa revisão foram consideradas características dos projetos que abrangem as abordagens adotadas para construção da Ciência de Situação, incluindo as estratégias para modelagem do contexto e para concepção da arquitetura de software destinada ao processamento das informações contextuais, considerando a infraestrutura da IoT. Ainda, é apresentada uma discussão dos trabalhos relacionados, bem como uma análise comparativa entre estes projetos e a Arquitetura SAUI.

3.1 Trabalhos Relacionados

Esta seção apresenta as principais características e funcionalidades de trabalhos relacionados, considerando o provimento de Ciência de Situação, a partir do processamento dos dados contextuais. Nesse sentido, entende-se que este processamento abrange os aspectos relacionados à coleta, interpretação, agregação, armazenamento, consulta e inferência do contexto; visando à identificação de situações de interesse das aplicações.

Daidalos

Daidalos (GALLACHER et al., 2014) é um projeto desenvolvido por um consórcio de universidades e empresas europeias. Emprega um modelo híbrido de contexto, constituído por um banco de dados relacional para gerenciar os dados e uma ontologia que descreve tipos e associações contextuais entre entidades para capturar o conhecimento relacionado ao contexto.

A limitação de escalabilidade da ontologia é tratada através de uma visão do contexto centrada na localização, baseada no modelo relacional. Com isso, o acesso aos dados de contexto fica restrito à área em que o usuário atualmente se encontra, criando uma hierarquia de localizações, por exemplo: cidade, prédio, sala.

O projeto Daidalos apresenta um cenário de uso que caracteriza o deslocamento de uma pessoa, usando um automóvel, buscando demonstrar o acesso contínuo e ubíquo aos dados e serviços, considerando contextos de interesse dos usuários.

O cenário prevê a possibilidade da adaptação de serviços e conteúdos, bem como a mobilidade de sessão. Com isso, o usuário pode manter acesso ao seu ambiente computacional e a outros serviços inerentes ao uso de um veículo em uma rodovia, através da integração com redes de sensores das rodovias para avisar sobre situações de perigo, bem como com serviços para entretenimento e informações sobre a situação de tráfego na rodovia.

Os componentes de software do Daidalos estão organizados em serviços que constituem os módulos de uma arquitetura que viabiliza um conjunto de funcionalidades para suporte às aplicações ubíquas.

Uma camada de gerenciamento e identidade permite acesso ubíquo aos serviços. Esta camada inclui componentes que possuem funcionalidades para descoberta e composição. A descoberta deve garantir que os usuários encontrem os serviços que desejam e a composição permite que serviços oferecidos por diferentes provedores sejam compostos, agregando maior valor para os usuários. O componente de gerenciamento de identidade garante que uma infraestrutura de segurança e a privacidade possa ser aplicada ao processo de descoberta e composição de serviços. Ainda, o componente de gerenciamento da ontologia define a interoperabilidade entre serviços e funciona como um glossário para provedores e consumidores dos mesmos. O componente de gerenciamento de recursos e sessões suporta um ambiente para composição de serviços em tempo de execução, no qual operadores e provedores podem aplicar políticas em combinação com personalizações dos usuários.

A camada de gerenciamento da experiência do usuário possui um componente para coletar dados brutos de contexto a partir de vários sensores, refina esses dados e dissemina estas informações de contexto de alto nível para outros serviços da plataforma e para aplicações que os tenham requisitado. De forma similar, o componente de gerenciamento do conhecimento aprende sobre o comportamento histórico dos usuários, como interações com serviços, e usa esse conhecimento para manter atualizadas as preferências dos usuários para vários serviços. Através do componente de gerenciamento de preferências, estas podem ser aplicadas à plataforma, como mudanças nos parâmetros de qualidade de serviços (QoS), ou podem ser providas a aplicações e serviços de terceiros. O componente de negociação de privacidade trabalha com o gerenciamento de identidade para melhorar os mecanismos de negociação entre usuários e serviços.

GEPSIR

GEPSIR (CIMINO et al., 2012) é uma arquitetura de software para gerenciamento da Ciência de Situação. A proposta busca prover uma abordagem geral para Ciência de Situação, na qual tanto a arquitetura como o conhecimento comportamental podem ser integrados em um ambiente aberto, suportando uma variedade de informações contextuais, possivelmente incertas, provendo conhecimento situacional para múltiplas aplicações.

Para garantir tanto a interoperabilidade estrutural como funcional, a proposta foi concebida com base em uma abordagem orientada a agentes, a qual opera em um nível de conhecimento, mostrando um comportamento flexível, de fácil manutenção, reutilização e independência de plataforma. Isso é obtido com o uso de tecnologias padronizadas, como Web Semântica e Lógica Fuzzy para o processo de raciocínio, assim como padrões arquiteturais ECA (*Event-Control-Action*).

O modelo comportamental básico proposto para o agente de situação é expresso em termos de regras de condição. A parte antecedente de cada regra é construída através de uma combinação lógica de condições contextuais, referida como um evento no padrão ECA. A correspondente parte consequente modela uma reação ao evento, avaliando a nova situação corrente e disseminando os resultados relacionados para serviços externos. Por isso, as transições das situações correntes são estabelecidas pela base de regras, as quais representam o controle do sistema ciente de situação.

O raciocínio automatizado sobre o contexto pode lidar com eventos que ocorrem

gradualmente e condições que são vagas e imprecisas. Nesse sentido, GEPSIR adota a lógica fuzzy para associar um nível de verdade com cada condição de contexto. Caso mais de uma situação seja inferida de uma dada condição, um grau de certeza para cada situação é computado baseado nos níveis de verdade da condição. Desta forma, as situações inferidas podem ser ranqueadas com base em seus graus de certeza. Por fim, situações são associadas com um conjunto de tarefas relevantes que os usuários podem realizar em situações específicas, por meio de conhecimento de domínio expresso através de ontologias de tarefas.

A arquitetura GEPSIR emprega padrões Web como SWRL (*Semantic Web Rule Language*) e FML (*Fuzzy Markup Language*) para viabilizar portabilidade, integração e extensibilidade do conhecimento.

Ainda, considerando que as regras são usualmente criadas por especialista do domínio que modelam situações na perspectiva de utilização por um usuário médio, a abordagem proposta prevê o uso do histórico do contexto e algoritmos genéticos para adaptar o modelo situacional ao interesse de um usuário específico.

MUSIC

MUSIC (FLOCH et al., 2013) é um *middleware* que provê uma plataforma para desenvolvimento de aplicações cientes de contexto. MUSIC propõe uma abordagem orientada a serviços e baseada em componentes para aquisição e gerenciamento dos dados de contexto, bem como para adaptação das aplicações ao contexto atual.

O *middleware* está organizado em quatro partes: (i) ”Context Plug-ins”, que são responsáveis por prover os dados de contexto; (ii) ”Context Listeners”, que correspondem aos consumidores dos dados de contexto; (iii) ”Context Manager”, que gerenciam as demais partes do *middleware*; e (iv) ”Context Query Processor”, que processa as requisições de dados de contexto.

As informações de contexto são providas através de sensores, sendo que as aplicações podem acessar diretamente essas informações. Os componentes do *middleware* que gerenciam o contexto são responsáveis por executar o processo de raciocínio sobre as mudanças nas informações de contexto, identificando seu interesse para as aplicações. O desenvolvedor das aplicações deve especificar durante a programação quando uma mudança de contexto é considerada significativa para a aplicação que está sendo desenvolvida.

O *middleware* MUSIC também provê suporte para distribuição de contexto, ou seja, as informações de contexto coletadas por um nodo podem ser compartilhadas com outros nodos. Além disso, essas informações contextuais distribuídas podem ser usadas para disparar processos de adaptação em dispositivos remotos.

Com relação à modelagem do contexto, o MUSIC define uma ontologia para representar os diferentes tipos de informações contextuais. Os dados de contexto são representados com base em dois valores: a entidade, que corresponde a fonte dos dados de contexto, por exemplo, um determinado dispositivo computacional, e o escopo, que diz respeito ao tipo de informação solicitada, como memória livre.

SCENE

SCENE (PEREIRA; COSTA; ALMEIDA, 2013) é uma plataforma para gerenciamento da Ciência de Situação. A plataforma viabiliza suporte ao desenvolvimento de aplicações cientes de situação disponibilizando artefatos de projeto para especificação de situações, bem como suporte em tempo de execução para o

gerenciamento do ciclo de vida de situações, incluindo a detecção de situação, o que pode envolver o reconhecimento de padrões para composição de situações e a desativação de situações.

A plataforma SCENE permite a especificação de situações e o gerenciamento das mesmas com base em regras e eventos. Para tanto propõe uma abordagem constituída por um motor de regras e um mecanismo para processamento de eventos complexos. O uso de eventos permite raciocínio temporal. Nesse sentido, SCENE permite a realização de inferência temporal entre situações, disponibilizando operadores que consideram os eventos de ativação e desativação de situações.

A plataforma SCENE propõe um controle automático do ciclo de vida das situações. Assim, a especificação de situações é realizada por meio de uma única regra, não sendo necessário o desenvolvedor criar regras para controlar, por exemplo, a desativação de situações que não são mais válidas.

Na perspectiva da SCENE as situações são caracterizadas através de aspectos estruturais, chamados Classes de Situação, e aspectos comportamentais, denominados Regras de Situação. Uma Classe de Situação é responsável por controlar aspectos temporais e de composição da situação. Por sua vez, as Regras de Situação definem as condições nas quais um tipo de situação ocorre, controlando o ciclo de vida de instâncias de situação na memória de trabalho de uma máquina de regras.

Uma limitação da plataforma SCENE diz respeito a não ter suporte a uma abordagem distribuída para o processamento de regras de detecção de situações. SCENE não permite a detecção distribuída com múltiplos mecanismos identificando situações independentes, nem um cenário com um nível superior de distribuição, atribuindo partes da funcionalidade de detecção de regras para os diferentes motores de regras.

WComp

WComp (SEHILI et al., 2016) é um *middleware* constituído por uma infraestrutura de software, uma arquitetura de composição de serviços, e um mecanismo composicional de reação ao contexto. WComp propõe uma extensão da arquitetura orientada a serviços tradicional, denominada WSOAD (*Web Service Oriented Architecture for Device*).

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 autorregulado 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 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 (FERRY et al., 2013), o que reduz a capacidade de construção da Ciência de Situação.

3.2 Discussão dos Trabalhos Relacionados

Esta seção apresenta uma discussão dos trabalhos relacionados, considerando o objetivo central da Arquitetura SAUI de prover suporte à Ciência de Situação para as aplicações ubíquas em infraestruturas de IoT, bem como suas principais premissas de concepção.

Desta forma, a Tabela 3.1 apresenta uma comparação entre os trabalhos relacionados, contemplando aspectos centrais na concepção da Arquitetura SAUI, quais sejam: (a) arquitetura empregada em diferentes domínios de aplicação; (b) abordagem distribuída para coleta dos dados de contexto; (c) modelo híbrido para representação do contexto; (d) aquisição autônoma dos dados de contexto na infraestrutura da IoT; (e) suporte à atuação sobre o meio físico no cenário da IoT; (f) coleta e processamento de contexto com base em regras e eventos associados aos dados coletados; e (g) abordagem híbrida para processamento de contexto na identificação de situações.

Considerando os aspectos listados no parágrafo anterior e tendo por base a Tabela 3.1, na continuidade é feita uma comparação entre os trabalhos relacionados e a Arquitetura SAUI, com o intuito de identificar os principais diferenciais da SAUI, caracterizando sua contribuição científica.

Tabela 3.1: Comparação entre os Trabalhos Relacionados

	Daidalos	**GEPSIR**	**MUSIC**	**SCENE**	**WComp**
a	não	sim	sim	sim	sim
b	sim	sim	sim	não	sim
c	sim	não	não	não	não
d	não	não	não	não	não
e	não	não	não	não	sim
f	sim	sim	sim	sim	sim
g	não	sim	não	sim	não

A Arquitetura SAUI pode ser empregada em diferentes domínios de aplicação. Essa característica também é observada na maioria dos projetos relacionados, demonstrando uma tendência nos trabalhos que vêm sendo desenvolvidos no tema desta tese. Dentre os projetos estudados, apenas o Daidalos caracteriza-se por ser concebido para um domínio específico de aplicação.

Com relação à concepção da arquitetura, a maioria dos trabalhos estudados possui arquiteturas distribuídas. Entretanto, estas arquiteturas não mantêm o caráter descentralizado para todas as etapas de tratamento de contextos e situações, não atendendo os requisitos de distribuição em larga escala dos ambientes ubíquos, providos pela IoT. Por sua vez, a Arquitetura SAUI diferencia-se dos trabalhos relacionados por estar estruturada de forma largamente distribuída, o que atende a característica

de elevada dispersão de recursos na IoT, bem como por abranger todas as etapas de tratamento do contexto, eventos e situações, desde a aquisição dos dados de contexto até os procedimentos de atuação sobre o meio.

Características como padronização e expressividade são consideradas relevantes para o processamento dos dados contextuais, particularmente no que tange ao processo de raciocínio. Assim, a modelagem semântica do contexto, que implementa estas características, tem se mostrado oportuna para representação dos dados contextuais nas infraestruturas de suporte à Ciência de Situação.

Dentre os projetos estudados, observa-se que o MUSIC emprega ontologias para modelagem do contexto. Já o projeto Daidalos emprega ao mesmo tempo ontologias e banco de dados para representação do contexto, aliando a elevada expressividade de um modelo semântico, com um modelo relacional. Por sua vez, a Arquitetura SAUI diferencia-se por propor um modelo híbrido para representação e processamento do contexto, que permite tanto selecionar um modelo semântico ou sintático, em função da necessidade do domínio de problema que está sendo tratado, como permite que estes tipos de modelo sejam combinados, explorando de forma sinérgica suas características e minimizando suas limitações.

A Arquitetura SAUI provê suporte para redes de sensores e atuadores na IoT. 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 aplicações da IoT, como da atuação distribuída sobre o meio físico. Tal característica é encontrada em parte no projeto Daidalos, 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. Os demais projetos permitem o gerenciamento de sensores, porém não em uma perspectiva de rede.

De modo geral, os projetos estudados preveem o emprego de mecanismos específicos para aquisição do contexto, adotando uma estratégia de separação entre a obtenção e o uso do contexto. Essa estratégia é um dos aspectos centrais para a concepção de arquiteturas para Ciência de Situação. A Arquitetura SAUI também obtém os dados contextuais de forma independente das aplicações que os utilizam. Por outro lado, diferentemente dos projetos relacionados, a Arquitetura SAUI 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.

A maioria dos projetos estudados possui suporte ao tratamento do contexto, empregando estratégias baseadas em eventos e regras, porém esta funcionalidade usualmente está restrita a algumas etapas do processamento, principalmente à interpretação dos dados contextuais. A Arquitetura SAUI diferencia-se destes trabalhos por sua arquitetura de software ter sido concebida para dar suporte a tratadores de contexto baseados em eventos e regras, distribuídos nas células de execução do ambiente ubíquo, os quais podem estar vinculadas aos diferentes níveis de tratamento dos dados contextuais.

Os projetos GEPSIR e SCENE proveem mecanismos baseados em especificação de regras para detecção de situações que estão ocorrendo no ambiente. Por sua vez, a Arquitetura SAUI diferencia-se por possuir suporte tanto para técnicas de identificação de situações baseadas em especificação como em aprendizagem de máquina. Com isso, fica caracterizada uma abordagem híbrida para raciocínio sobre o contexto, visando à identificação de situações. Essa abordagem adotada pela Arquitetura SAUI busca combinar as características de ambas as técnicas na construção de mecanismos que viabilizem o processamento com o conhecimento e a semântica requeridos para raciocinar

sobre o contexto, bem como permitam a análise de dados brutos, extraindo padrões e lidando com a incerteza destes dados.

3.3 Considerações do Capítulo

A análise dos trabalhos relacionados realizada neste capítulo foi baseada nas premissas de concepção da Arquitetura SAUI. Esta análise buscou caracterizar os diversos desafios de pesquisa que foram considerados na proposição da SAUI, especialmente os aspectos relacionados ao processamento dos dados contextuais visando à identificação de situações, considerando uma estratégia híbrida para tratamento do contexto.

A revisão de literatura apresentada neste capítulo apontou a necessidade de uma abordagem para representação e processamento do contexto que seja capaz de lidar com a elevada distribuição do ambiente ubíquo, bem como tenha capacidade de raciocínio para identificação de situações.

A comparação da Arquitetura SAUI com os projetos relacionados caracterizou sua contribuição científica, demonstrando seu principal diferencial em relação a estes projetos, o qual corresponde ao suporte concomitante para operação distribuída, tratamento autônomo dos dados contextuais baseado em eventos e regras, e processamento híbrido do contexto, visando à identificação de situações, considerando um ambiente ubíquo cuja infraestrutura é provida pela IoT.

O próximo capítulo apresenta a concepção da Arquitetura SAUI, destacando suas principais características e funcionalidades.

4 CONCEPÇÃO DA ARQUITETURA SAUI

Este capítulo apresenta a concepção dos componentes da Arquitetura SAUI para tratamento das informações contextuais, visando prover suporte à Ciência de Situação das aplicações ubíquas, tendo em vista a infraestrutura computacional da Internet das Coisas.

O esforço de concepção da Arquitetura SAUI considera os princípios operacionais do *middleware* EXEHDA, contribuindo com o seu Subsistema de Reconhecimento de Contexto e Adaptação, ampliando suas funcionalidades de forma a atender demandas inerentes à Ciência de Situação.

Nas Seções deste Capítulo são apresentadas as características e funcionalidades da Arquitetura SAUI, sendo sistematizados: (i) o processo de identificação de situações, com base em uma abordagem híbrida definida para a representação e o processamento do contexto; e (ii) a especificação dos módulos que constituem os servidores de Contexto e de Borda que formam a arquitetura, tendo em vista sua organização de forma distribuída, e suas funcionalidades disparadas por eventos e tratadas por regras.

4.1 Visão Geral da Arquitetura SAUI

A Arquitetura SAUI busca contemplar uma abordagem: (i) distribuída, permitindo a coleta e processamento das informações contextuais em diferentes localizações, bem como a atuação sobre o meio, considerando a infraestrutura da Internet da Coisas; (ii) dirigida por eventos, viabilizando a possibilidade de associação de regras aos contextos de interesse das aplicações, as quais possuem as condições e ações associadas a um evento, sendo estas regras disparadas automaticamente em função de mudanças no estado dos contextos de interesse; e (iii) híbrida, possibilitando o uso combinado de modelos de contexto por uma determinada aplicação, bem como a utilização concomitante de modelos de contexto específicos para as diferentes aplicações em execução. Ainda, viabiliza a combinação de técnicas baseadas em especificação e aprendizagem visando à identificação de situações. Esse aspecto provê suporte ao processamento híbrido do contexto de interesse das aplicações e a decorrente identificação de situações que podem ser usadas pelas aplicações no processo de tomada de decisão para disparo de ações.

No ambiente ubíquo gerenciado pelo *middleware* EXEHDA (vide Seção 2.5.1) é instanciada a Arquitetura SAUI, com a inclusão de abstrações que ampliam a proposta original do ambiente provido pelo EXEHDA, conforme mostra a Figura 4.1. Particularmente, a SAUI inclui funcionalidades que proveem suporte para a Ciência de Situação. A proposta original do EXEHDA não contempla a Ciência de Situação, abrangendo a Ciência de Contexto nos seguintes aspectos: obtenção de dados de contexto, distribuição destes dados e tradução dos mesmos em elementos de contexto com maior

nível de abstração. Este processo de tradução possui elevada dependência das aplicações, sendo feito por algoritmos e estruturas de dados particulares para cada tipo de aplicação (vide Seção 2.5).

O ambiente ubíquo é constituído por células em que se distribuem os dispositivos computacionais, sendo seus componentes básicos: (i) EXEHDAbase: elemento central da célula responsável por todos serviços básicos e referência para os demais elementos; (ii) EXEHDAnodo: corresponde aos dispositivos computacionais responsáveis pela execução das aplicações; (iii) EXEHDAnodo móvel: tipo específico de EXEHDAnodo que corresponde aos dispositivos móveis que podem se deslocar entre as células do ambiente ubíquo, como notebooks, tablets ou smartphones; e (iv) EXEHDAborda: elemento de borda do ambiente ubíquo, responsável por fazer a interoperação entre os serviços do *middleware* e os dispositivos da Internet das Coisas.

A abordagem para tratamento do contexto na Arquitetura SAUI compreende dois tipos de servidores: o Servidor de Borda, cujos componentes de software são responsáveis pelas funcionalidades que permitem interação com o meio físico através de sensores e atuadores, e o Servidor de Contexto que é constituído por componentes de software que proveem armazenamento e processamento das informações de contexto, bem como raciocínio para identificação de situações.

Na Figura 4.1 é possível ver os servidores de Borda e Contexto da Arquitetura SAUI mapeados sobre o ambiente ubíquo. O Servidor de Borda é instanciado em equipamentos do tipo EXEHDAborda, enquanto o Servidor de Contexto é alocado no EXEHDAbase da célula.

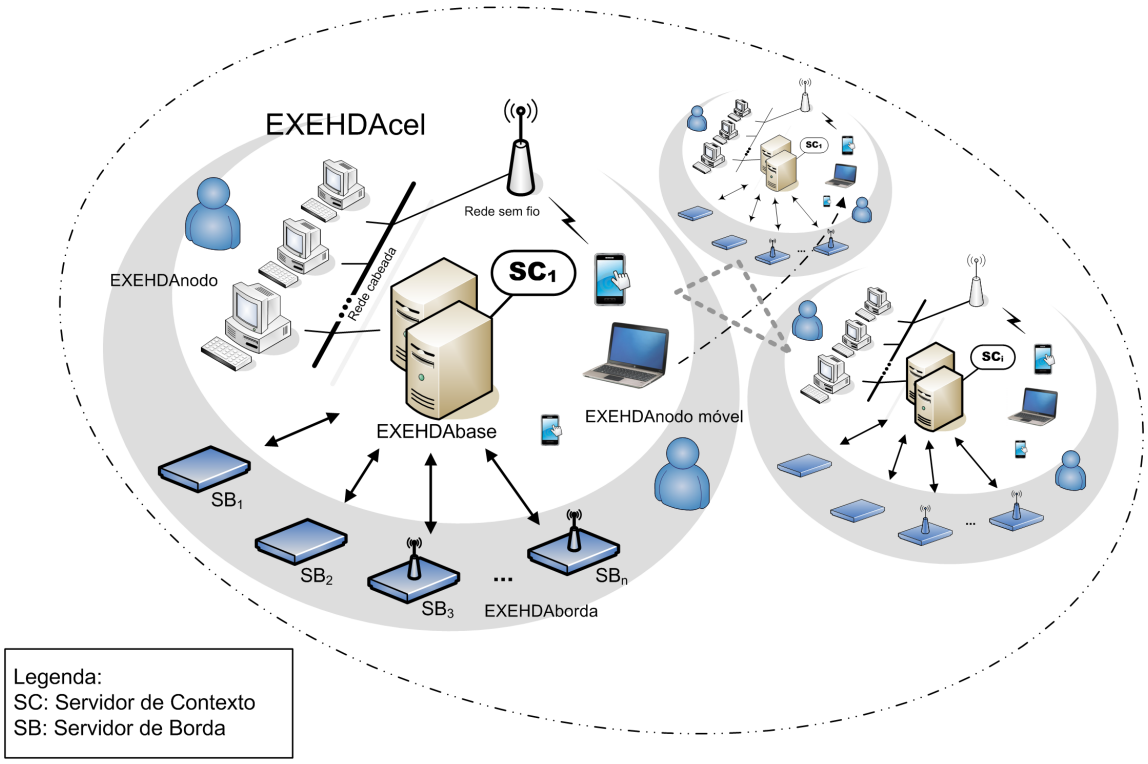


Figura 4.1: Ambiente Ubíquo Gerenciado pelo EXEHDA

A Arquitetura SAUI provê interoperabilidade: (i) entre os Servidores de Borda e o Servidor de Contexto; (ii) entre os Servidores de Contexto localizados em diferentes células do ambiente ubíquo gerenciado pelo EXEHDA; e (iii) com outros serviços do

middleware ou aplicações. Uma visão geral da Arquitetura SAUI é apresentada na Figura 4.2.

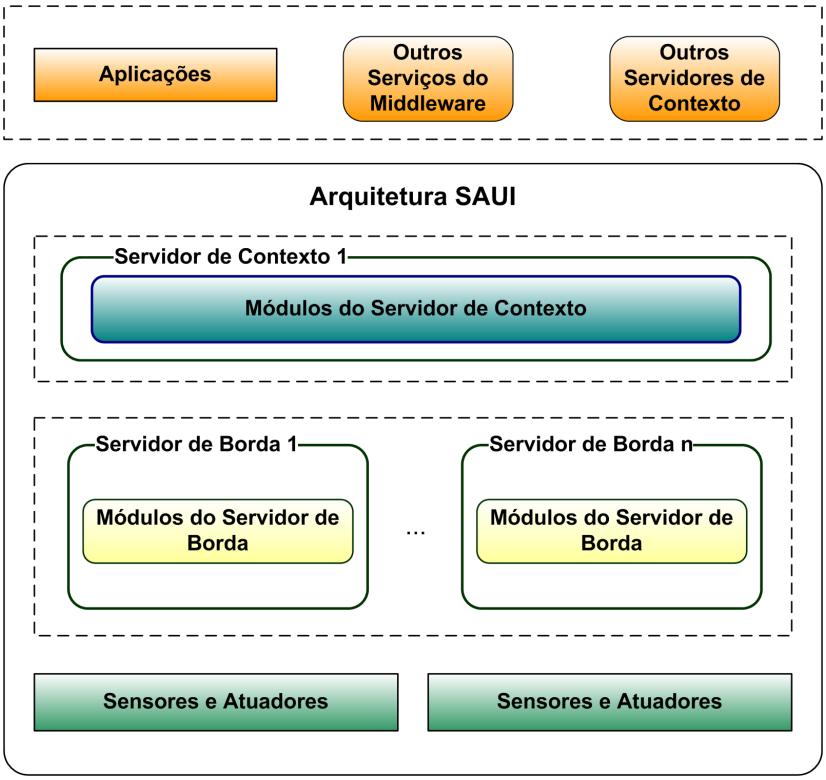


Figura 4.2: Visão Geral da Arquitetura SAUI

4.2 Suporte à Ciência de Situação na Arquitetura SAUI

O suporte à Ciência de Situação é aspecto central na concepção da Arquitetura SAUI. Este suporte corresponde à capacidade de perceber e compreender elementos contextuais de interesse instanciados, agregando significado aos mesmos de forma a prover alguma informação, válida em um intervalo de tempo específico.

Esta tese defende que uma arquitetura para prover Ciência de Situação para as aplicações ubíquas na infraestrutura da IoT deve contemplar os seguintes aspectos:

- uma camada de gerenciamento da infraestrutura computacional com suporte às demandas típicas da Internet das Coisas, responsável pelo sensoriamento e atuação;
- uma camada central, cujos componentes são responsáveis pelo tratamento dos dados de contexto sensoriados e a identificação de situações relacionadas às mudanças no estado dos contextos de interesse das aplicações;
- uma camada para interface com as aplicações. Estas aplicações são compostas por componentes distribuídos que podem ter necessidade de Ciência de Situação;
- uma estratégia para representação do contexto que compatibilize aspectos de expressividade e computabilidade, e que dê suporte às camadas da arquitetura.

Estes aspectos que proveem suporte à Ciência de Situação na Arquitetura SAUI são mostrados na Figura 4.3, destacando o foco central desta tese nos componentes de processamento do contexto e identificação de situações, bem como no modelo para representação do contexto.

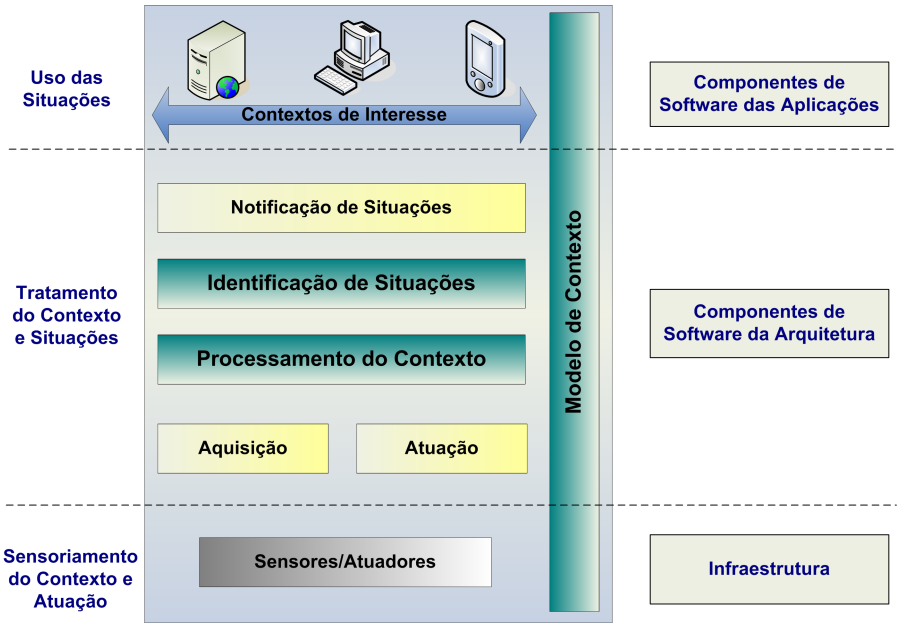


Figura 4.3: Ciência de Situação na Arquitetura SAUI

4.3 Abordagem Híbrida na Arquitetura SAUI

Uma etapa necessária para processar as informações contextuais é a modelagem do contexto. Um modelo estrutural de contexto define os elementos básicos do domínio e forma a base para a identificação de situações.

O modelo de contexto na Arquitetura SAUI é tratado por dois atores: (i) desenvolvedor de aplicações, responsável pela lógica de negócios específica da aplicação e sua interface com a arquitetura; e (ii) desenvolvedor do *middleware*, responsável por disponibilizar na arquitetura a estrutura necessária para armazenamento e processamento das informações contextuais.

A representação de contexto da Arquitetura SAUI provê modelos semânticos e sintáticos, os quais oferecem suporte para armazenamento e consulta dos dados contextuais coletados, bem como para raciocínio sobre os contextos que estes representam.

O modelo sintático proposto para a Arquitetura SAUI corresponde à representação direta das informações contextuais, sem agregação de significado às mesmas. Por sua vez, o modelo semântico corresponde à representação de entidades do modelo e seus relacionamentos em uma perspectiva semântica que considera o significado das mesmas, sendo implementado principalmente através de ontologias.

Considerando os requisitos das aplicações, estas podem empregar tanto um modelo semântico, como um modelo sintático, bem como integrar estes dois tipos de modelo, visando combinar suas características e minimizar suas limitações. Nesse sentido, a integração de diferentes modelos pode propiciar maior flexibilidade e generalidade na construção de aplicações cientes de situação, minimizando limitações apresentadas

individualmente pelos modelos existentes (vide Seção 2.2.2). Ainda, a proposta pode se ajustar às necessidades do domínio do problema, em função da escolha das técnicas de modelagem mais adequadas à aplicação.

Deste modo, na Arquitetura SAUI é definida uma abordagem híbrida para representação e processamento do contexto, conforme mostra a Figura 4.4).

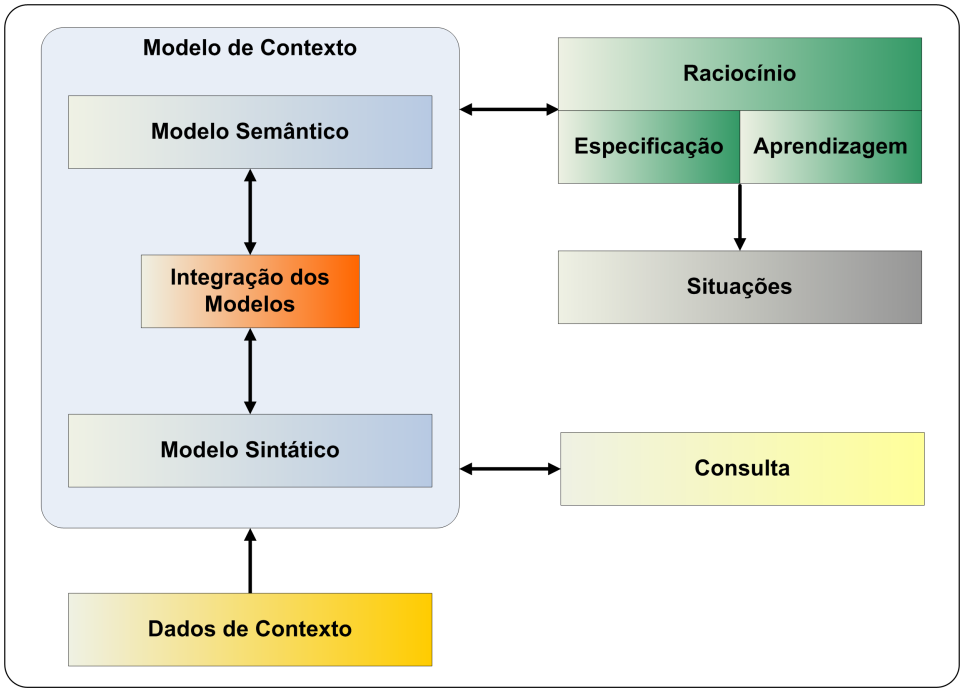


Figura 4.4: Abordagem Híbrida para Representação e Processamento do Contexto na Arquitetura SAUI

A abordagem híbrida para processamento do contexto na Arquitetura SAUI abrange ainda o processo de raciocínio para identificação de situações. Nesse sentido, são empregadas técnicas baseadas em especificação e aprendizagem para raciocínio sobre as informações contextuais que se encontram armazenadas de acordo com o modelo de contexto.

A combinação destas técnicas pode ser oportuna para dar suporte a identificação de situações em ambientes ubíquos que se caracterizam por serem altamente dinâmicos. As técnicas baseadas em especificação podem viabilizar o processamento do contexto com o conhecimento e a semântica requeridos para raciocinar sobre o mesmo. Por sua vez, as técnicas baseadas em aprendizagem podem permitir a análise de dados brutos, extraindo padrões e lidando com a incerteza destes dados.

Uma abordagem híbrida possibilita a combinação sinérgica de diferentes técnicas de identificação de situações, podendo ser capaz de lidar com a incerteza dos dados sensoriados e as regras de raciocínio. Nesse sentido, uma abordagem para identificação de situações deve incorporar primitivas suficientemente ricas para representar diferentes tipos de informações, incluindo dados sensoriados, contextos de interesse, conhecimento de domínio e de situação, e as regras para verificação da consistência e integridade da base de conhecimento.

4.4 Abordagem Baseada em Eventos e Regras na Arquitetura SAUI

Na concepção da Arquitetura SAUI, no que diz respeito a sua operação, é empregada uma abordagem baseada em eventos para tratamento dos dados contextuais, a qual viabiliza a possibilidade de associação de regras aos contextos de interesse das aplicações.

Os eventos, na perspectiva da SAUI, dizem respeito a mudanças no estado do contexto, tais como: uma propriedade do ambiente físico (temperatura fora da faixa definida, por exemplo) que sejam de interesse das aplicações.

Os eventos disparam regras para tratamento dos dados contextuais. Quando são atendidas as condições de uma regra de contexto é disparado um processo de raciocínio sobre as informações contextuais visando à identificação de situações relacionadas aos contextos de interesse.

Tanto as regras de contexto como os mecanismos empregados para o processo de raciocínio interagem com os dados contextuais armazenados de acordo com o modelo de contexto.

A ocorrência de uma situação gera ações da Arquitetura SAUI, as quais podem corresponder ao envio de alertas para os usuários, bem como ao controle processos de atuação sobre o meio.

A Figura 4.5 mostra a abordagem baseada em eventos e regras definida para Ciência de Situação na Arquitetura SAUI.

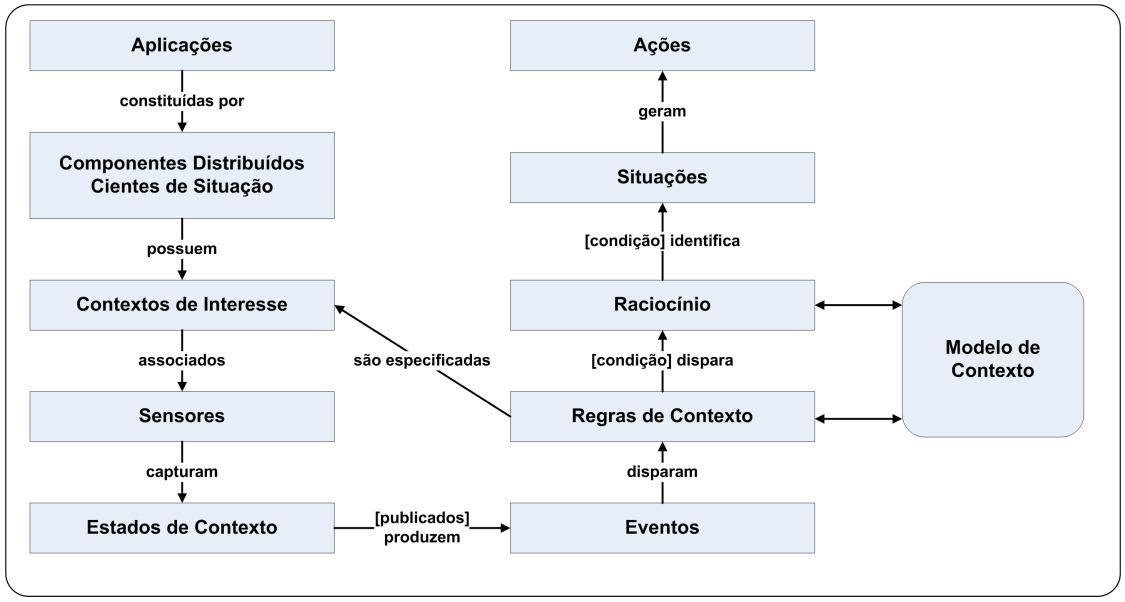


Figura 4.5: Abordagem Baseada em Eventos e Regras na Arquitetura SAUI

4.5 Funcionalidades da Arquitetura SAUI

A descrição dos módulos previstos para a Arquitetura SAUI, apresentada nas próximas seções, 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.

4.5.1 Arquitetura SAUI: Servidor de Contexto

Esta seção apresenta a descrição dos módulos que constituem o Servidor de Contexto, bem como a estrutura de processamento prevista para tratamento dos dados contextuais neste servidor.

Módulos do Servidor de Contexto

Os módulos da Arquitetura SAUI que constituem o Servidor de Contexto interoperam no provimento das funcionalidades para suporte à Ciência de Situação. Estes módulos, descritos a seguir, são responsáveis pelas etapas que abrangem desde a aquisição do contexto, a partir dos dados coletados através dos Servidores de Borda, até o momento em que este é armazenado e processado com o intuito de identificar situações, cuja ocorrência pode ser notificada às aplicações e/ou aos demais serviços do *middleware* EXEHDA.

Uma visão geral da arquitetura do Servidor de Contexto é apresentada na Figura 4.6, na qual é caracterizada a relação com (i) Servidores de Borda; (ii) outros serviços do *middleware* EXEHDA; (iii) outros Servidores de Contexto remotos; e (iv) aplicações de usuários e administradores.

Módulo de Comunicação

O Módulo de Comunicação foi concebido para gerenciar a troca de informações com os Servidores de Borda, bem como com os demais serviços do EXEHDA, inclusive outros Servidores de Contexto remotos, e com as aplicações.

A concepção deste módulo tem como referência o estilo arquitetural REST (*REpresentational State Transfer*) (FIELDING, 2000) (vide Apêndice A para informações a respeito de conceitos, princípios e características do REST). O protocolo HTTP (*Hypertext Transfer Protocol*) e suas operações (GET, PUT, POST, DELETE) é utilizado no suporte às trocas de informações.

O Módulo de Comunicação provê o suporte necessário para disponibilização de uma API (*Application Programming Interface*) REST para acesso aos recursos oferecidos pelo Servidor de Contexto. Com isso, a interoperação ocorre por meio de operações HTTP sobre URIs (*Uniform Resource Identifier*), as quais são o caminho para o acesso a todos os recursos disponibilizados pela arquitetura.

A utilização de REST como modelo de interoperação proporciona a construção de uma interface padronizada e independente da linguagem de programação utilizada, tanto na implementação dos recursos oferecidos quanto no uso dos mesmos.

Ainda, o Módulo de Comunicação é utilizado por Servidores de Contexto remotos e/ou aplicações para solicitação de dados de contexto armazenados no RHIC (Repositório Híbrido de Informações de Contexto) ou coletados de forma instantânea pelos sensores, bem como para subscrição em contextos de interesse, os quais tem seus dados coletados, armazenados e processados com o intuito de identificar situações, cuja ocorrência pode ser notificada pela Arquitetura SAUI aos consumidores interessados.

Além destas funcionalidades, o Módulo de Comunicação também é constituído pelos componentes Gerenciador de Configuração e Gerenciador de Acesso Móvel.

O Gerenciador de Configuração tem por objetivo permitir aos usuários administradores um gerenciamento das configurações do Servidor de Contexto. Esse componente provê funcionalidades para que sejam especificados os diferentes aspectos dos sensores e atuadores.

Por sua vez, o componente Gerenciador de Acesso Móvel tem como premissa que

a disponibilização de alertas em uma plataforma de hardware que possa acompanhar o usuário, enquanto este desempenha suas atividades em diferentes lugares, se mostra um procedimento que potencializa a ubiquidade da solução de Ciência de Situação provida pela Arquitetura SAUI. Nesse sentido, este componente provê acesso às funcionalidades da Arquitetura SAUI através de dispositivos móveis baseados em Android. O Gerenciador de Acesso Móvel está organizado em dois blocos: (i) Bloco de Exibição de Informações Contextuais, que disponibiliza ao usuário relatórios, tanto gráficos como textuais, acerca das informações contextuais de seu interesse; e (ii) Bloco de Tratamento de Alertas, que tem por objetivo enviar notificações ao usuário, quando ocorrem estados de contextos de seu interesse.

Módulo de Aquisição

O Módulo de Aquisição é responsável por prover suporte à captura das informações contextuais, coletadas pelos Servidores de Borda considerando dois tipos de sensores: sensores lógicos, implementados através de interfaces de software, ou sensores físicos providos por interfaces de hardware.

Esse módulo provê suporte tanto para o procedimento de coleta agendada dos dados contextuais, como para obtenção instantânea dos mesmos por solicitação de consumidores interessados.

A obtenção de dados de contexto pelo Módulo de Aquisição acontece através de publicações do Servidor de Borda, o qual considera os seguintes parâmetros: (i) o intervalo de tempo entre medições; (ii) a flutuação mínima do valor coletado para que aconteça a publicação; e (iii) a inserção na faixa de valores que deverão ser publicados.

Módulo de Atuação

O Módulo de Atuação requisita ao Servidor de Borda procedimentos sobre os atuadores. Estes procedimentos compreendem comandos de ativação e desativação dos atuadores, bem como de parametrização de seu comportamento operacional.

Esse módulo recebe o identificador do atuador e os parâmetros operacionais a serem utilizados e interopera com o Servidor de Borda para controle do atuador pertinente.

Desta forma, o Módulo de Atuação é responsável por disparar no ambiente ubíquo ações que mudam o estado do meio, materializando a ação dos mecanismos de Ciência de Situação providos pela Arquitetura SAUI.

Módulo de Notificação

O Módulo de Notificação é responsável por disseminar o resultado do processamento das informações de contexto e da consequente identificação de situações realizada pelo Módulo de Processamento.

Esse módulo opera recebendo subscrições dos serviços e/ou aplicações que desejem notificações a respeito dos estados contextuais e das situações identificadas, interoperando através do Módulo de Comunicação. De modo específico, o Módulo de Notificação possui as seguintes funcionalidades:

- Realizar notificação a respeito da variação do estado das informações contextuais e da decorrente identificação de situações aos outros serviços do *middleware* EXEHDA;

- Receber subscrições de aplicações que necessitam de informações sobre o estado

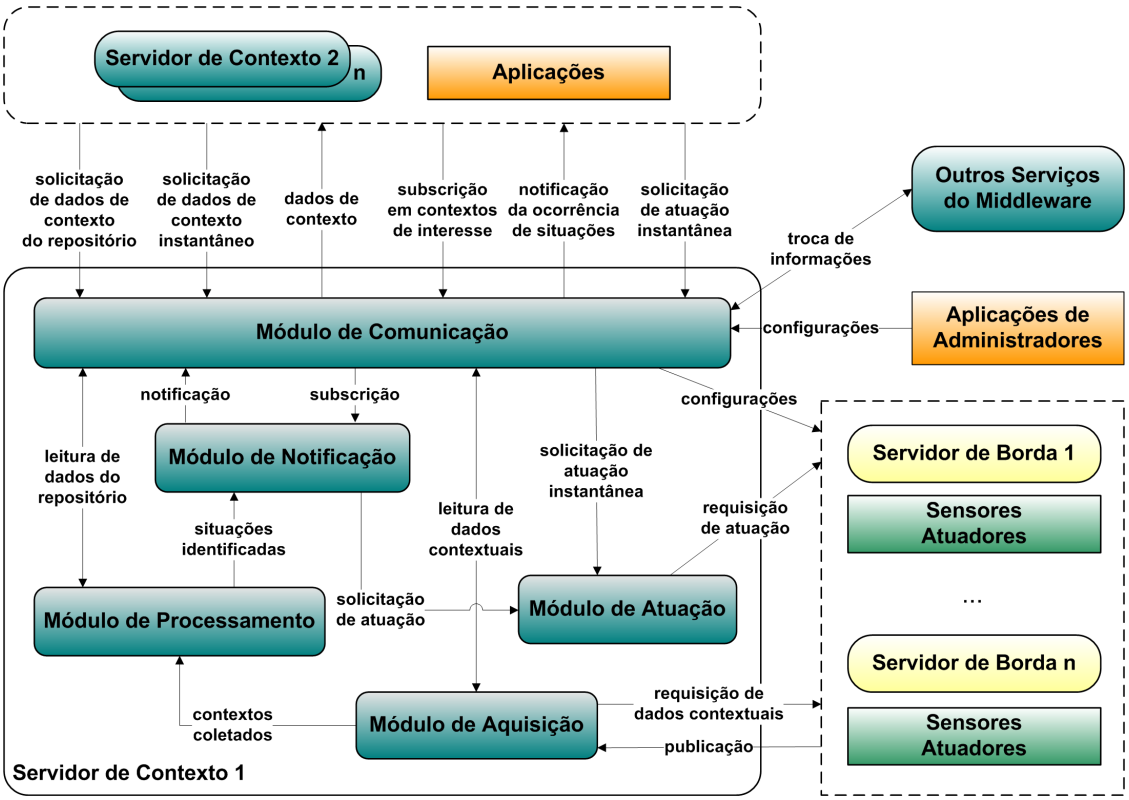


Figura 4.6: Arquitetura SAUI: Servidor de Contexto

dos contextos de interesse e as situações detectadas, notificando-as com as informações desejadas;

- Repassar ao Módulo de Atuação as decisões referentes às alterações a serem realizadas no meio.

Desta forma, passam pelo Módulo de Notificação todas as decisões de atuação provenientes do tratamento das regras de processamento do contexto realizadas pelo Módulo de Processamento.

Módulo de Processamento

O Módulo de Processamento (vide Figura 4.7) tem como principal função realizar as tarefas pertinentes ao tratamento das informações contextuais para identificar situações relacionadas aos contextos de interesse das aplicações. Para execução destas tarefas, esse módulo é constituído por quatro componentes e um conjunto de repositórios que estão descritos a seguir.

Gerenciador de Modelos

O componente Gerenciador de Modelos possui funcionalidades que permitem a representação do contexto através de modelos semânticos e sintáticos, considerando a proposta de uma modelagem híbrida do contexto prevista para a Arquitetura SAUI (vide Seção 4.3).

Diferentes técnicas de modelagem para representação do contexto podem ser empregadas em função do domínio do problema. Por exemplo, uma aplicação que trate a descrição de perfis (pessoas, equipamentos) pode empregar uma técnica de modelagem

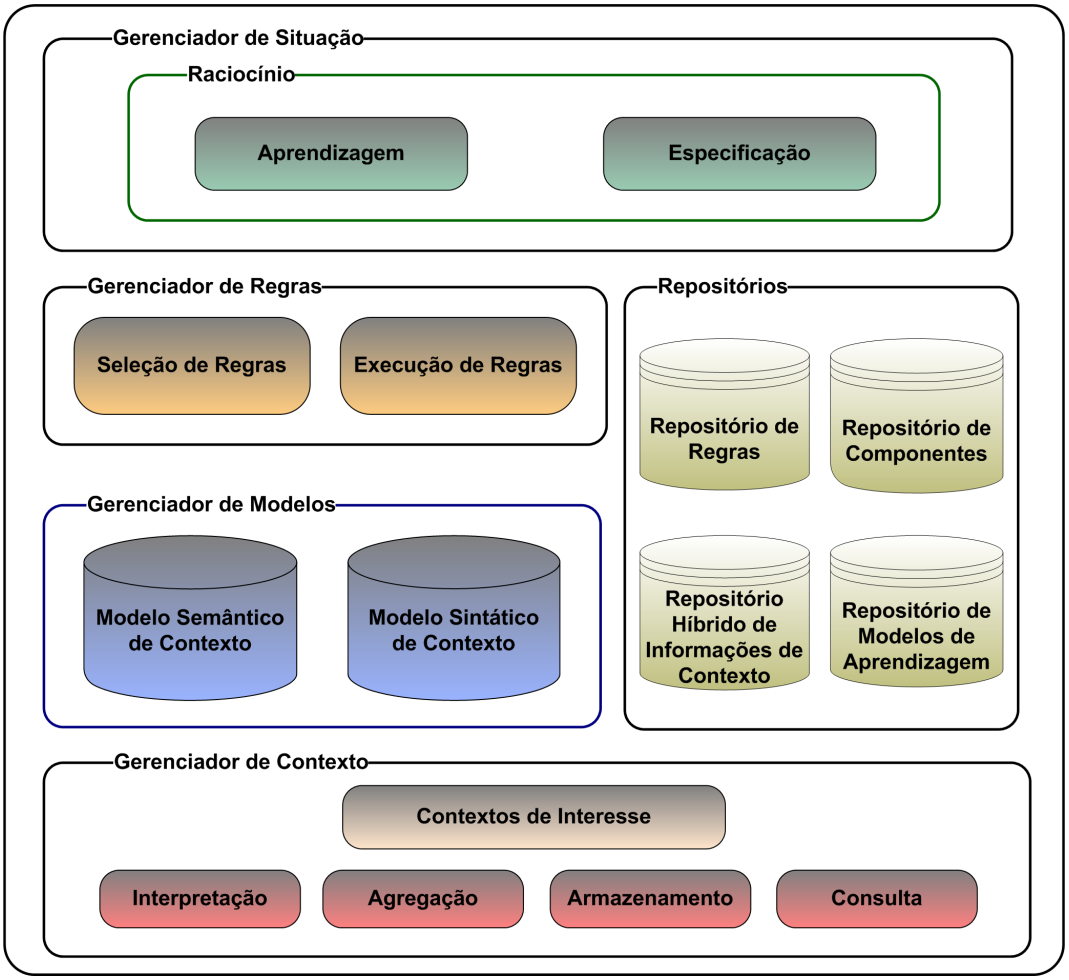


Figura 4.7: Arquitetura SAUI: Módulo de Processamento do Servidor de Contexto

que use linguagem de marcação XML. Aplicações para consulta e visualização de dados podem ter o contexto representado através de um banco de dados relacional. Um domínio de aplicação que exija raciocínio sobre os dados de contexto pode utilizar um modelo ontológico. Ainda, estas técnicas podem ser combinadas, na perspectiva de uma modelagem híbrida. No exemplo, a integração de linguagem XML e banco de dados constituiriam o modelo sintático, por sua vez a ontologia caracterizaria o modelo semântico de contexto.

Assim, este componente possibilita que os modelos de contexto sejam gerenciados de forma independente, provendo suporte para as várias aplicações em execução empregarem de forma concomitante cada uma sua técnica específica de modelagem do contexto. Além disso, o Gerenciador de Modelos permite que as aplicações possam empregar de forma combinada modelos de representação sintática e semântica do contexto.

As informações contextuais são armazenadas em um repositório híbrido denominado RHIC. Este repositório viabiliza diferentes formas de armazenamento, provendo o suporte para o registro das informações de contexto, tanto sintáticas como semânticas. O Gerenciador de Modelos é responsável por disponibilizar os métodos que possibilitam o acesso às informações presentes no RHIC.

Gerenciador de Contexto

O componente Gerenciador de Contexto é responsável por prover as funcionalidades necessárias ao processamento das informações contextuais, abrangendo as etapas de interpretação, agregação, armazenamento e consulta. Considerando os Contextos de Interesse das aplicações, este processamento tem como objetivo elevar o grau de abstração dos dados de contexto coletados, melhorando sua disponibilidade e usabilidade no processo de identificação de situações. Os componentes de software para gerenciamento do contexto são armazenados no Repositório de Componentes, sendo acessados através de APIs do tipo REST.

O Gerenciador de Contexto realiza o processamento dos dados contextuais. Dentre outras funcionalidades, possibilita consultas sobre as instâncias e a correspondente visualização dos dados de contexto, sendo tratadas as informações que estão instanciadas através dos modelos sintáticos no Repositório Híbrido de Informações de Contexto, não realizando um processo de raciocínio sobre os dados contextuais.

Após o processamento pelo Gerenciador de Contexto, os dados contextuais armazenados podem ser tratados pelo mecanismo de Raciocínio do componente Gerenciador de Situação, sendo possível aplicar técnicas de raciocínio baseadas em especificação e/ou aprendizagem sobre os dados de contexto, com objetivo de agregar significado aos mesmos e como consequência identificar situações.

Gerenciador de Regras

Este componente disponibiliza as funcionalidades para seleção e execução das regras associadas ao processamento das informações de contexto.

As regras são armazenadas no Repositório de Regras, o qual é acessado pelo mecanismo de Seleção de Regras com o intuito de selecionar a regra a ser executada. A seleção das regras ocorre em função dos eventos produzidos a partir de mudanças no estado dos contextos de interesse, tratados pelo Gerenciador de Contexto, bem como por regras relacionadas ao processo de raciocínio para identificação de situações gerenciado pelo Gerenciador de Situação.

O Gerenciador de Regras emprega regras do tipo ECA (Evento-Condição-Ação) para tratar eventos gerados a partir de mudanças nos estados dos contextos de interesse, que podem acarretar no disparo de ações pertinentes em função do estado contextual. A recepção de uma publicação de valores sensoriados a partir do Servidor de Borda é o gatilho para o disparo de regras.

Embora este componente utilize regras ECA como mecanismo básico de tratamento do contexto, tanto a condição a ser tratada quanto a ação a ser executada pela regra admitem outros modelos de processamento que podem ser chamados pela regra, os quais são decorrência do tipo de domínio de aplicação.

Considerando que as regras podem estar associadas aos componentes Gerenciadores de Contexto e de Situação, as mesmas têm a possibilidade de tratar desde aspectos relacionados à interpretação e agregação de dados de contexto até processos de raciocínio para identificação de situações.

Também, foi desenvolvido para o Gerenciador de Regras um conjunto de funcionalidades para processamento das informações de contexto que possui como característica principal a capacidade de combinar em uma mesma regra as informações presentes nas diferentes formas de armazenamento sintático e semântico providas pelo Gerenciador de Modelos, instanciadas no RHIC. Esta estratégia, denominada na

Arquitetura SAUI de Regras de Integração dos Modelos, utiliza *tags* de marcação na regra criada, as quais são substituídas pelas informações que serão buscadas no RHIC. Na Seção 5.1.2 é descrito um cenário de uso que detalha as funcionalidades da Arquitetura SAUI desenvolvidas para composição de regras.

Gerenciador de Situação

O componente Gerenciador de Situação disponibiliza as funcionalidades necessárias para o processo de raciocínio sobre os dados contextuais, visando à identificação de situações.

Considerando a proposta de uma abordagem híbrida para a Arquitetura SAUI, o mecanismo de Raciocínio emprega de forma combinada técnicas baseadas em aprendizagem e em especificação para identificar as situações relacionadas aos contextos de interesse das aplicações. Para tanto, o mecanismo de Raciocínio está organizado em dois blocos: Aprendizagem e Especificação.

O bloco de Aprendizagem permite que os algoritmos de aprendizagem possam ser parametrizados e utilizados de maneira transparente para as aplicações, evitado desta forma que detalhes do seu funcionamento necessitem de tratamento por parte do desenvolvedor das aplicações.

Para permitir a abstração das técnicas de identificação de situações baseadas em aprendizagem neste bloco é incluído um recurso de treinamento, o qual utiliza o mesmo modelo de dados independentemente do algoritmo escolhido. Dessa forma, o usuário fornece os dados para o treinamento, escolhe a técnica que deseja utilizar e as regras para extração de características que devem ser executadas sob os dados de entrada. Após a execução do treinamento, os parâmetros obtidos são armazenados no repositório Modelos de Aprendizagem, podendo ser acessados durante a execução das aplicações.

Os algoritmos de processamento são armazenados na forma de componentes de software no Repositório de Componentes, sendo possível a inclusão de novos componentes. As interfaces dos componentes devem ser construídas de acordo com o modelo de interoperabilidade padronizado para o módulo em questão.

O bloco de Aprendizagem ainda possui um recurso para extração de características que é responsável por pré-processar as variáveis de contexto ou sinais lidos diretamente de sensores, extraindo o conjunto de características relevantes para o treinamento e posterior execução do algoritmo de aprendizagem. As rotinas para pré-processamento, assim como os algoritmos de aprendizagem, são componentes de software armazenados no Repositório de Componentes, sendo sua execução disparada através de regras criadas por um usuário administrador do *middleware*. Como exemplos de algoritmos para extração de características, podem ser citados métodos estatísticos como média, variância, covariância, máximo e mínimo, ou ainda, métodos de processamento de sinais como FFT (*Fast Fourier Transform*).

O bloco de Especificação permite que sejam empregadas técnicas baseadas em especificação de regras para inferência de situações. Considerando tanto os dados de contexto coletados, como os resultantes do processamento realizado no bloco de Aprendizagem, este bloco executa um processo de normalização que transforma os valores numéricos destes dados contextuais para o domínio da técnica de especificação empregada. Por exemplo, no cenário de uso da Seção 5.1.3, é coletado o sinal vital frequência cardíaca de pacientes, o qual é convertido para o domínio fuzzy através de uma função de pertinência triangular.

As regras relacionadas ao tipo de técnica de identificação de situação adotada,

aprendizagem ou especificação, são armazenadas no Repositório de Regras e processadas pelo componente Gerenciador de Regras.

Estrutura de Processamento dos Dados de Contexto

A estrutura de processamento dos dados contextuais no Servidor de Contexto, visando à identificação de situações, contempla dois fluxos: (i) os Servidores de Borda executam a coleta de dados contextuais do meio de forma autônoma; e (ii) outros Servidores de Contexto podem disponibilizar dados contextuais que serão considerados no processamento dos dados coletados.

A estrutura de processamento foi concebida para permitir acesso através de APIs do tipo REST para o Gerenciador de Regras do Módulo de Processamento. Na Figura 4.8 pode ser vista a representação de um componente de processamento da arquitetura **<Componente-Processamento/id>**. Para compor um fluxo de processamento contextual são criadas instâncias dos componentes de processamento, alocadas em URIs (*Uniform Resource Identifier*) do tipo **<Instancia-Processamento/id>**.

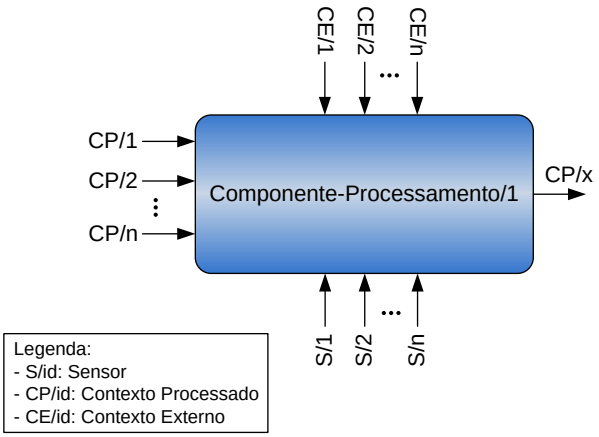


Figura 4.8: Componente de Processamento Elementar

As regras de processamento dos dados contextuais podem ser compostas pela combinação de componentes de processamento. A execução das instâncias destes componentes é disparada por eventos de modificação nos dados de entrada, correspondendo a mudanças no estado dos contextos de interesse. Cada instância deve possuir como entrada ao menos uma URI para cada sensor, contexto processado, ou contexto externo. A saída do processamento é armazenada em uma URI do tipo **<Contexto-Processado/id>**. Em cada instância de processamento estão envolvidos quatro tipos de URIs:

- **Sensores** <Sensor/id>: sensor gerenciado por um Servidor de Borda.
- **Contextos Processados** <Contexto-Processado/id>: saída de uma instância de um componente de processamento, representa uma informação de contexto processado que poderá ser utilizada na identificação de situações ou composição de outros contextos processados.
- **Contextos Externos** <Contexto-Externo/id>: dados de contexto requisitados a outros Servidores de Contexto, recebidos através do Módulo de Comunicação.

- **Componente de Processamento** <Componente-Processamento/id>: componente de software armazenado no Repositório de Componentes do Módulo de Processamento do Servidor de Contexto.

A Figura 4.8 mostra que os componentes de processamento de contexto podem receber dados contextuais processados por outros módulos, permitindo a criação de arranjos de processamento, conforme é exibido na Figura 4.9.

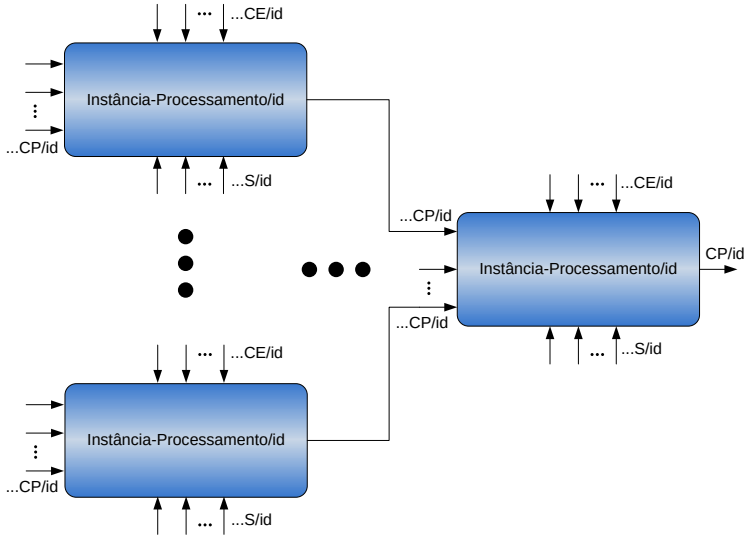


Figura 4.9: Processamento de Contexto Combinando Múltiplas Instâncias de Componentes de Processamento

A descrição da API do tipo REST pode ser vista na Tabela 4.1. Os recursos quando acessados sem um ID possuem somente o comando GET, para o qual é retornada a lista de recursos disponíveis. Quando é especificado um ID, podem ser empregados os comandos GET, POST, PUT e DELETE, os quais possuem o comportamento padrão do protocolo HTTP.

4.5.2 Arquitetura SAUI: Servidor de Borda

A arquitetura concebida para o Servidor de Borda é formada por módulos destinados a tratar a rede de sensores e/ou atuadores, bem como efetuar as publicações dos dados contextuais coletados no Servidor de Contexto. A arquitetura do Servidor de Borda é mostrada na Figura 4.10.

Módulo de Interoperação

O Módulo de Interoperação é destinado a gerenciar a comunicação com o Servidor de Contexto, tendo como referência o estilo arquitetural REST. Com isso, todas as funcionalidades associadas aos sensores e atuadores são mapeadas na forma de recursos e acessadas via URIs.

Através desse módulo é disponibilizada uma interface padronizada para manipulação dos dispositivos conectados ao Servidor de Borda, abrangendo aspectos como: coleta, configuração, acionamento, inserção e remoção de sensores ou atuadores. A manipulação destes dispositivos pode ocorrer a partir de diferentes serviços do *middleware*. Essa funcionalidade do Módulo de Interoperação é importante para o desenvolvimento de

Tabela 4.1: Recursos da API do Tipo REST para Processamento do Contexto

URI	Comandos	Descrição
Sensor	GET	Lê o conjunto de sensores disponíveis
Sensor/id	GET	Lê os dados de um sensor
Contexto-Processado	GET	Lê o conjunto de Contextos Processados
Contexto-Processado/id	POST, GET, PUT e DELETE	Operações sobre Contextos Processados, incluindo leitura do valor atual
Contexto-Externo	GET	Lê o conjunto de Contextos Externos
Contexto-Externo/id	POST, GET, PUT e DELETE	Operações sobre Contextos Externos, incluindo leitura do valor atual
Componente-Processamento	GET	Lê o conjunto de Componentes de Processamento
Componente-Processamento/id	POST, GET, PUT e DELETE	Operações sobre Componentes de Processamento
Instancia-Processamento	GET	Lê o conjunto de instâncias de Componentes de Processamento criadas para a aplicação
Instancia-Processamento/id	POST, GET, PUT e DELETE	Operações sobre uma dada instância de um componente de processamento

aplicações em ambientes ubíquos, altamente distribuídos e heterogêneos, pois restringe ao Servidor de Borda o tratamento dos aspectos tecnológicos envolvidos na interação com o meio, evitando que isso se reflita no restante da arquitetura.

Módulo de Coleta

Este módulo provê o tratamento de uma rede de sensores permitindo que cada sensor possa ser lido por parâmetros individualmente configurados. Este tratamento é responsável por aspectos desde a gerência física (interfaces, frequência de leitura) até a normalização computacional (validação, tradução) dos valores coletados. Também, este módulo provê funcionalidades para publicação das informações coletadas pela rede de sensores no Servidor de Contexto.

O componente Parâmetros Operacionais dos Sensores especifica o *driver* a ser utilizado para leitura dos diferentes sensores, bem como a agenda de aquisição dos 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. O componente Agendador, a partir dos Parâmetros Operacionais dos Sensores, dispara

leituras considerando o interesse do usuário. O Agendador emprega o relógio do Servidor de Borda para o disparo dos seus procedimentos, sendo possível especificar a frequência com que estes disparos ocorrem.

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

O componente Validador avalia se uma determinada coleta de dados sensoriados deve ser publicada ou não, empregando para isso critérios especificados pelo usuário. Esses critérios são baseados em regras que estão identificadas no componente Parâmetros Operacionais dos Sensores, a partir do ID do sensor (IDs). Por exemplo, com o intuito de minimizar custos de comunicação na rede, um valor sensoriado cuja diferença entre o último valor lido e o antepenúltimo for menor que a precisão operacional do sensor, a critério do usuário, poderá não ser publicado.

O componente Tradutor objetiva adequar o dado coletado à natureza da aplicação do usuário, por exemplo, faixas de temperatura podem ser convertidas em descrições do tipo “Alta”, “Média”, “Baixa”, através de uma regra. O componente Leitura Instantânea tem 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 recebe de forma assíncrona as solicitações e, a partir do ID do sensor, dispara o *driver* correspondente.

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. Estas regras atuam sobre o mecanismo de atuação, ativando ou desativando atuadores. Esse componente é ativado sempre que ocorrer a leitura de um determinado sensor, adquirindo elevada importância quando a comunicação com o Servidor de Contexto for interrompida por problemas de infraestrutura de rede.

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 em que a publicação ficar inviabilizada.

O componente Publicador tem a função de disparar as requisições de envio de informações contextuais para as demais camadas do *middleware*, trocando dados com o Servidor de Contexto através do Módulo de Interoperação.

As publicações são organizadas em um sistema de fila FIFO (*First In, First Out*) e são processadas conforme a disponibilidade da rede. Considerando as possíveis falhas de comunicação entre o Servidor de Borda e o Servidor de Contexto, bem como eventuais atrasos da rede, o componente Persistência Local realiza o armazenamento temporário da fila de informações contextuais até que as mesmas sejam publicadas.

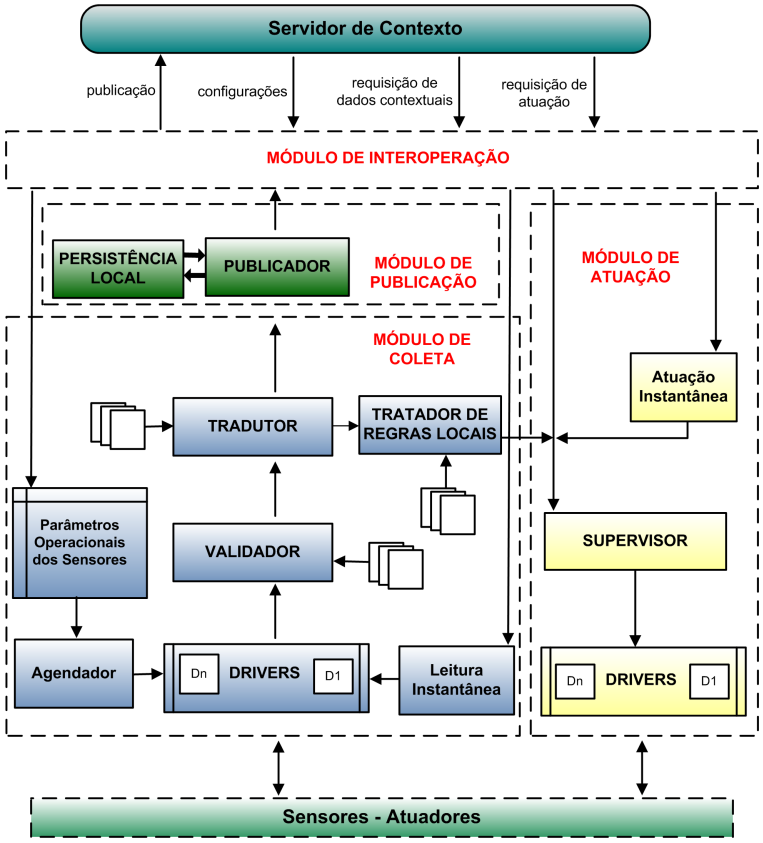


Figura 4.10: Arquitetura SAUI: Servidor de Borda

Módulo de Atuação

O Módulo de Atuação é responsável pelo gerenciamento dos atuadores. O componente Atuação Instantânea é similar ao componente Leitura Instantânea dos sensores, recebendo comandos assíncronos, a qualquer momento. Esses comandos são originários tanto do Servidor de Contexto, em decorrência 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.

O componente Supervisor aglutina comandos de atuação provenientes de três fontes distintas: (i) atuação regular; (ii) atuação instantânea; e (iii) Tratador de Regras Locais. Uma vez recebidos os parâmetros para controle da atuação, o componente Supervisor tem como funcionalidade ativar o *driver* necessário para o atuador que está sendo gerenciado.

Com o intuito de identificar comportamentos anômalos no que diz respeito à gerência dos atuadores, o Supervisor 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.

Os *drivers* dos atuadores têm objetivo similar aos 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.

4.6 Considerações do Capítulo

A concepção da Arquitetura SAUI, direcionada ao provimento de Ciência de Situação para as aplicações ubíquas na perspectiva da infraestrutura da Internet das Coisas, foi apresentada neste Capítulo.

As premissas de concepção da SAUI consideram os aspectos operacionais do *middleware* EXEHDA, bem como buscam contemplar uma abordagem distribuída, dirigida por eventos e regras no que diz respeito a operação da arquitetura, e híbrida quanto ao processamento dos contextos de interesse das aplicações, visando à identificação das situações relacionadas aos mesmos.

Este capítulo descreveu a abordagem para tratamento do contexto definida para a Arquitetura SAUI, a qual prevê dois tipos de servidores: (i) o Servidor de Contexto que provê armazenamento e processamento das informações contextuais, viabilizando o processo de raciocínio para identificação de situação; e (ii) o Servidor de Borda que é responsável por coletar e pré-processar os dados contextuais, publicando-os no Servidor de Contexto.

Os módulos previstos para a Arquitetura SAUI foram descritos neste capítulo, tendo sido organizados em função dos servidores de Contexto e de Borda, bem como suas respectivas funcionalidades.

Ainda, foi apresentada a estrutura de processamento dos dados contextuais concebida para o Servidor de Contexto, a qual contempla o emprego de APIs do tipo REST para processar os dados tanto coletados pelos Servidores de Borda, como requisitados a outros Servidores de Contexto.

No próximo capítulo é apresentada a avaliação das funcionalidades da Arquitetura SAUI, tendo por base cenários de uso nas áreas de agropecuária e saúde.

5 AVALIAÇÃO DA ARQUITETURA SAUI

Este Capítulo apresenta a avaliação das funcionalidades da Arquitetura SAUI, com base em cenários de uso nas áreas de agropecuária e saúde. Ainda, é destacada neste Capítulo a avaliação de usabilidade de uma das aplicações prototipadas com o emprego da Arquitetura SAUI.

A escolha destas áreas e dos correspondentes cenários de uso decorre de projetos que vem sendo desenvolvidos com a participação dos grupos de pesquisa GPPD/UFRGS (Grupo de Processamento Paralelo e Distribuído da Universidade Federal do Rio Grande do Sul) e LUPS/UFPel (*Laboratory of Ubiquitous and Parallel Systems* da Universidade Federal de Pelotas).

Segundo SATYANARAYANAN (2010), cenários de uso têm sido uma das formas de avaliação mais utilizadas pela comunidade científica para trabalhos na área de Computação Ubíqua. Particularmente, KNAPPMEYER et al. (2013) considera que a prototipação tem se mostrado uma escolha adequada para avaliação de arquiteturas, pois permite a identificação de não conformidades a partir da experiência dos usuários com aplicações que utilizam as mesmas.

Assim, na continuidade deste Capítulo, a Seção 5.1 apresenta os cenários de uso empregados na avaliação da Arquitetura SAUI, sendo caracterizados os protótipos desenvolvidos, as tecnologias empregadas e os testes realizados. Por sua vez, na Seção 5.2 é apresentado o experimento realizado para avaliação da usabilidade de uma aplicação prototipada com a Arquitetura SAUI, empregando o método TAM (*Technology Acceptance Model*).

5.1 Avaliação por Cenários de Uso

Esta seção descreve os cenários de uso empregados na avaliação da Arquitetura SAUI, relacionados às áreas de agropecuária e saúde. A avaliação em cenários de diferentes áreas tem por premissa que a Arquitetura SAUI não é concebida para um domínio de aplicação em particular, mas sim se propõe a atender diferentes domínios, sem a necessidade de alteração em seus componentes de software.

5.1.1 Tecnologias Empregadas

Na implementação dos componentes de software da Arquitetura SAUI foram utilizadas tecnologias escaláveis e robustas. A definição do hardware dos equipamentos empregados como Servidores de Contexto e de Borda considerou aspectos como baixo custo e reduzido consumo de energia.

A linguagem Python foi utilizada na implementação do código dos servidores

de Contexto e de Borda, incluindo os mecanismos de coleta, armazenamento e processamento do contexto, e atuação junto ao meio. Python é uma linguagem de alto nível, orientada a objetos, de tipagem dinâmica, interpretada e interativa.

O estilo arquitetural REST foi empregado para implementação da interoperabilidade entre os servidores. A linguagem de marcação JSON (*JavaScript Object Notation*) foi utilizada como estratégia de formatação dos dados para comunicação entre os componentes de processamento aplicados ao fluxo de processamento contextual. A linguagem JSON é voltada para a troca de informações estruturadas, permitindo a representação textual de dados numéricos, listas e vetores. O protocolo HTTP foi utilizado como mecanismo de transporte dos dados.

O armazenamento de informações contextuais emprega o gerenciador PostgreSQL para implementação das bases de dados relacionais. A ferramenta Protégé e a API Jena (JENA, 2015) foram utilizas para construção, manutenção e processamento do modelo ontológico.

O pacote de software WekA foi utilizado para criação dos componentes baseados em algoritmos de aprendizagem de máquina. Este software disponibiliza um conjunto de métodos que permitem a manipulação dos dados através de uma interface gráfica ou através de um conjunto de classes Java.

A prototipação dos Servidores de Borda foi realizada sobre o Sistema Operacional Raspbian (distribuição Linux), sendo usado como hardware o Raspberry Pi Modelo B+ (BROCK; BRUCE; CAMERON, 2013). O Raspberry Pi é um computador desenvolvido pelo Laboratório de Computação da Universidade de Cambridge que tem como principais características tamanho reduzido, baixo custo e baixo consumo energético, possuindo uma comunidade de usuários bastante ativa e que se baseia nos princípios de software livre. A utilização do Raspberry Pi como Servidor de Borda se mostrou interessante, pois ajusta-se bem ao cenário da IoT, tendo boa capacidade de processamento, o que possibilita que regras sejam processadas adequadamente, além da gerência e comunicação de diferentes protocolos e dispositivos de sensoriamento, através de um barramento que fica à disposição do usuário para prototipações.

Como Servidor de Contexto foi utilizado um hardware com processador Intel de dois núcleos, memória RAM de 4Gb e sistema operacional Linux Ubuntu Server. A escolha da distribuição Ubuntu foi decorrente da ampla utilização da mesma pelo grupo de pesquisa em seus equipamentos servidores, o que simplifica o seu suporte.

5.1.2 Cenário de Uso na Área de Agropecuária

O cenário de uso explorado nesta Seção é direcionado para a avaliação das funcionalidades da Arquitetura SAUI destinadas à aquisição dos dados contextuais, bem como ao processamento híbrido do contexto, visando à identificação de situações que são de interesse das aplicações.

Esta avaliação das funcionalidades da SAUI se valeu do projeto denominado plenUS (*plentiful Ubiquitous Systems*) que vem sendo desenvolvido por um consórcio de pesquisa envolvendo o GPPD/UFRGS, o LUPS/UFPel e a Embrapa Clima Temperado. O plenUS tem como premissa o emprego dos serviços de Ciência de Situação da Arquitetura SAUI e demais serviços do *middleware* EXEHDA com o objetivo de disponibilizar sistemas ubíquos que explorem relações proativas entre usuários, softwares e equipamentos, promovendo assim soluções computacionais que contribuam de forma sinérgica no atendimento das atividades de pesquisa da Embrapa Clima Temperado.

Para o projeto plenUS foi desenvolvida uma infraestrutura de hardware e software

na qual se diferenciam, de acordo com o nível de acesso ao *middleware* EXEHDA, três tipos de usuários: (i) Desenvolvedor: possui acesso total às funcionalidades do *middleware*, podendo realizar mudanças em sua arquitetura de software, seja pela reprogramação de módulos já existentes no *middleware* ou seja pela criação de novos módulos; (ii) Administrador: possui atribuições de gerência do *middleware*, configurando o mesmo para atender as demandas de um segmento usuário em particular; e (iii) Usuário Final: utiliza os serviços providos pelo *middleware* por meio de aplicações.

Neste sentido, os mecanismos concebidos pela Arquitetura SAUI tem como função facilitar os aspectos de gerência do Administrador, além de fornecer ao Usuário Final aplicações para utilização dos serviços do *middleware*.

Outro propósito da Arquitetura SAUI é o de atender as demandas do projeto em relação ao provimento de serviços para Ciência de Situação, permitindo um registro histórico dos estados contextuais em que se encontram os equipamentos dos laboratórios de Qualidade do Leite (Lableite) e de Análise de Sementes Oficial (LASO). Também, são atendidas outras necessidades de sensoriamento em ambientes administrativos como o do Núcleo de Tecnologia da Informação (NTI), e de uma sala de testes do projeto plenUS.

Um Servidor de Contexto e dois Servidores de Borda foram instalados para atender as demandas do projeto. Os Servidores de Borda foram posicionados na unidade Estação Experimental Terras Baixas (ETB), sendo distribuídos um em cada prédio específico onde a coleta de informações contextuais deve acontecer. Por sua vez, o Servidor de Contexto foi posicionado no Núcleo de Tecnologia da Informação, localizado na unidade Sede da Embrapa Clima Temperado. A escolha desta localização para o Servidor de Contexto considerou fatores, tais como: climatização adequada, sistema de fornecimento de energia ininterrupto (nobreak e gerador), bem como acesso facilitado para manutenção do equipamento.

As regras para operacionalização deste cenário são do tipo ECA (Evento-Condição-Ação) distribuídas entre os Servidores de Borda e Contexto, tendo como entrada as informações contextuais coletadas e como condição as faixas operacionais para cada contexto específico. Caso a informação contextual coletada não esteja dentro da faixa operacional, uma ação será executada.

A interação do usuário com o cenário é disponibilizada por meio de uma aplicação desenvolvida para navegadores Web, a qual contempla tarefas de gerenciamento dos dispositivos de sensoriamento. Além disso, a aplicação também permite a visualização do registro histórico das informações contextuais, tanto de forma gráfica como textual.

Outra característica avaliada foi a responsividade da aplicação Web desenvolvida, o que veio agregar maior funcionalidade ao seu uso, pois possibilitou que as configurações e testes dos dispositivos de sensoriamento pudessem ser realizadas no local de instalação através de smartphones ou tablets, devido ao seu layout se adaptar ao dispositivo computacional utilizado.

Uma caracterização dos laboratórios atendidos pelo projeto plenUS é realizada a seguir.

Laboratório de Qualidade do Leite

A Estação Experimental Terras Baixas (ETB) da Embrapa Clima Temperado abriga o SISPEL (Sistema de Pesquisa e Desenvolvimento em Pecuária Leiteira). Este sistema é constituído de instância física e operacional que inclui equipe de pesquisa e parceiros, rebanho experimental da raça Jersey, instalações de manejo e laboratórios. No SISPEL destaca-se o Lableite, credenciado pelo MAPA (Ministério da Agricultura, Pecuária e

Abastecimento), e que faz parte da Rede Brasileira de Laboratórios de Qualidade do Leite (RBQL).

O Lableite atua na pesquisa e prestação de serviços, atendendo a indústrias e cooperativas de laticínios, universidades e órgãos de pesquisa em leite, além de produtores particulares. Possui como objetivo realizar análises para avaliar a qualidade do leite, visando à adequação aos padrões previstos em regulamentação federal que exige, na avaliação da matéria prima, pelo menos duas amostras de leite de cada produtor passem por avaliações a cada mês.

Neste laboratório são avaliados o estado contextual dos equipamentos ou ambientes listados na Tabela 5.1:

Tabela 5.1: Equipamentos do Laboratório de Leite Avaliados pela Arquitetura SAUI

Equipamento ou Ambiente	**Objetivo**	**Sensor**	**Regime Operacional**	**Ação**
Geladeira de Amostras	Manter as amostras oriundas dos produtores rurais em condição adequada de temperatura até que as mesmas entrem em processo de análise	Sensor de temperatura 1-Wire encapsulado para emprego submerso em líquido (água)	de 0 à 7^{o}C	Envio de alerta para e-mails cadastrados
Geladeira de Insumos	Manter em condições adequadas de temperatura os insumos necessários as diferentes etapas do processo de análise da qualidade do leite	Sensor de temperatura 1-Wire encapsulado para uso não submerso (ambientes úmidos)	de 0 à 4^{o}C	Envio de alerta para e-mails cadastrados
Câmara Fria	Armazenar insumos no seu estado bruto, bem como outros produtos necessários a operação do Lableite	Sensor de temperatura 1-Wire encapsulado para uso não submerso (ambientes úmidos)	de 0 à 7^{o}C	Envio de alerta para e-mails cadastrados
Sala de Análises	Ambiente no qual os equipamentos que realizam a análise da qualidade do leite operam. Assim, com o intuito de minimizar a proliferação de fungos e/ou bactérias, a temperatura desta sala não deve ultrapassar determinado limite	Sensor de temperatura 1-Wire não encapsulado	de 15 à 25^{o}C	Envio de alerta para e-mails cadastrados

Laboratório de Análise de Sementes Oficial

O LASO realiza atividades de análise e pesquisa de diversas espécies de sementes, prestando serviços a produtores de sementes do país, bem como propicia a realização de aulas práticas para estudantes e treinamento para analistas de sementes de instituições oficiais e privadas.

A principal finalidade da análise de sementes é determinar a qualidade de um lote de sementes e, consequentemente, o seu valor para a semeadura. Os 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.

Para que os objetivos das análises sejam atingidos é necessário que os laboratórios possuam equipamentos adequados e sigam métodos e procedimentos uniformes. Com a implementação de um sistema de qualidade no LASO aumentaram as exigências nos controles dos equipamentos onde são realizadas as análises, visando a precisão e confiabilidade nos resultados. Com isso, registros diários de leituras de grandezas como temperatura dos germinadores, câmaras e salas são necessários.

Muitos laboratórios ainda utilizam planilhas em papel que são preenchidas manualmente várias vezes ao dia e para realização desta tarefa, funcionários precisam ocupar parte da sua jornada de trabalho diária. Alguns laboratórios possuem *dataloggers*, no entanto, embora os dados coletados possam ser armazenados, não há atuação automatizada em decorrência desses dados. Devido a isso, muitas das análises são perdidas por atraso na detecção de problemas nos germinadores e/ou outros equipamentos.

Assim, o ambiente do LASO com suas demandas ubíquas possui diversos sensores, bem como diferentes grandezas monitoradas. Na Tabela 5.2 são apresentadas algumas especificações referentes a coleta das informações contextuais dos equipamentos e salas avaliadas no LASO.

Infraestrutura de Software e Hardware

Neste cenário de uso é utilizado o modelo sintático do RHIC para representação dos dados contextuais, sendo empregado o gerenciador PostgreSQL para implementação das bases de dados.

O Servidor de Contexto foi instalado nas dependências do Núcleo de Tecnologia da Informação da Embrapa com o intuito de maximizar a sua disponibilidade, além do fato de poder contar com *nobreak* e um sistema gerador de energia elétrica.

Para as demandas de sensoriamento deste cenário de uso foi utilizada a tecnologia 1-Wire (MAXIM, 2016) como protocolo físico de acesso aos sensores. A tecnologia 1-Wire se destaca por ser um protocolo para transmissão dos dados em rede, baseado em dispositivos eletrônicos endereçáveis, robusto, e que não exige cabeamento especializado para sua operação. A leitura dos sensores é efetivada por meio de *drivers*, que são softwares específicos por tipo de sensor, responsável por traduzir o dado captado por meio do protocolo 1-Wire em uma informação contextual compatível com a grandeza mensurada.

Os *drivers* para a captação de dados dos sensores são acionados mediante requisição de leitura pelo Servidor de Borda, que os armazena temporariamente em virtude de uma eventual falha de comunicação com o Servidor de Contexto. A publicação dos dados contextuais captados no Servidor de Contexto acontece de acordo com o tempo de publicação estipulado nos parâmetros operacionais dos sensores no Servidor de Borda.

Para atender as funcionalidades previstas para este cenário de uso, foi necessária a criação de alguns dispositivos de sensoriamento, bem como hardwares específicos e o respectivo firmware para operarem na rede de sensores 1-Wire. Dentre os dispositivos desenvolvidos destacam-se: sensor de temperatura com diferentes tipos de encapsulamento, sensor de umidade, sensor de presença de luz, interface serial 1-Wire e HUB para conexão de sensores 1-wire.

Aquisição dos Dados de Contexto no plenUS

Nesta seção é apresentada a avaliação das funcionalidades da Arquitetura SAUI destinadas ao gerenciamento dos recursos direcionados ao sensoriamento e à atuação sobre o meio e à visualização dos dados de contexto coletados. Esta avaliação envolve os laboratórios atendidos pelo projeto plenUS.

Atendendo a premissa que as informações devem estar acessíveis independentemente do dispositivo computacional, as tecnologias utilizadas pelas aplicações no âmbito da IoT devem possuir recursos que permitam se ajustarem às mudanças ocorridas no ambiente de execução.

Tabela 5.2: Coleta de Informações Contextuais no LASO

Equipamento/Sala	Grandeza	Faixa de Operação
Sala das Balanças	Temperatura	10 à 30 C
	Umidade	50% à 98%
Câmara de Conserv. Amostras	Temperatura	10 à 25 C
	Umidade	50% à 98%
Sala dos Germinadores	Temperatura	20 à 27 C
Germinador 01	Temperatura	18 à 22 C
	Presença de luz	00:00:00 às 11:59:59 - sem luz 12:00:00 às 23:59:59 - com luz
Germinador 02	Temperatura	08 à 12 C
Germinador 03	Temperatura	18 à 22 C
	Presença de luz	00:00:00 às 11:59:59 - sem luz 12:00:00 às 23:59:59 - com luz
Germinador 04	Temperatura	23 à 27 C
Germinador 05	Temperatura	23 à 27 C
Germinador 06	Temperatura	23 à 27 C
Germinador 07	Temperatura	18 à 27 C
Germinador 08	Temperatura	12:30:00 às 20:00:00 - 28 à 32 C 20:00:01 às 12:29:59 - 18 à 22 C
	Presença de luz	12:00:00 às 20:00:00 - com luz 20:00:01 às 11:59:59 - sem luz
Germinador 09	Temperatura	12:30:00 às 20:00:00 - 28 à 32 C 20:00:01 às 12:29:59 - 18 à 22 C
	Presença de luz	12:00:00 às 20:00:00 - com luz 20:00:01 às 11:59:59 - sem luz

Considerando isto, no desenvolvimento da aplicação Web foram empregadas tecnologias responsivas no seu *layout*. O *layout* é considerado responsivo, quando ele se adapta ao tamanho da tela do dispositivo, sem prejudicar suas funcionalidades. Portanto, a aplicação pode ser acessada da mesma forma, tanto por *desktops* e *notebooks* como por *tablets* e *smartphones*.

A aplicação desenvolvida contempla dois modos de operação, um para a gerência de recursos de sensoriamento e/ou atuação, com suporte ao acesso a diferentes níveis de gerenciamento de acordo com o perfil do usuário, e outra para visualização de dados históricos das informações contextuais.

Aplicação de Gerenciamento

A aplicação para gerência dos recursos de sensoriamento e atuação desenvolvida para o Servidor de Contexto é responsável por realizar as operações básicas utilizadas em banco de dados relacionais, ou seja, está capacitada a cadastrar, ler, atualizar e remover dados na camada de armazenamento do Repositório Híbrido de Informações Contextuais.

A aplicação foi desenvolvida utilizando princípios arquiteturais REST, possibilitando que qualquer operação possa ser realizada por uma URI específica. Ao acessar cada URI, parâmetros devem ser passados viabilizando assim que diversas funções requisitadas pelo Servidor de Contexto possam ser efetuadas.

O acesso à aplicação de gerenciamento somente acontece por meio de login, sendo que cada usuário poderá ter um nível de acesso diferenciado aos menus de acordo com a sua permissão.

Além de viabilizar operações no banco de dados, a aplicação é responsável por criar uma interface para que o usuário possa gerenciar seus recursos. Considerando estes aspectos, algumas funcionalidades foram criadas, as quais estão dispostas em um menu (vide Figura 5.1) e encontram-se descritas a seguir.

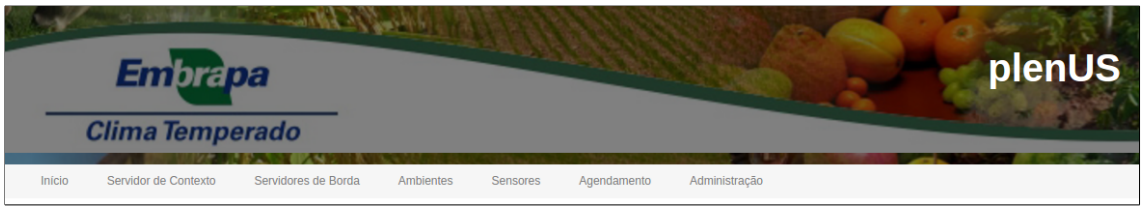


Figura 5.1: Menu da Aplicação de Gerenciamento

- **Servidor de Contexto**: responsável por editar informações descritivas a respeito do Servidor de Contexto, como nome e localização.
  - **Contexto de Interesse**: responsável por promover todas as operações de gerência sobre os contextos de interesse. Estes Contextos de Interesse relacionam um nome a um determinado grupo de sensores que são de interesse de uma determinada aplicação.
  - **Regras**: permite que sejam carregadas regras em arquivos Python, vinculadas a sensores e/ou contextos de interesse para serem utilizadas no processamento das informações contextuais.
- **Servidores de Borda**: tem como função efetuar a criação, alteração e remoção de cadastros dos Servidores de Borda (vide Figura 5.2).
- **Ambientes**: por meio deste recurso é possível efetuar operações de gerência relacionadas ao local onde cada sensor está posicionado fisicamente, por exemplo, o ambiente pode ser o equipamento ou sala que o sensor está inserido. Este recurso serve como um parâmetro para a função de Agendamento.
- **Sensores**: responsável por configurar características dos sensores, como as faixas operacionais previstas para um determinado horário, que caracterizam as condições para as regras serem executadas, bem como a vinculação de cada sensor ao servidor de borda ao qual está atrelado (vide Figura 5.3);

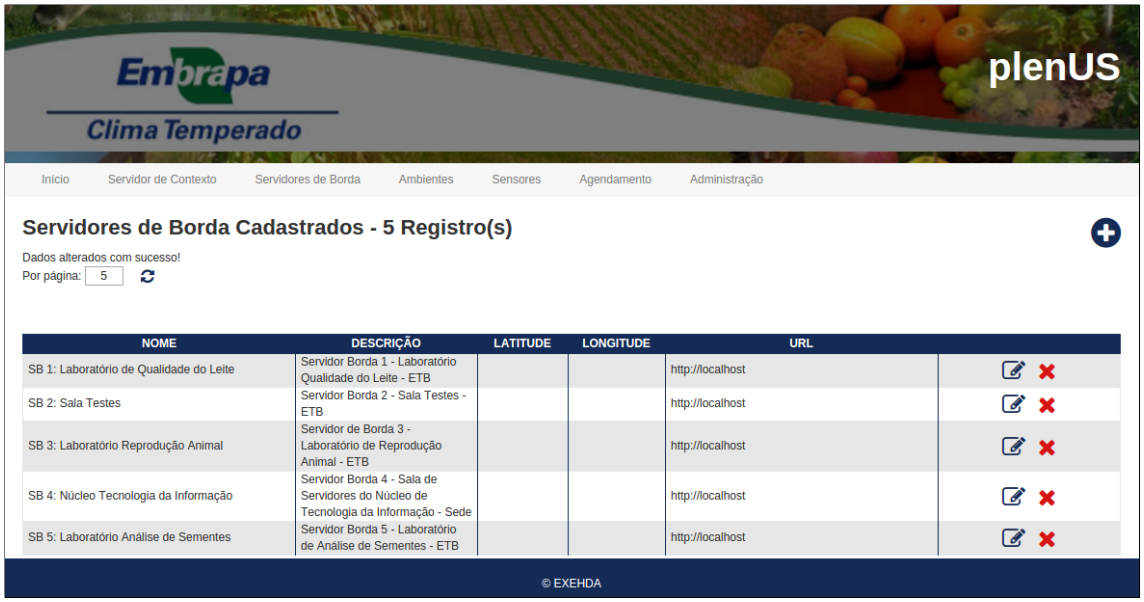


Figura 5.2: Gerência dos Servidores de Borda

Figura 5.3: Gerência dos Sensores

– **Tipos de Sensores**: com este recurso é possível manipular os tipos de sensores, envolvendo desde o nome até a unidade de referência (vide Figura 5.4);

– **Publicações Realizadas**: permite visualizar, cadastrar e remover manualmente publicações referentes a um sensor. A função de cadastro de publicações de forma manual foi criada com o objetivo de avaliar algumas funcionalidades da aplicação.

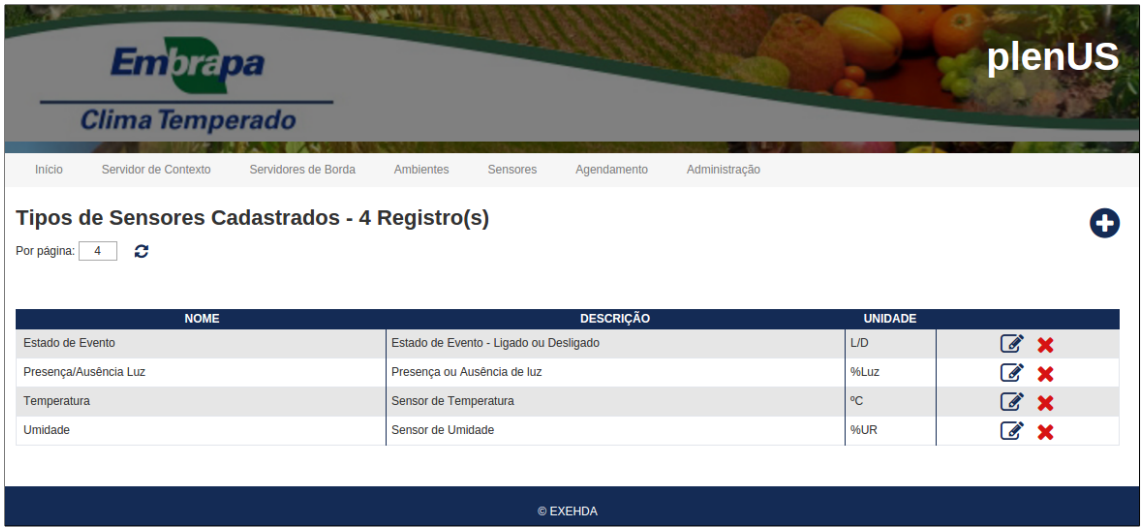


Figura 5.4: Gerência dos Tipos de Sensores

- **Agendamento**: esta funcionalidade é direcionada ao gerenciamento do uso de sensores em determinados ambientes controlados. Este recurso permite efetuar uma reserva de um ambiente para uma determinada data, relacionada com um usuário do sistema. Assim, se um usuário tentar efetuar o agendamento de um determinado ambiente que já foi agendado, o sistema o avisará que já existe um agendamento para este ambiente (vide Figura 5.5), evitando que haja conflitos em uma eventual tomada de decisão por meio de regras distintas vinculadas aquele dispositivo. Neste menu também é disponibilizado a função Visualizar, que possibilita a visualização dos agendamentos registrados e a função Relatórios que disponibiliza a geração de relatórios dos agendamentos relacionados a um determinado ambiente.

- **Administração**: por este recurso da aplicação, pode-se realizar gerência de menus, usuários, perfis de usuários e fabricantes de dispositivos.

Aplicação de Visualização

A aplicação no seu modo de visualização possibilita a seleção de um ou mais sensores, dentro de um contexto de interesse, para exibição do registro histórico das suas informações contextuais. Estas informações podem ser apresentadas na forma de um relatório textual (vide Figura 5.6) ou através de modo gráfico (vide Figura 5.7).

Através dessa aplicação os usuários podem ter acesso à visualização das variações dos valores das grandezas medidas ocorridas nos equipamentos e/ou salas durante os períodos de análise dos experimentos.

O relatório gráfico desenvolvido permite visualizar simultaneamente as curvas de variação dos valores de vários sensores utilizados no plenUS. A seleção dos sensores a serem visualizados é feita a partir de um menu com suporte a múltipla seleção. Também é disponibilizado um recurso de inspeção que permite a comparação dos valores em um determinado instante do tempo. A janela de tempo dos dados que estão sendo visualizados pode ser definida pelo usuário através da mesma interface gráfica que exibe os valores sensoriados.

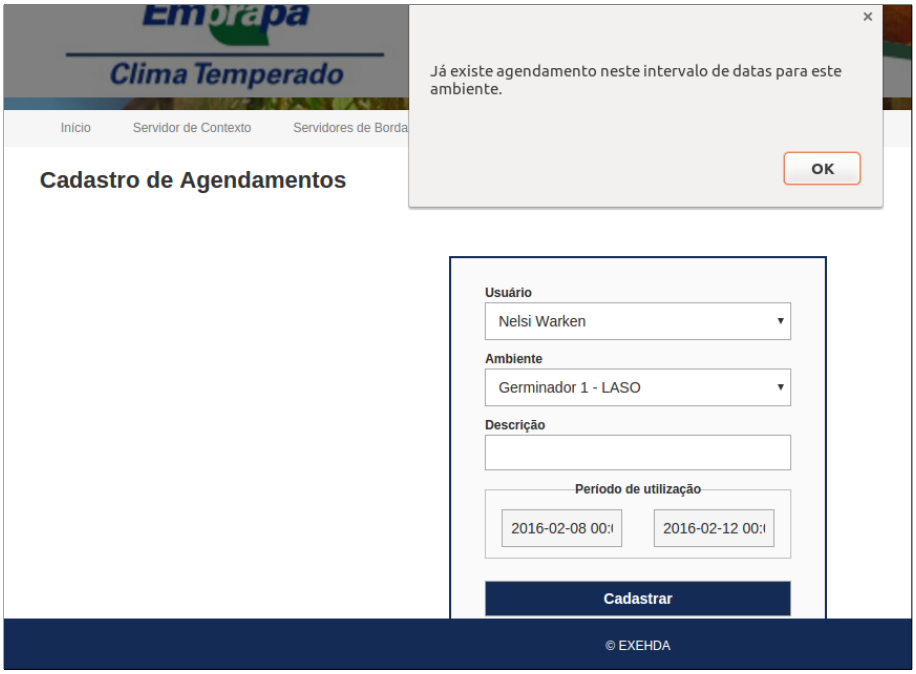


Figura 5.5: Notificação de Registro de Agendamento

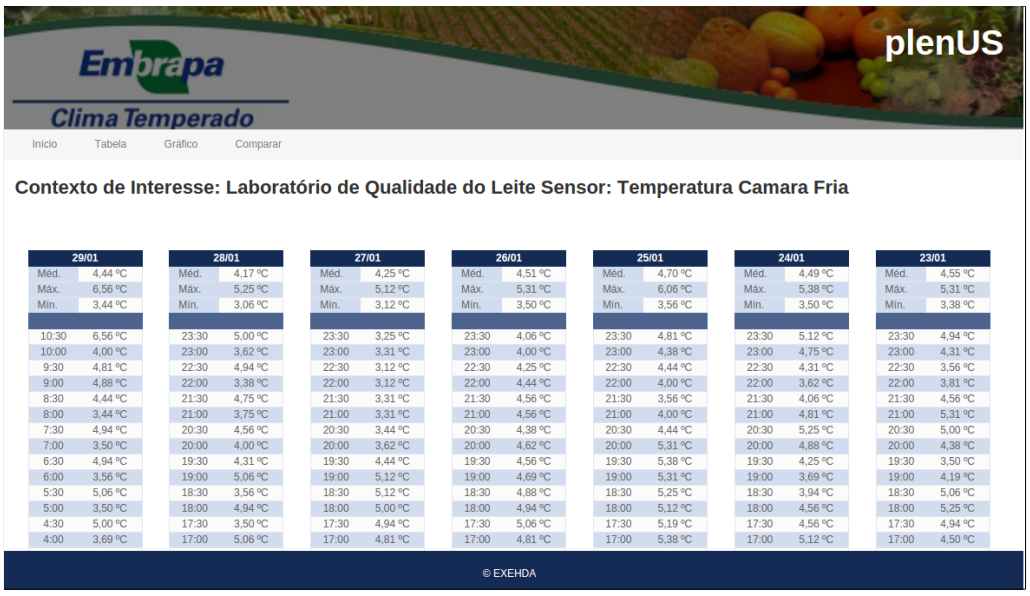


Figura 5.6: Visualização Textual das Informações Contextuais

Segurança da Infraestrutura Computacional do plenUS

Considerando os experimentos desenvolvidos nos laboratórios atendidos pelo projeto plenUS, os servidores de contexto devem funcionar em regime de 24 horas, 7 dias por semana, para manter o processamento dos dados contextuais gerados por estes experimentos. Assim, neste cenário são avaliadas as funcionalidades da Arquitetura SAUI direcionadas à modelagem e armazenamento dos dados contextuais, ao gerenciamento de regras, bem como ao processamento híbrido do contexto. Estas funcionalidades são empregadas neste cenário com o intuito de notificar o administrador dos servidores de rede do projeto plenUS a ocorrência de uma situação que caracterize uma tentativa de

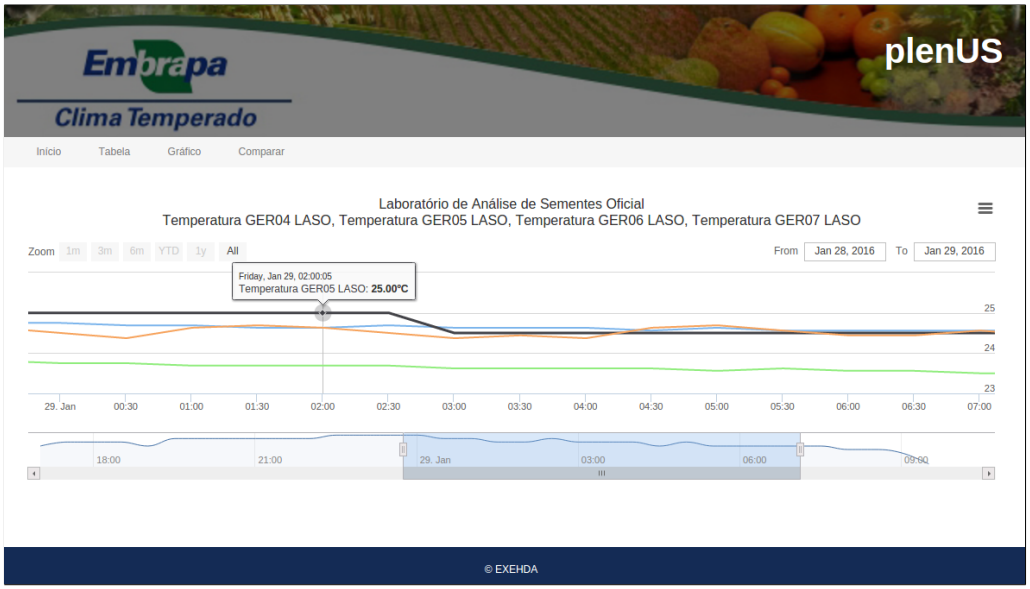


Figura 5.7: Visualização Gráfica das Informações Contextuais

ataque.

Instanciação do RHIC

A modelagem semântica das informações de uma infraestrutura de servidores de rede através de ontologias tem sido uma das formas mais exploradas para identificação de ataques com base no tráfego da rede (ABDOLI; KAHANI, 2009) (ISAZA et al., 2010) (KHAIRKAR, 2013).

Uma das dificuldades encontradas diz respeito ao número significativo de falsos positivos e falsos negativos, os quais acontecem quando uma conexão "normal" é identificada como uma categoria de "ataque" e quando um "ataque" é classificado como uma "conexão normal".

Assim, na perspectiva da Arquitetura SAUI, considera-se que o uso de forma combinada de informações semânticas e sintáticas pode melhorar o processo de identificação de ataques. Dentre as informações sintáticas, destaca-se as que dizem respeito ao status do dispositivo que está sendo monitorado, tais como: quantidade de memória utilizada, utilização do processador, entre outras.

Desse modo, para armazenamento dos dados de contexto tratados neste cenário de uso, o Repositório Híbrido de Informações de Contexto (RHIC) foi instanciado com dois diferentes modelos de dados, caracterizando a abordagem híbrida proposta para representação do contexto na Arquitetura SAUI: (i) modelo de ontologia, representado por triplas OWL (*Web Ontology Language*); e (ii) modelo entidade-relacionamento, representado através de banco de dados relacional.

No que diz respeito a ontologias, dentre as encontradas na literatura, optou-se por empregar neste cenário de uso a ontologia definida por UNDERCOFFER; JOSHI; PINKSTON (2003), a qual faz o mapeamento das conexões do tráfego da rede e identifica possíveis ataques a mesma.

O armazenamento das instâncias da ontologia que detecta tentativas de ataques aos servidores de rede é feito no modelo de triplas do RHIC. Por sua vez, as informações referentes ao status dos dispositivos ocorre no modelo relacional. Além disso, com a funcionalidade de integração dos modelos sintático e semântico do Gerenciador de

Regras do Módulo de Processamento (vide Seção 4.5.1) torna-se possível a combinação destas informações, possibilitando melhorar a identificação do ataque a um dispositivo computacional.

Manipulação dos Dados de Contexto no RHIC

O Gerenciador de Modelos é responsável pela disponibilização das funcionalidades que permitem o acesso aos dados contextuais armazenados no RHIC, correspondendo neste cenário de uso ao modelo de ontologia (triplas OWL), bem como ao modelo de entidade-relacionamento (banco de dados relacional).

As consultas ao repositório de triplas são realizadas através da linguagem de consulta SPARQL (*SPARQL Protocol and RDF Query Language*) (SPARQL, 2013). Para tanto, as consultas podem ser executadas por uma interface Web, na qual podem ser visualizadas as informações presentes no repositório, ou através de uma API que permite às aplicações realizarem a manipulação dos dados de contexto do modelo de triplas do RHIC de forma integrada a seu código fonte.

Na implementação das funcionalidades do componente Gerenciador de Modelos direcionadas ao modelo semântico é utilizada a API Jena, a qual permite a manipulação em memória de ontologias, possibilitando a realização de consultas SPARQL e de inferências, bem como a execução de tarefas de edição e inserção de novas informações nas triplas OWL armazenadas no RHIC. Nesse sentido, o processamento semântico das informações de contexto com o uso de padrões da Web Semântica como ontologias, linguagens para consulta, raciocinadores e APIs para manipulação de modelos ontológicos é descrito no Apêndice B.

Testes das Funcionalidades da SAUI

Na avaliação das funcionalidades da Arquitetura SAUI para identificação de situações de ataque aos servidores de rede do plenUS é empregada a base de dados "KDD Cup 99 Data" como fonte de dados de contexto relacionados ao tráfego da rede. Esta base é considerada uma das principais utilizadas na avaliação de mecanismos para detecção de tentativas de ataques a servidores de rede (ELEKAR; WAGHMARE; PRIYADARSHI, 2015). O conjunto de dados utilizado contém um número significativo de conexões já classificadas em cinco categorias, sendo uma delas a categoria de conexão "Normal" e quatro categorias de ataques (LEE; STOLFO; MOK, 1999).

Além disso, neste teste foi considerada a percentagem de utilização do processador, sendo a mesma categorizada em: Alta, Média, e Baixa. Com isto foi possível testar combinações com os atributos de identificações de ataques e a categoria de utilização do processador em um determinado momento.

Com o intuito de processar os contextos adquiridos foi adicionada ao Repositório de Regras uma regra para integração dos modelos sintático e semântico a ser executada sobre o RHIC. Inicialmente, é realizado um *parser* na regra para identificar as informações que devem ser buscadas nos respectivos modelos, as quais são representadas pelas *tags* # seguido de um número, sendo o símbolo # usado para identificar as consultas auxiliares que devem ser realizadas antes da execução da regra de integração e o número de identificação da consulta.

Com os números das consultas auxiliares, estas são buscadas no Repositório de Regras, após é realizada a aquisição das informações necessárias para execução da regra com o intuito de identificar uma situação. O fluxo de execução da regra empregada para a identificação de ataques a um servidor de rede é apresentado na Figura 5.8. Pode-se

observar que a regra de integração possui duas *tags* de marcação, onde a *tag* #1 é referente a uma busca no modelo relacional e a #2 corresponde a uma busca no modelo ontológico.

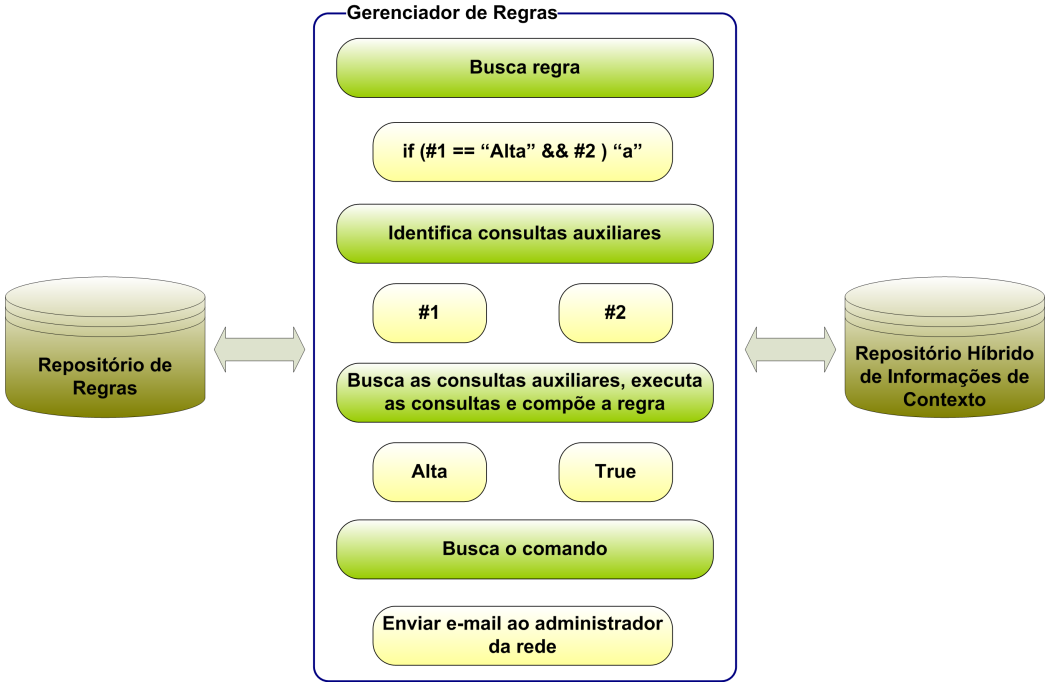


Figura 5.8: Funcionamento da Regra para Detecção de Ataques

Nesse cenário, a regra de integração executada é *if (#1 == "Alta" && #2) "a"*, onde a *tag* de marcação #1 é referente a consulta SQL apresentada na Figura 5.9, a qual busca a categoria que foi classificada a utilização do processador do dispositivo desejado no modelo relacional do RHIC.

```
Select lastvalue, date  from items_hosts
where name = 'CPU [Utilizada]' and hostid = 'x'
```

Figura 5.9: Consulta SQL - Busca pela Categoria de Utilização do Processador

A *tag* #2 é referente a consulta apresentada na Figura 5.10, onde pode ser visualizada a consulta que será realizada no RHIC, sobre o modelo ontológico, representado por triplas. A consulta apresentada na Figura 5.10 é responsável por buscar neste modelo a identificação de um ataque classificado como DoS (*Denial of Service*) para o dispositivo, este tipo de categoria de ataque costuma gerar uma quantidade significativa de tentativas de aberturas de conexão com um servidor com o intuito de sobrecarregá-lo, gerando assim um aumento na utilização do processador do mesmo.

Com o intuito de melhorar a identificação da situação, a consulta SPARQL realizada leva em consideração uma janela de tempo, já que a ontologia pode vir a identificar a situação de ataque antes de aumentar a utilização do processador, da mesma forma que o inverso é possível. Também, como condição na regra de integração dos modelos é utilizado o método que verifica se foi retornado algum resultado para consulta SPARQL realizada.

```
SELECT ?individuo ?host ?data
  WHERE {
    ?individuo  rdf:type  trafego:DoS.
    ?individuo  trafego:host  ?host.
    ?host  trafego:id  “x”^^xsd:string.
    ?individuo  trafego:data  ?data.
    FILTER (?data >= data_inic^^xsd:dateTime || ?data <= data_final^^xsd:dateTime).
    }
```

Figura 5.10: Consulta SPARQL - Busca por Ataque DoS a um Dispositivo

Após as *tags* de marcação serem substituídas pelos valores resultantes de cada consulta, a regra de integração dos modelos é executada, a identificação da situação de interesse pode ser alcançada, resultando na execução do comando “a”, o qual é referente ao envio de um e-mail para o administrador de rede do projeto plenUS, informando que foi identificada uma tentativa de ataque ao respectivo host.

5.1.3 Cenário de Uso na Área de Saúde

Os projetos na área de saúde que vem sendo desenvolvidos com a participação dos grupos de pesquisa GPPD/UFRGS e LUPS/UFPel contemplam o processo de identificação de situações relacionadas com: (i) avaliação de metas terapêuticas para pacientes de UTIs hospitalares (projeto EXEHDA-TG - *Therapeutic Goals*) (LOPES et al., 2016); e (ii) detecção de risco de saúde para pacientes em reabilitação cardíaca (projeto EXEHDA-IS - *IoT Situations*) (LOPES et al., 2016).

Assim, neste cenário de uso são utilizados os serviços de Ciência de Situação da Arquitetura SAUI, bem como os demais serviços do *middleware* EXEHDA, para prover suporte à execução de aplicações, na perspectiva da infraestrutura provida pela IoT. O cenário desenvolvido tem como objetivo avaliar as funcionalidades da SAUI destinadas à coleta, armazenamento e processamento dos dados contextuais, bem como ao tratamento híbrido do contexto com o emprego de técnicas baseadas em especificação de regras e aprendizagem, visando à identificação de situações. Para tanto, são empregados os diferentes módulos que constituem os Servidores de Contexto e de Borda, especialmente os componentes de software que constituem o Módulo de Processamento do Servidor de Contexto (vide Seção 4.5).

Identificação de Situações: Metas Terapêuticas para Pacientes de UTIs

O acompanhamento dos sinais vitais dos pacientes constitui-se em uma das atividades mais importantes em uma UTI (Unidade de Tratamento Intensivo) de um hospital (ANTONY; ANJU; MATHEW, 2013). Desta forma, os mecanismos para Ciência de Situação da Arquitetura SAUI são empregados em um cenário de uso direcionado a UTIs hospitalares.

O objetivo central é proporcionar suporte computacional ao processo de medicação dentro do ambiente hospitalar. Para isso, os efeitos dos medicamentos são verificados através do acompanhamento da evolução dos sinais vitais, permitindo que o médico possa confirmar se está alcançando o resultado desejado com a administração de medicamentos ao paciente. Na perspectiva deste trabalho, esse resultado desejado pelos médicos nos

parâmetros vitais é chamado de Meta Terapêutica. Uma Meta Terapêutica pode ser, por exemplo, elevar os batimentos cardíacos ou estabilizar a pressão arterial do paciente em determinados patamares, considerando um intervalo de tempo específico.

O cenário de uso contempla as tarefas referentes ao sensoriamento e à coleta e armazenamento de dados contextuais, bem como seu processamento e a decorrente notificação quando necessário dependendo da situação dos pacientes com relação à meta terapêutica prevista. Com isso, são utilizados principalmente os seguintes módulos da Arquitetura SAUI: (i) Sensoriamento e Publicação do Servidor de Borda; e (ii) Aquisição, Processamento, Notificação, Comunicação e Atuação do Servidor de Contexto.

As funcionalidades previstas para Arquitetura SAUI têm como premissa reduzir o esforço necessário na coleta e acompanhamento dos sinais vitais, através da automatização do processo de aquisição e processamento dos dados de contexto, bem como a notificação da ocorrência de situações relacionadas aos pacientes.

Na Figura 5.11 é apresentada uma visão geral do escopo deste cenário de uso, no qual estão contemplados os seguintes aspectos:

- definição da meta terapêutica pelos médicos;
- interface para comunicação de dados com os monitores de sinais vitais, possibilitando a aquisição dos parâmetros fisiológicos dos pacientes e o processamento pelos componentes de software da Arquitetura SAUI;
- interface para comunicação de dados com as bombas de infusão para acompanhamento do processo de administração de medicamentos;
- avaliação da Meta Terapêutica através dos mecanismos de raciocínio da Arquitetura SAUI;
- envio, de forma ubíqua, dos alertas referentes ao processamento da meta terapêutica e dos dados das bombas de infusão para os dispositivos computacionais dos profissionais de saúde.

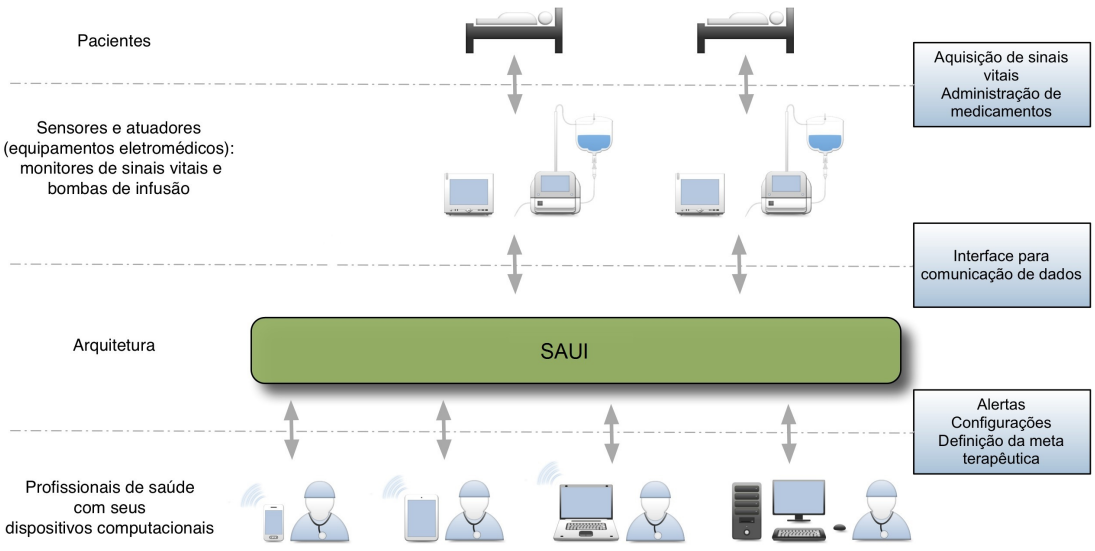


Figura 5.11: Visão Geral do Cenário de Metas Terapêuticas

Descrição do Ambiente

No cenário atual da maioria das UTIs, os sinais vitais são observados em equipamentos eletromédicos, como monitores de sinais vitais, sendo anotados manualmente e posteriormente transcritos para o prontuário do paciente. Com isso, torna-se necessário que os enfermeiros ou seus auxiliares percorram todos os leitos para coletar esses dados e os anotem em planilhas. Posteriormente, estes dados são avaliados pelos médicos, para verificar se os efeitos dos medicamentos administrados estão tendo os resultados esperados nos sinais vitais dos pacientes.

Todo este processo manual consome uma parte significativa da rotina de trabalho dos profissionais envolvidos. Além disso, como se trata de uma atividade manual, podem ocorrer erros de transcrição dos valores observados, o que pode resultar em uma interpretação equivocada do estado do paciente.

Considerando que no ambiente hospitalar os profissionais de saúde podem estar envolvidos em diversas outras atividades, tais como: administrar medicamentos, gerar gráficos e anotações de progresso no tratamento de saúde dos pacientes, solicitar exames e verificar seus resultados; a observação e a coleta dos sinais vitais podem não ser realizadas de forma regular ou ocorrerem em intervalos de tempo inadequados. Em função disto, anormalidades no estado clínico dos pacientes poderão não ser detectadas imediatamente, sendo identificadas pela equipe somente em momento posterior, quando a situação do paciente poderá ter se agravado.

Assim, com o emprego das funcionalidades da Arquitetura SAUI para processamento das informações de contexto e detecção de situações, os profissionais de saúde podem se concentrar em sua função primária, que é prover cuidados e atenção ao paciente.

Técnica Adotada para Identificação de Situações

A detecção de situações associadas a pessoas pode ser obtida pela identificação da relação entre os eventos associados, bem como pela sua duração. Neste sentido, pode ser utilizada a Lógica Temporal e Espacial de Allen, proposta originalmente em ALLEN; FERGUSON (1994), na avaliação situacional das metas terapêuticas.

A Lógica Temporal pode ser usada para descrever, limitar e compreender as sequências temporais entre dois eventos. Essa lógica usa intervalos para descrever eventos específicos, e as relações entre os intervalos para descrever as interdependências entre os eventos. Por exemplo, considerando os seguintes eventos: a porta do quarto é aberta (I1) antes da pessoa entrar no quarto (I2) e o sensor é acionado (I3) devido à presença da pessoa no quarto. Portanto, a composição destes eventos sugere que a porta tenha sido aberta antes do sensor ser disparado, o que é inferido desta forma: I1 $<$ I2 e I2 di I3, portanto I1 $<$ I3, onde “di” indica a relação temporal “duração”.

AUGUSTO et al. (2008) introduz na proposta de Allen os operadores ANDlater e ANDsim para regras ECA, através dos quais o conhecimento temporal de eventos pode ser representado. Com o uso destes operadores é possível tratar a ordem temporal de ocorrência de duas condições.

No exemplo apresentado a seguir, o comando X será executado caso a condição ’a’ seja verdadeira e posteriormente a condição ’b’ seja verdadeira, indicando que o evento ’a’ ocorreu antes do ’b’:

IF (a ANDlater b)
THEN X

No próximo exemplo, o comando Y será executado caso a condição ’c’ for verdadeira

e, simultaneamente, a condição 'd' for verdadeira, indicando que o evento 'c' ocorreu ao mesmo tempo que 'd':

IF (c ANDsim d)
THEN Y

A Lógica Temporal também pode ser usada para processamento espacial, interpretando intervalos como objetos unidimensionais e pontos temporais como locais no espaço, respectivamente.

A partir da Lógica Temporal, várias extensões multidimensionais têm sido sugeridas, as quais usam tuplas formadas por intervalos e que representam projeções de regiões sobre os eixos de um sistema de referência cartesiano. Estas representações capturam os aspectos topológicos e orientacionais do espaço.

Considera-se que os aspectos topológicos do espaço são os mais relevantes para a modelagem do comportamento humano em um ambiente. Por exemplo, para determinar qual a atividade que uma determinada pessoa está envolvida, é importante saber em que sala ela está (embora não seja necessário saber a posição exata da pessoa naquela sala). Ou, para determinar os possíveis movimentos de uma pessoa, é útil saber quais os quartos que estão conectados a outros quartos. Para fazer o processamento considerando os aspectos topológicos do espaço em um ambiente, pode-se utilizar cálculos baseados em regiões e conexões (*Region Connection Calculus* - RCC) (RANDELL; CUI; COHN, 1992).

O emprego da Lógica Temporal e Espacial para identificação de situações ganha dimensão em sistemas dinâmicos, cuja composição e organização são variáveis, como os ambientes hospitalares (GUESGEN; MARSLAND, 2010). Considerando isto, esta foi a técnica adotada para avaliar se a situação do paciente está atendendo a Meta Terapêutica especificada pelo médico.

Nesta perspectiva, a detecção de situações de interesse é realizada através dos componentes de software do Módulo de Processamento da Arquitetura SAUI, considerando a ocorrência de eventos significativos e os contextos em que os mesmos ocorrem, bem como o emprego de regras do tipo ECA (Evento-Condição-Ação) como uma forma de reagir às informações produzidas pelos sensores. As regras ECA têm uma sintaxe com o seguinte formato:

```
ON <evento>, IF <condição>, DO <ação>
```

Isto significa que toda vez que a ocorrência do evento descrito na cláusula `ON` for detectada, e se a condição presente na cláusula IF (geralmente impondo restrições sobre os diferentes aspectos dos eventos considerados na cláusula `ON`) é verdadeira, a ação especificada na cláusula `DO` é executada pelo sistema. Quando a cláusula `ON` for satisfeita a regra é "acionada" (*triggered*) e se, também a cláusula `IF` for satisfeita, então a regra é "disparada" (*fired*).

Especificação da Meta Terapêutica

Para todo tratamento medicamentoso, especificado por um médico, os parâmetros da correspondente Meta Terapêutica deverão ser informados à Arquitetura SAUI. Considerando esta parametrização, a evolução dos sinais vitais é avaliada pela arquitetura.

Os parâmetros que podem ser especificados para a Meta Terapêutica pelo médico são:

- **Medicamento**: define o medicamento e a dosagem que será administrada ao paciente.

- **Sinal vital**: determina qual será o parâmetro vital (por exemplo: frequência cardíaca, pressão arterial, oximetria) usado para monitorar o resultado do tratamento medicamentoso.
- **Limite inicial inferior (*LIi*)**: valor mínimo aceitável para o sinal vital no início do acompanhamento da Meta Terapêutica.
- **Limite inicial superior (*LIs*)**: valor máximo aceitável para o sinal vital no início do acompanhamento da Meta Terapêutica.
- **Limite final inferior (*LFi*)**: valor mínimo aceitável para o sinal vital no final do acompanhamento da Meta Terapêutica.
- **Limite final superior (*LFs*)**: valor máximo aceitável para o sinal vital no final do acompanhamento da Meta Terapêutica.
- **Limite da taxa de variação**: especifica a taxa máxima de variação aceitável para o sinal vital. Será gerado um alerta caso o valor aumente ou diminua em uma taxa superior a especificada.
- **Intervalo de medições**: intervalo de tempo entre as medições dos valores coletados e aplicação das regras da arquitetura.
- **Tempo de acompanhamento**: especifica o tempo em que os sinais vitais serão avaliados pela arquitetura.
- **Tempo de inibição dos alertas**: especifica o intervalo de tempo inicial em que a arquitetura inibirá o envio de alertas aos profissionais de saúde. Este parâmetro é usado para evitar que alertas desnecessários sejam enviados aos médicos no início do tratamento do paciente, quando os sinais vitais ainda não estabilizaram em função da medicação dada.

Através da especificação pelo médico dos limites iniciais e finais, a arquitetura determina uma região considerada normal para variação dos sinais vitais do paciente em função da medicação (vide Figura 5.12). Os limites inferiores (LIi e LFi) determinam os valores mínimos aceitáveis para o sinal vital durante o tratamento. Da mesma forma, os limites superiores (LIs e LFs) determinam os valores máximos aceitáveis para o sinal vital. Os valores intermediários aceitáveis (área sombreada do gráfico) são calculados através da interpolação linear dos limites iniciais e finais.

Os dados clínicos que caracterizam a situação do paciente são comparados com a Meta Terapêutica considerando a Lógica Temporal e Espacial. A premissa contemplada é que, após decorrido algum tempo do início da aplicação da terapia medicamentosa, os sinais vitais do paciente se mantenham contidos no intervalo da Meta Terapêutica (Figura 5.13 - caso 6: `I1 f I2`); para esta analogia, considera-se `I1` como sendo o intervalo de tempo em que a Meta Terapêutica está sendo atingida e `I2` o intervalo de tempo em que o paciente está sendo medicado. Idealmente os sinais vitais, já no início da terapia medicamentosa, irão se posicionar dentro da Meta Terapêutica (Figura 5.13 - caso 7: `I1 = I2`). Por fim, existe a possibilidade de concluído o período da terapia medicamentosa o paciente se manter dentro da Meta Terapêutica, mesmo sem estar recebendo medicamento (Figura 5.13 - caso 3: `I1 o I2`). Este último caso caracteriza uma recuperação do paciente.

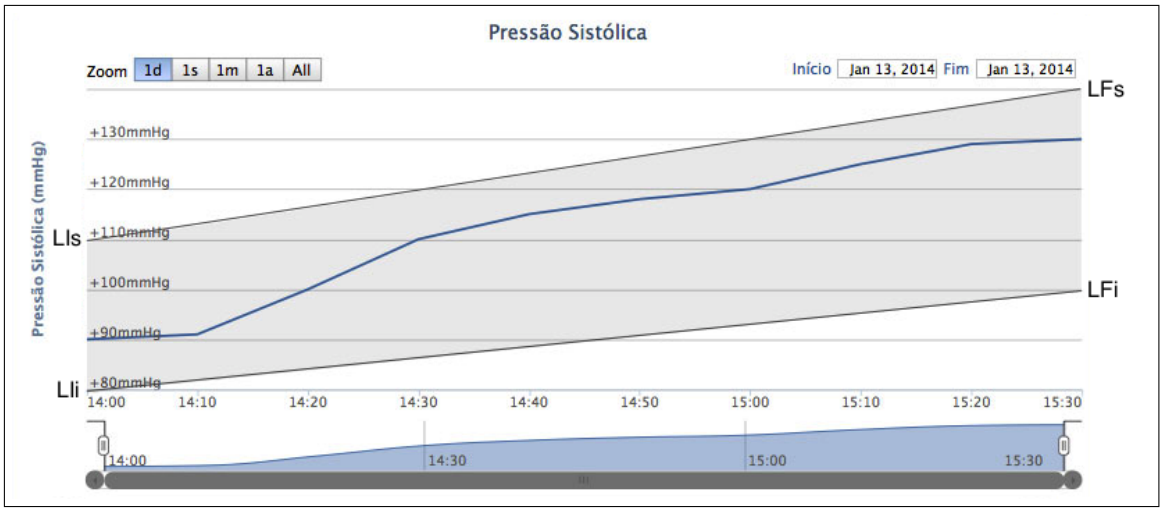


Figura 5.12: Exemplo de Meta Terapêutica

Quando os sinais vitais do paciente não se mantiverem no intervalo da Meta Terapêutica durante o período de medicação (Figura 5.13 - caso 1: `I1 < I2`, caso 2: `I1 m I2`, caso 4: `I1 s I2` e caso 5: `I1 d I2`), serão disparado os correspondentes procedimentos de alerta.

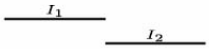
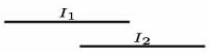
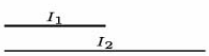
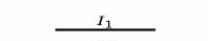
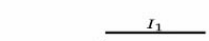
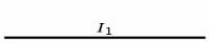
Relação	Ilustração	Interpretação
$I_1 < I_2$ $I_2 > I_1$		I1 antes de I2 I2 depois de I1
$I_1 \mathrm{m} I_2$ $I_2 \mathrm{mi} I_1$		I1 encontra I2 I2 é encontrado por I1
$I_1 \mathrm{o} I_2$ $I_2 \mathrm{oi} I_1$		I1 sobrepõe I2 I2 é sobreposto por I1
$I_1 \mathrm{s} I_2$ $I_2 \mathrm{si} I_1$		I1 inicia com I2 I2 é iniciado por I1
$I_1 \mathrm{d} I_2$ $I_2 \mathrm{di} I_1$		I1 durante I2 I2 contém I1
$I_1 \mathrm{f} I_2$ $I_2 \mathrm{fi} I_1$		I1 finaliza I2 I2 é finalizado por I1
$I_1 = I_2$		I1 equivale a I2

Figura 5.13: Lógica Temporal. Adaptado de (GUESGEN; MARSLAND, 2010)

Os monitores de sinais vitais que estão sendo considerados neste cenário de uso possuem comunicação *Wi-Fi*, o que possibilita fazer a leitura dos sinais vitais coletados e verificar o estado de seus sensores/alarmes remotamente. Também, na perspectiva deste cenário de uso, a administração de medicamentos ao paciente acontece através de bombas de infusão. A verificação do estado de seus sensores/alarmes e o andamento da infusão,

podem acontecer remotamente, através do recurso de comunicação *Wi-Fi* das mesmas.

A saída do paciente da UTI é identificada através de etiqueta RFID, sendo a localização do paciente avaliada considerando a Lógica Temporal e Espacial. Nesse sentido, conforme mostra a Figura 5.14, considera-se `X` como sendo o paciente e `Y` como sendo a UTI. Sempre que o paciente sair temporariamente da UTI por algum motivo, como para a realização de um exame, os limites considerados como esperados na análise dos sinais vitais em função da Meta Terapêutica são desativados. Esta adequação é feita para evitar alertas gerados pelo fato do paciente não estar mais em um ambiente controlado, pois, fora deste, o paciente tem um suporte parcial de equipamentos eletromédicos. Com a eventual desconexão de equipamentos, como bombas de infusão, alguns medicamentos podem deixar de ser administrados ao mesmo e, com isso, os sinais vitais passam a ter sua evolução diferente do que é esperado pelo médico que definiu a Meta Terapêutica. Por sua vez, a situação emocional do paciente durante seus deslocamentos ao longo dos vários ambientes da infraestrutura hospitalar pode se modificar, afetando o estado geral de seus sinais vitais.

Relação	Ilustração	Interpretação
DC(X,Y)		*X desconectado de Y*
EC(X,Y)		*X externamente conectado a Y*
PO(X,Y)		*X sobrepõe parcialmente Y*
EQ(X,Y)		*X idêntico a Y*
TPP(X,Y)		*X tangencia e pertence a Y*
TPPi(X,Y)		*Y tangencia e pertence a X*
NTPP(X,Y)		*X não tangencia e pertence a Y*
NTPPi(X,Y)		*Y não tangencia e pertence a X*

Figura 5.14: Lógica Espacial. Adaptado de (GUESGEN; MARSLAND, 2010)

Aplicações Desenvolvidas

Duas aplicações foram desenvolvidas com o objetivo de acompanhamento e avaliação das Metas Terapêuticas. A primeira é direcionada para uso com navegadores Web, enquanto a segunda é direcionada para a plataforma Android.

Através da aplicação para Web é possível: (i) consultar os dados de contexto coletados pelos sensores dos equipamentos eletromédicos e armazenados no Servidor de Contexto, visualizando-os através de gráficos e tabelas; (ii) visualizar relatórios textuais com os dados coletados; (iii) acompanhar a evolução de mais de um sinal vital em um mesmo gráfico, permitindo compará-los; e (iv) realizar análises estatísticas dos dados coletados.

Na tela inicial da aplicação para navegadores é possível fazer a seleção de pacientes (vide Figura 5.15) e os dados clínicos (vide Figura 5.16) de interesse do profissional de saúde.

Figura 5.15: Interface de Seleção do Paciente

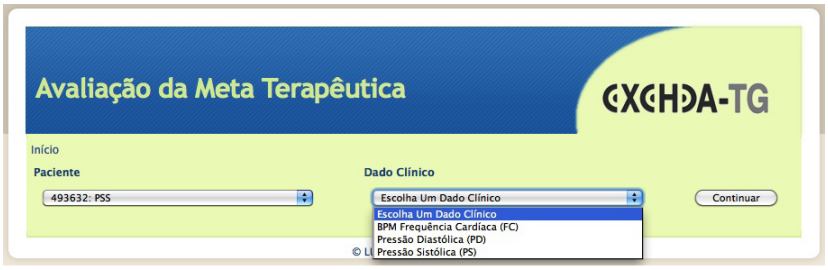


Figura 5.16: Interface de Seleção do Dado Clínico

Na Figura 5.17 é apresentado um gráfico típico da evolução dos sinais vitais. Através do mesmo é possível acompanhar a variação dos sinais vitais em função da medicação administrada ao paciente. Posicionando o *mouse* no gráfico é detalhado o valor e o momento (data e horário) de coleta do mesmo.

Figura 5.17: Interface Gráfica

Também, é possível visualizar mais de um sinal vital simultaneamente em um mesmo gráfico (vide Figura 5.18), sendo viabilizada assim a comparação entre os mesmos.

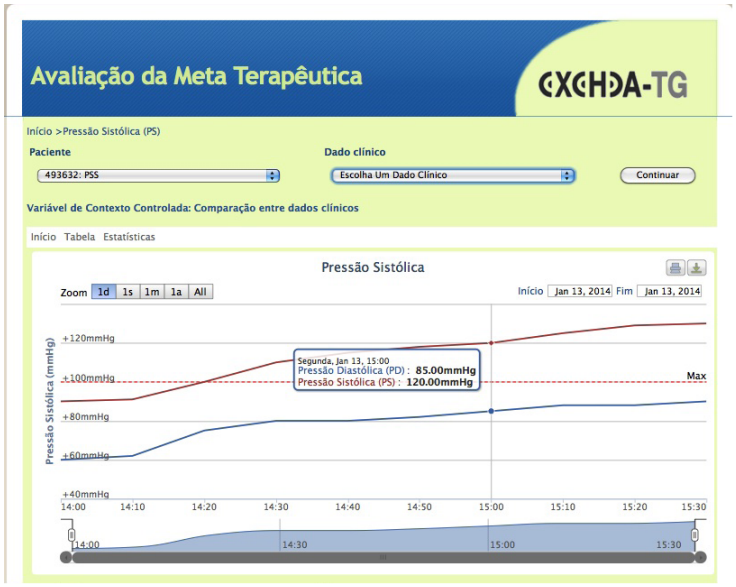


Figura 5.18: Comparação entre Dados Clínicos

Através da opção da interface Web pode-se visualizar os sinais vitais de um paciente coletados na última semana (vide Figura 5.19). Estas informações são apresentadas em colunas, uma para cada dia da semana, onde também é possível ver os valores médio, máximo e mínimo de cada dia.

Figura 5.19: Visualização dos Dados Coletados

Na opção Estatística (vide Figura 5.20) é apresentada uma funcionalidade da aplicação Web que viabiliza o cruzamento de dados contextuais envolvendo múltiplos sinais vitais a partir de diferentes regras. Esse recurso permite a adição, remoção e edição de regras e parâmetros.

Na Figura 5.21 é apresentado um exemplo de alerta enviado aos profissionais de saúde através de e-mail e mensagem SMS. Estes alertas são gerados pela Arquitetura SAUI quando os sinais vitais evoluem de forma diferente do esperado pelo médico ou ocorre algum alarme nas bombas de infusão ou monitores de sinais vitais. Isto é realizado através

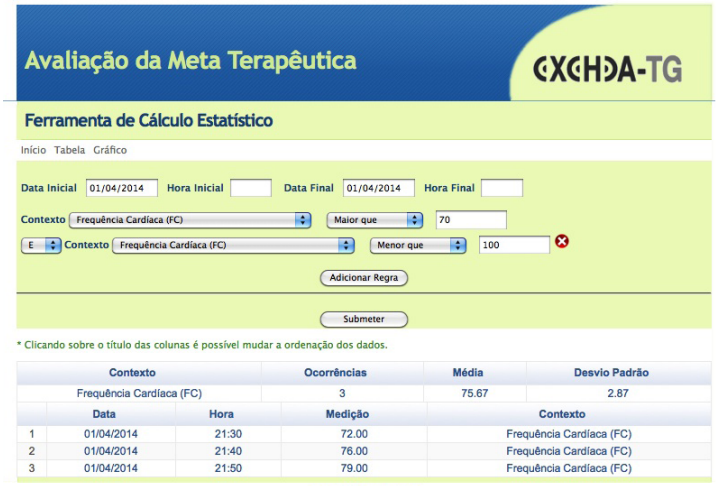


Figura 5.20: Informações Estatísticas

do mecanismo de Raciocínio do Gerenciador de Situação do Servidor de Contexto.

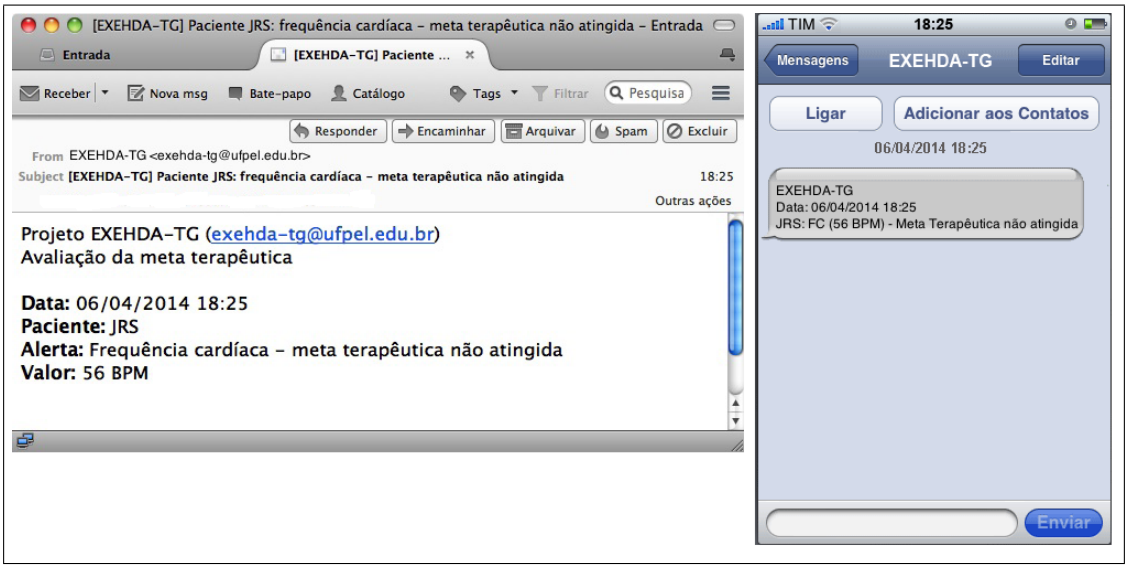


Figura 5.21: Alerta Enviado por E-mail e SMS

O acesso através de dispositivos móveis é voltado para *smartphones* e/ou *tablets* que empregam o sistema operacional Android. Na Figura 5.22 - A é apresentada a tela de abertura da aplicação, através da qual é possível selecionar um paciente, o dado clínico de interesse e a forma (gráfica ou textual) desejada para visualizar o mesmo.

Na tela de informações gráficas da aplicação móvel (vide Figura 5.22 - B) é apresentado o gráfico com a evolução dos sinais vitais, onde é possível selecionar o período de tempo desejado de exibição (uma hora, um dia, uma semana ou um ano). Ao clicar na curva do sinal vital é apresentado o valor e o momento (data e horário) de coleta do mesmo. O eixo vertical do gráfico é ajustado de forma automática, levando em consideração os valores mínimos e máximos a serem plotados.

Na Figura 5.22 - C é apresentado um exemplo de relatório textual de dados clínicos de um paciente. Para disponibilização dos alertas ao usuário foi utilizado o mecanismo nativo de notificação da plataforma Android (vide Figura 5.22 - D). Esta opção tem como

característica propiciar ao usuário um gerenciamento integrado da natureza dos alertas praticados pelo seu dispositivo móvel.

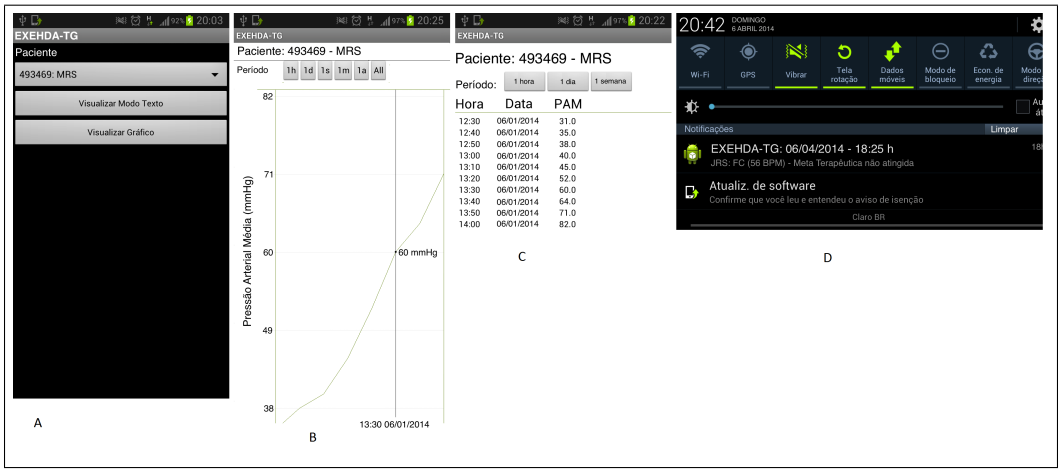


Figura 5.22: Interfaces da Aplicação Android

Identificação de Situações: Reabilitação Cardíaca de Pacientes

A reabilitação cardíaca pode envolver diversas terapias, sendo considerada uma terapia central a prescrição de atividades físicas. Estudos indicam que a reabilitação baseada em exercícios físicos foi associada a uma redução de 20 a 30% nas taxas de mortalidade, quando comparada com cuidados sem exercício (RABELO; GIL; ARAÚJO, 2006).

Um aspecto que deve ser considerado na terapia através de exercícios físicos diz respeito a respostas desproporcionais na frequência cardíaca, o que pode indicar uma situação de risco para o paciente. Desta forma, o reconhecimento da atividade física do paciente e a correlação com a sua frequência cardíaca pode permitir uma recuperação mais segura (NEGRÃO; BARRETO, 2010).

Nesse sentido, o emprego de sistemas computacionais para detecção autônoma de atividades e a correlação destas com parâmetros fisiológicos pode minimizar a necessidade de intervenção do próprio paciente ou do terapeuta na identificação de situações de risco.

Assim, neste cenário para avaliação das funcionalidades da Arquitetura SAUI, foi concebida uma aplicação para monitoramento de pacientes em reabilitação cardíaca. Esta aplicação explora as funcionalidades da SAUI relacionadas ao processamento híbrido do contexto, especialmente o uso combinado de técnicas baseadas em especificação e aprendizagem para identificação de situações, bem como emprega a API do tipo REST disponibilizada através do Módulo de Comunicação do Servidor de Contexto para compor os fluxos de processamento dos dados contextuais.

Cenário Concebido

Este cenário consiste em classificar a atividade física realizada pelo paciente, e correlacioná-la com a frequência cardíaca, permitindo identificar situações de risco, caso os parâmetros estejam fora da faixa de normalidade estabelecida pelo terapeuta. A saída do sistema representa o nível de risco de saúde do paciente, classificado no domínio linguístico como “baixo”, “moderado” e “alto”.

A Arquitetura SAUI executa a detecção da atividade física do paciente e coleta a informação de frequência cardíaca do mesmo. Através de uma interface de configuração são fornecidos os parâmetros de normalidade do paciente, utilizados no processamento dos dados de contexto pela SAUI, visando à identificação de situações relacionadas a risco de saúde do paciente.

Uma visão geral deste cenário de uso é mostrada na Figura 5.23, onde é possível ver a aquisição de dados contextuais, seu processamento e a possível notificação de situações de risco. A detecção da atividade física do paciente é feita utilizando algoritmos de classificação baseados em técnicas de aprendizagem executadas sobre os sinais coletados do *smartphone* do paciente. As regras para identificação de potencial risco de saúde são criadas utilizando Lógica Fuzzy, considerando a atividade física e a frequência cardíaca do paciente.

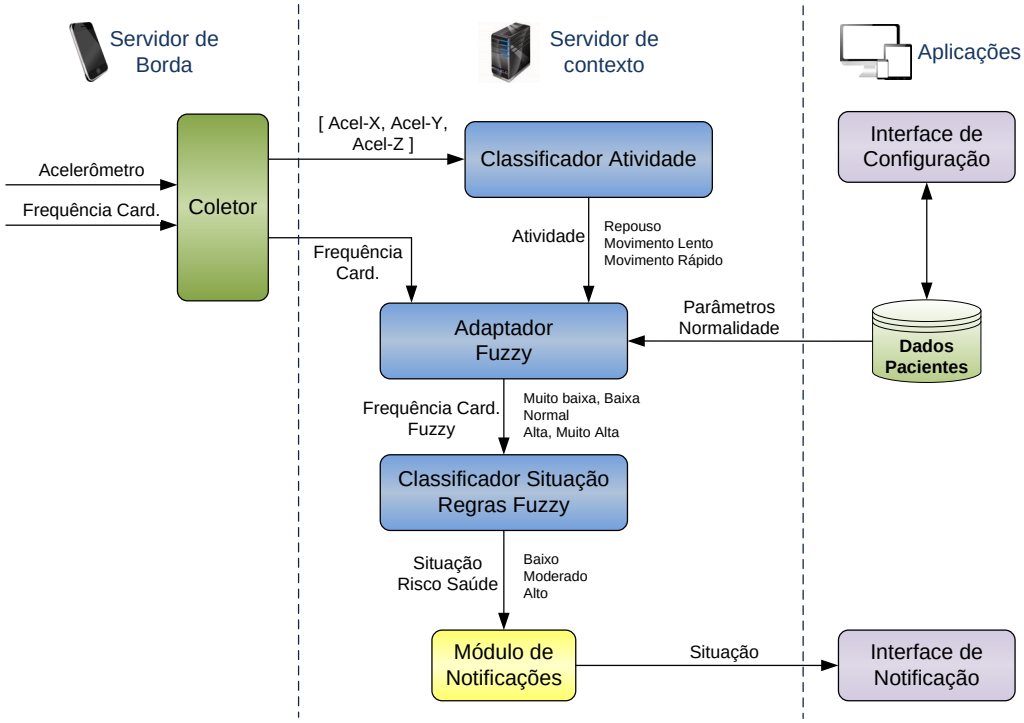


Figura 5.23: Visão Geral do Cenário de Reabilitação Cardíaca

O treinamento e o teste do componente para classificação de atividades foi realizado com o uso da base de dados disponibilizada no trabalho de KWAPISZ; WEISS; MOORE (2011). Neste trabalho foram capturados sinais de acelerômetros de *smartphones* de 29 voluntários durante a execução das seguintes atividades físicas: caminhando, correndo, subindo escada, descendo escada, sentado e de pé. A base de dados empregada contém tanto sinais pré-processados com as características extraídas, como também os valores obtidos diretamente do acelerômetro do celular.

Os testes foram realizados com quatro tipos de classificadores: Árvore de Decisão C4.5, Rede Bayesiana, Rede Neural e Regressão Lógica. Estes classificadores são apontados na literatura como os mais utilizados no reconhecimento de atividades através de sensores (LARA; LABRADOR, 2012).

O primeiro teste realizado consistiu em executar a classificação de atividades sobre as características já pré-processadas por KWAPISZ; WEISS; MOORE (2011). Os resultados para este teste podem ser vistos na Tabela 5.3. Os autores utilizaram um total de 43

características do sinal, incluindo: média, desvio padrão e tempo entre picos, além de um método empírico de distribuição intervalar de máximos e mínimos que gera 30 características. Este último método não foi claramente descrito pelos autores.

Tabela 5.3: Percentual de Atividades Classificadas Corretamente - Características do Estudo de KWAPISZ; WEISS; MOORE (2011)

	Árvore de Decisão	Rede Bayesiana	Rede Neural	Regressão Lógica
Caminhando	87.33%	82.94%	89.23%	92.37%
Correndo	95.80%	94.19%	98.89%	97.41%
Subindo escadas	59.21%	39.31%	52.96%	24.51%
Descendo escadas	54.66%	30.64%	40.17%	10.35%
Sentado	98.66%	94.98%	98.33%	95.32%
De pé	97.56%	84.96%	93.90%	91.06%
Total	**84.88%**	77.50%	84.39%	79.05%

O segundo teste realizado consistiu em treinar os classificadores utilizando um novo conjunto de características extraídas dos sinais lidos diretamente do acelerômetro. O sinal do acelerômetro foi obtido com uma taxa de 20 amostras por segundo e o processamento das características foi executado em janelas temporais de 10 segundos. Para realizar essa atividade foi criada uma rotina de importação e processamento dos dados utilizando o software Matlab, gerando um novo conjunto com 15 características, listadas a seguir:

- Frequência das três componentes de maior amplitude da FFT calculada sobre o módulo da aceleração resultante.
- Aceleração média nos eixos x, y e z.
- Variância nos eixos x, y e z.
- Aceleração máxima nos eixos x, y e z.
- Aceleração mínima nos eixos x, y e z.

Na Figura 5.24 pode ser visto o histograma da distribuição em frequência das componentes de frequência com maior amplitude para as atividades classificadas. Nota-se no histograma que existe uma diferença bem definida da frequência média para cada atividade. Isso indica que a FFT é uma boa característica para uso nos classificadores.

O resultado obtido com o novo conjunto de características pode ser visto na Tabela 5.4. Pode-se observar que houve uma melhora na performance média dos classificadores de cerca de 7,0%. Além disso, o número de características utilizadas foi reduzido de 43 para 15. Isso simplifica o treinamento e a execução dos modelos de aprendizagem.

Com a finalidade de adequar as atividades para a aplicação proposta, as mesmas foram agrupadas como:

- **Repouso**: sentado e de pé;
- **Movimento lento**: caminhando, descendo escada e subindo escada;
- **Movimento rápido**: correndo.

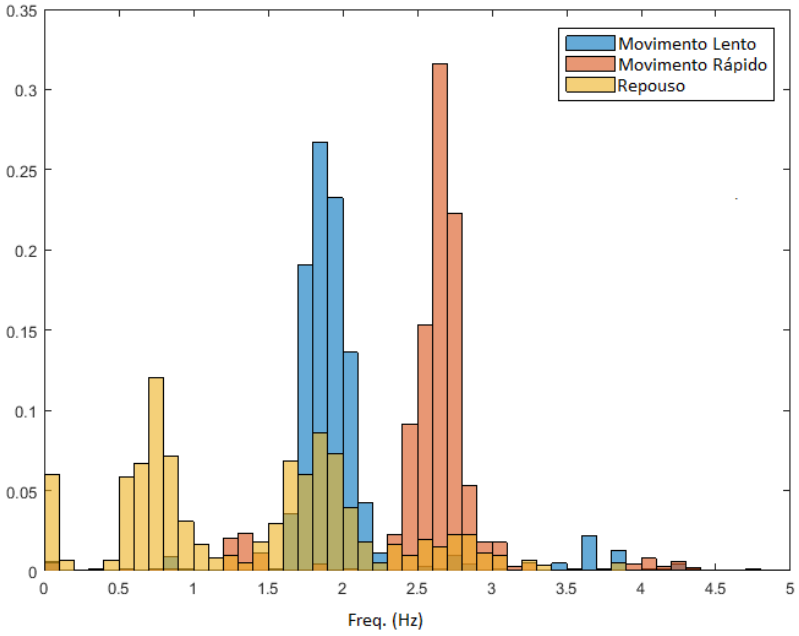


Figura 5.24: Histograma da Distribuição em Frequência - Atividades Classificadas

Tabela 5.4: Percentual de Atividades Classificadas Corretamente - Novo Conjunto de Características

	Árvore de Decisão	Rede Bayesiana	Rede Neural	Regressão Lógica
Caminhando	92,31%	92,55%	93,35%	91,37%
Correndo	97,41%	98,71%	97,35%	96,94%
Subindo escadas	76,81%	60,50%	75,63%	63,03%
Descendo escadas	68,98%	62,86%	51,22%	40,00%
Sentado	97,90%	94,41%	93,71%	94,06%
De pé	97,40%	99,57%	96,54%	94,81%
Total	**90,61%**	88,68%	89,01%	85,65%

Utilizando a classificação agrupada e o conjunto de características proposto para este cenário, o processo de treinamento e teste dos classificadores foi repetido. O classificador Árvore de Decisão se manteve com a melhor precisão. Após o treinamento, foi gerada uma Árvore de Decisão com 40 ramos e 79 elementos. O resultado da classificação de atividades para a Árvore de Decisão é apresentado na matriz de confusão disponível na Tabela 5.5. O percentual total de classificação correta obtido foi de **98,32%**.

Na etapa de inferência de situação é feita a determinação do nível de risco de saúde do paciente baseado na sua atividade, previamente classificado e na sua frequência cardíaca.

A frequência cardíaca apropriada para cada atividade física é obtida através de uma interface para a aplicação em que é feito o gerenciamento dos pacientes. O domínio linguístico para frequência cardíaca é dado por: “muito baixa”, “baixa”, “normal”, “alta” e “muito alta”. No bloco **Adaptador Fuzzy**, representado na Figura 5.23 os valores numéricos obtidos do sensor de frequência cardíaca são transformados para o domínio fuzzy aplicando funções de pertinência triangulares, parametrizadas com os padrões de normalidade do paciente. O mesmo procedimento é realizado para a variável que representa a duração da atividade, classificada como “Baixa”, “Média” e “Alta”.

Tabela 5.5: Matriz de Confusão para Classificação de Atividades Utilizando Árvore de Decisão

	Movimento lento	Movimento rápido	Repouso
Movimento lento	**98,60%**	2,36%	1,15%
Movimento rápido	1,25%	**97,64%**	0,00%
Repouso	0,16%	0,00%	**98,85%**
Total correto	**98,32%**		

A escolha da lógica fuzzy para a criação das regras para classificação de situações se deu tendo em vista os seus mecanismos para tratamento de dados imprecisos e construção de algoritmos através de modelos interpretáveis, isto facilita a criação de regras por especialistas da área de aplicação (SOBREVILLA; MONTSENY, 2003).

A inferência foi feita aplicado o método Mamdani, no qual, são previstos quatro módulos: fuzzificação, base de regras, inferência e defuzzificação (SHAW; SIMOES, 2007).

Na Figura 5.25 podem ser vistos exemplos de funções de pertinência utilizadas na *fuzzificação* da frequência de batimento cardíaco e duração da atividade. Observa-se que são estabelecidas funções de pertinência para a frequência cardíaca em cada uma das atividades que podem ser identificadas pela aplicação. Isto simplifica a criação da base de regras fuzzy, visto que, a frequência cardíaca tratada nas regras é previamente parametrizada pelo tipo de atividade.

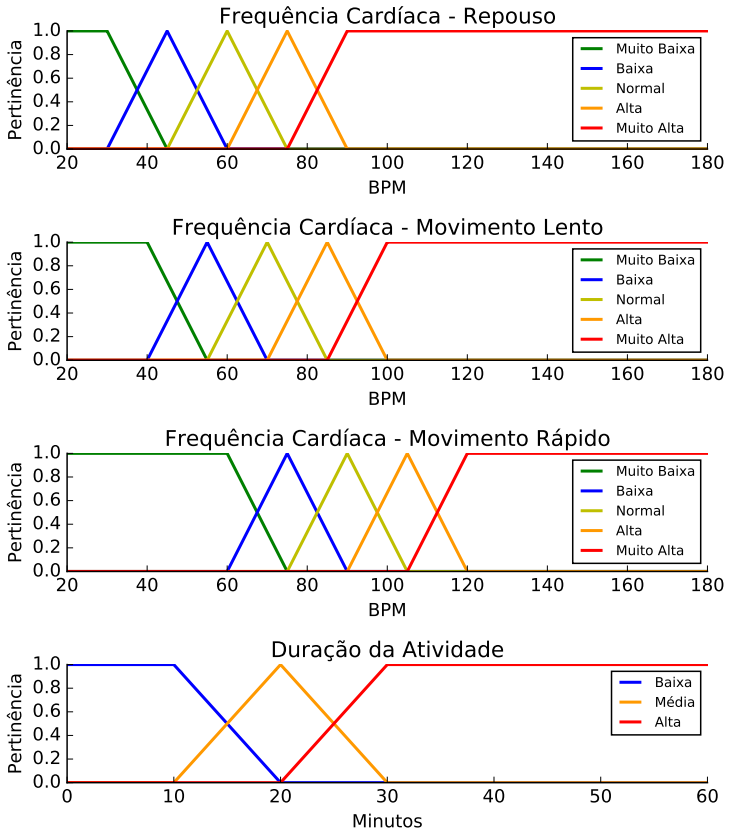


Figura 5.25: Funções de Pertinência para Frequência Cardíaca

A base de regras fuzzy criada para este cenário de uso pode ser vista na Figura 5.26. Foram aplicados três tipos de operadores fuzzy:

- OR = Máximo(x,y).
- AND = Mínimo(x,y).
- NOT = 1 - x.

Onde x e y são valores de pertinência de variáveis linguísticas (SHAW; SIMOES, 2007).

Cada regra é formada por um conjunto de antecedentes que, relacionados através dos operadores fuzzy, geram um valor de pertinência resultante. Este valor resultante é utilizado para ponderar a função de pertinência da variável linguística consequente. Neste trabalho, é aplicado o método *clipped*, onde a função de pertinência do consequente é "cortada" para o nível de pertinência resultante dos antecedentes avaliados na regra (SHAW; SIMOES, 2007).

```
IF freq_card IS normal THEN risco IS baixo
IF atividade IS repouso AND freq_card IS alta THEN risco IS moderado
IF atividade IS repouso AND freq_card IS baixa THEN risco IS moderado
IF atividade IS repouso AND freq_card IS muito_alta THEN risco IS alto
IF atividade IS repouso AND freq_card IS muito_baixa THEN risco IS alto
IF atividade IS NOT repouso AND duracao IS baixa OR media AND freq_card IS alta THEN risco IS moderado
IF atividade IS NOT repouso AND duracao IS baixa OR media AND freq_card IS baixa THEN risco IS moderado
IF atividade IS NOT repouso AND duracao IS baixa OR media AND freq_card IS muito_alta THEN risco IS alto
IF atividade IS NOT repouso AND duracao IS baixa OR media AND freq_card IS muito_baixa THEN risco IS alto
IF atividade IS NOT repouso AND duracao IS alta AND freq_card IS alta THEN risco IS alto
IF atividade IS NOT repouso AND duracao IS alta AND freq_card IS baixa THEN risco IS alto
IF atividade IS NOT repouso AND duracao IS alta AND freq_card IS muito_alta THEN risco IS alto
IF atividade IS NOT repouso AND duracao IS alta AND freq_card IS muito_baixa THEN risco IS alto
```

Figura 5.26: Base de Regras para Inferência da Situação de Risco de Saúde do Paciente

Após a execução das regras são geradas funções de pertinência agregadas para cada variável linguística do conjunto de saída, as quais são agregadas e aplicadas na etapa de defuzzificação.

A saída agregada obtido através das regras é deffuzificadas utilizando o método do centróide (SHAW; SIMOES, 2007). Na Figura 5.27 são mostradas as funções de pertinência envolvidas no processo de inferência da situação de risco de saúde para a atividade "Movimento Rápido" com duração de 40 minutos e frequência cardíaca de 110 BPM. A situação inferida após a defuzzificação é "Risco Moderado".

Para verificar o funcionamento da aplicação desenvolvida foram gerados dados através de um *smartphone*, enviando para a Arquitetura SAUI os valores de frequência cardíaca e os vetores de aceleração obtidos do banco de dados disponibilizado por KWAPISZ; WEISS; MOORE (2011).

O registro gráfico criado para uma sequência simulada nesse cenário pode ser visto na Figura 5.28, sendo destacado, através do cursor vertical, o momento em que foi inferida uma situação com risco "Alto". A ocorrência dessa situação notificada através de e-mail para o terapeuta, com o uso do Módulo de Notificação da Arquitetura SAUI.

A estrutura de processamento adotada pela Arquitetura SAUI foi modelada para permitir acesso aos componentes de processamento através de uma API do tipo REST (vide Seção 4.5.1). Assim, para compor o fluxo de processamento mostrado na Figura 5.23 foram alocados recursos através da API do tipo REST do Módulo de Comunicação do Servidor de Contexto, os quais são apresentados na Tabela 5.6.

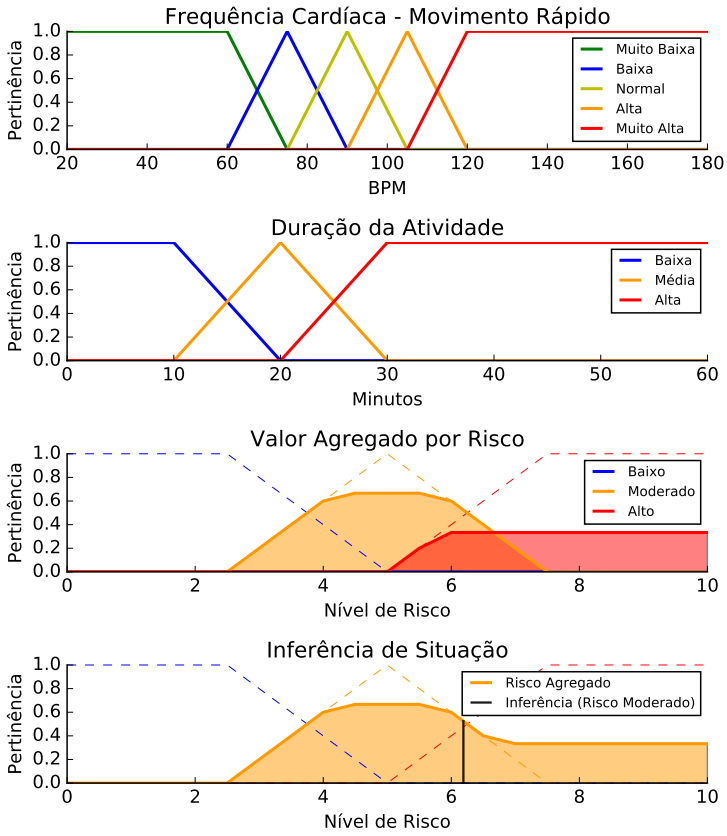


Figura 5.27: Exemplo de Inferência de Situação de Risco de Saúde

5.2 Avaliação de Usabilidade

Esta seção apresenta a avaliação de usabilidade das funcionalidades concebidas para a Arquitetura SAUI, considerando a aplicação prototipada para o cenário de uso da área de agropecuária.

Segundo KNAPPMEYER et al. (2013), a necessária transparência decorrente da operação autônoma de *middlewares* em aplicações Cientes de Situação introduz uma dificuldade quanto à avaliação das funcionalidades propostas. Nesse sentido, uma das principais estratégias utilizadas para avaliar *middlewares* é o emprego de aplicações, nas quais as experiências dos usuários podem ser exploradas de maneira explícita ou implícita.

A experiência do usuário com a aplicação pode ser avaliada de forma explícita, com o emprego de entrevistas e questionários, ou implicitamente através de observações. Essa estratégia pode refletir diretamente a usabilidade das aplicações, indicando a capacidade da arquitetura em atender os requisitos das mesmas, bem como possibilitando verificar o correto funcionamento de seus módulos (KNAPPMEYER et al., 2013).

5.2.1 Metodologia Empregada

O experimento realizado para avaliação de usabilidade da aplicação prototipada com a Arquitetura SAUI empregou o Modelo de Aceitação de Tecnologia (TAM - *Technology Acceptance Model*) proposto por DAVIS (1989). O TAM foi projetado para compreender a relação causal entre variáveis externas de aceitação dos usuários e o uso real de uma tecnologia da informação, buscando entender o comportamento deste usuário através do conhecimento da utilidade e da facilidade de utilização percebida por ele (YOON; KIM,

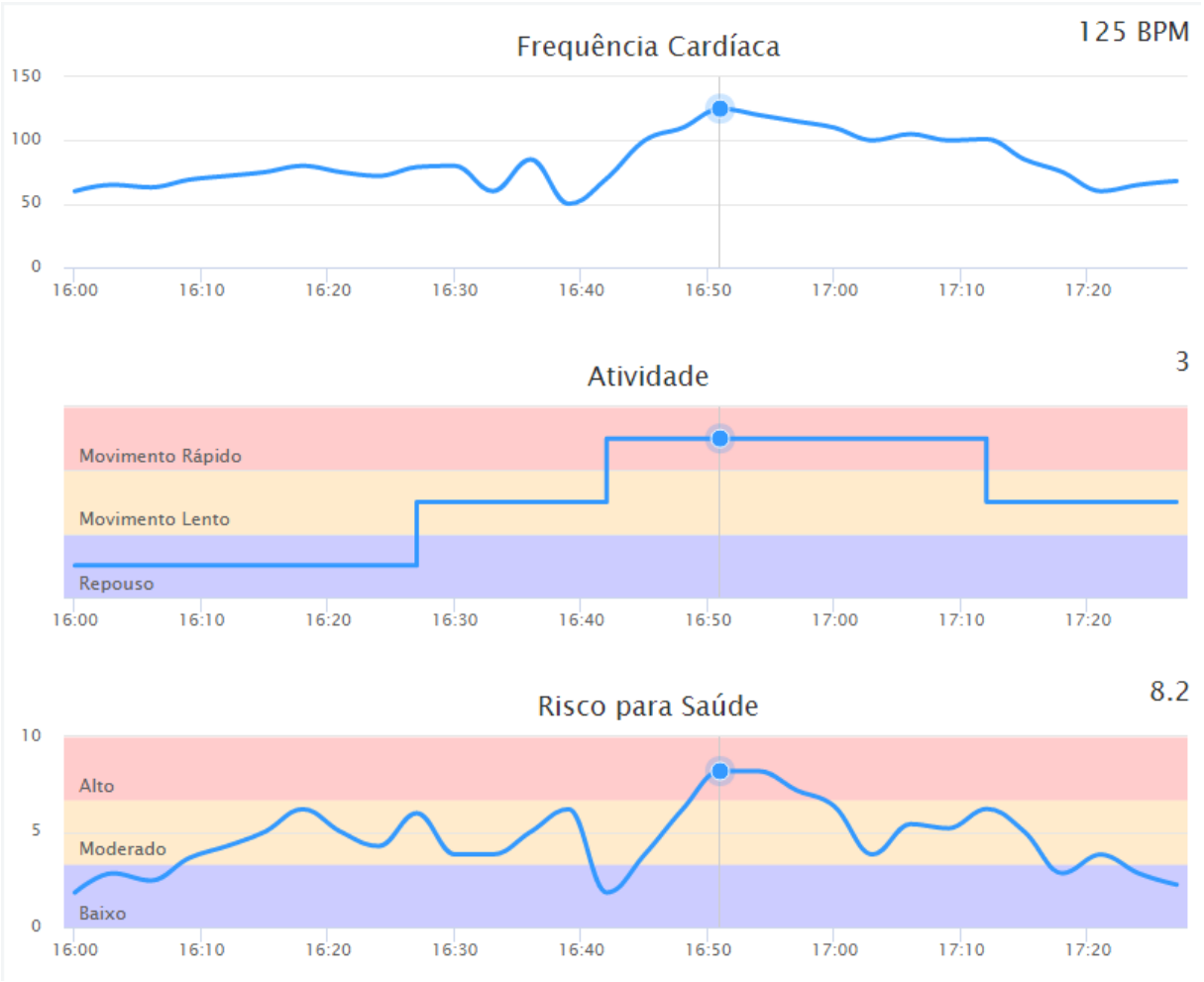


Figura 5.28: Registro Gráfico de Frequência Cardíaca, Atividade e Inferência de Risco

2007).

Para DAVIS (1989) as pessoas tendem a usar uma tecnologia com o objetivo de melhorar o desenvolvimento de suas atividades, esse aspecto corresponde a "utilidade percebida". Porém, mesmo que uma pessoa entenda que uma determinada tecnologia é útil, a utilização da mesma poderá ser prejudicada em função de sua complexidade. Nesse caso, a "facilidade percebida" traduz o esforço de uso de uma tecnologia.

Desta forma, as pessoas usarão a tecnologia se acreditarem que este uso fornecerá resultados positivos, tendo em vista a facilidade de uso percebida e a utilidade percebida. Assim, o TAM normalmente é utilizado para entender os motivos que levam os usuários a aceitar ou rejeitar uma tecnologia de informação e como melhorar a aceitação, oferecendo, desse modo, um suporte para prever e explicar a aceitação (RAITOHARJU, 2007).

Para a avaliação da Arquitetura SAUI, dentre as aplicações desenvolvidas, tendo como critério o número de usuários, a Aplicação de Visualização de Informações Contextuais (vide Seção 5.1.2) utilizada no LASO (Laboratório de Análise de Sementes da Embrapa) foi escolhida para ter sua usabilidade avaliada. Dentre as 20 pessoas que desenvolvem atividades técnico-administrativas relacionadas ao LASO, foram selecionadas aleatoriamente 10 para participar do experimento.

Cada participante foi orientado a utilizar a aplicação, tanto com dispositivo móvel como com desktop. Na sequência, os participantes responderam um questionário de

Tabela 5.6: Recursos da API do Tipo REST para Identificação de Situação de Risco de Saúde

URI	Descrição
Contexto-Processado/Atividade	Classificação da atividade empregando árvore de decisão
Contexto-Processado/Freq-Cardiaca-Fuzzy	Frequência cardíaca em domínio fuzzy
Contexto-Processado/Situacao	Indicação do risco de saúde do paciente
Instancia-Processamento/Adaptador-Fuzzy	Instância de componente de processamento para conversão da frequência cardíaca para domínio fuzzy
Instancia-Processamento/Classificador-Atividade	Instância de componente de processamento para classificação de atividade utilizando árvore de decisão
Instancia-Processamento/Classificador-Situacao	Classificador de situação utilizando regras em lógica fuzzy

avaliação, considerando a experiência de uso. As questões foram elaboradas com base no TAM, considerando: (i) Facilidade de uso: grau em que o participante do experimento avalia que a aplicação pode reduzir seu esforço; e (ii) Percepção de utilidade: grau em que o participante do experimento avalia que a aplicação pode melhorar a sua experiência.

As alternativas de resposta das questões foram estruturadas de acordo com a escala Likert, sendo constituídas por cinco opções que avaliam a experiência do participante com a aplicação. A escala de Likert fornece cinco alternativas em um intervalo que começa em 1 (discordo totalmente) até 5 (concordo totalmente).

A consistência interna de um questionário refere-se ao grau com que os itens do mesmo estão correlacionados entre si e com o resultado geral da pesquisa, o que representa uma mensuração da confiabilidade do questionário. Um dos procedimentos estatísticos mais utilizados para mensuração da consistência interna é o coeficiente alfa de Cronbach, apresentado por Lee J. Cronbach em 1951 (HORA; MONTEIRO; ARICA, 2010). Assim, neste experimento é empregado o coeficiente alfa de Cronbach para verificação da confiabilidade do questionário para avaliação de usabilidade da aplicação utilizada no LASO.

5.2.2 Questionário Aplicado

A Tabela 5.7 contém o questionário aplicado aos participantes, sendo que as cinco primeiras questões correspondem à facilidade de uso e as demais dizem respeito à percepção de utilidade. As respostas obtidas são mostradas na Tabela 5.8. Nessa Tabela, a primeira coluna corresponde à questão, as seguintes cinco colunas apresentam os resultados obtidos em cada escala, em graus relativos e absolutos, e a última coluna mostra a média consolidada da percentagem, variando de 0 a 5.

A Figura 5.29 mostra o cálculo do coeficiente alfa de Cronbach. Este coeficiente é calculado a partir da variância dos itens individuais e da variância da soma dos itens de cada participante de todos os itens de um questionário que utilizem a mesma escala de medição. O coeficiente deste experimento foi de 0,74 o que indica que o questionário é

Tabela 5.7: Questionário de Avaliação

Nro.	Categoria	Questão
01	Facilidade de uso	A aplicação é fácil de entender.
02	Facilidade de uso	A aplicação é fácil de usar.
03	Facilidade de uso	As opções são claras e objetivas.
04	Facilidade de uso	Com pouco esforço consigo selecionar um contexto de interesse.
05	Facilidade de uso	Com pouco esforço consigo acessar os relatórios gráficos.
06	Utilidade	As opções apresentadas são relevantes.
07	Utilidade	A aplicação facilita a obtenção de dados de contexto envolvendo múltiplos sensores.
08	Utilidade	A aplicação facilita a minha mobilidade.
09	Utilidade	A aplicação facilita a tomada de decisões a partir do envio de uma mensagem.
10	Utilidade	Eu usaria essa aplicação no meu trabalho.

confiável por estar acima do valor de 0,7 (HORA; MONTEIRO; ARICA, 2010).

Participantes	Questões										
	01	02	03	04	05	06	07	08	09	10	Total
1	5	5	5	5	5	4	5	5	5	5	49
2	5	5	5	5	5	5	5	5	5	5	50
3	4	5	5	5	5	5	4	4	5	5	47
4	5	4	4	5	4	5	5	5	3	3	43
5	5	5	5	5	5	5	4	4	5	5	48
6	5	5	5	4	4	5	5	4	4	4	45
7	4	4	3	4	4	5	4	5	3	3	39
8	4	5	5	5	5	4	5	5	5	5	48
9	5	5	5	5	5	4	5	4	5	5	48
10	4	4	4	5	5	5	5	5	4	3	44
Variância	0,24	0,21	0,44	0,16	0,21	0,21	0,21	0,24	0,64	0,81	

k	10
Somatório Variância	3,37
Variância Total	10,09
Alfa	0,74

Figura 5.29: Cálculo do Coeficiente Alfa de Cronbach

Analisando os resultados pode-se observar que a aprovação é alta, tanto para facilidade de uso, como para percepção de utilidade. Entretanto, ocorrem resultados na escala "indiferente" nas duas últimas questões da percepção de utilidade. Isso pode ser interpretado como uma preocupação com o controle da qualidade dos experimentos desenvolvidos no LASO, em função do uso de mecanismos sem a usual intervenção humana na emissão de alertas para estados contextuais que exijam uma atuação imediata. Nesse caso, uma estratégia que pode ser adotada é intensificar os testes e validações com os usuários e iniciar uma implantação gradativa das aplicações.

Tabela 5.8: Avaliação da Facilidade de Uso e da Percepção de Utilidade

Questão	Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente	Média
01	0,0%(0)	0,0%(0)	0,0%(0)	40,0%(4)	60,0%(6)	4,6
02	0,0%(0)	0,0%(0)	0,0%(0)	30,0%(3)	70,0%(7)	4,7
03	0,0%(0)	0,0%(0)	0,0%(0)	20,0%(3)	70,0%(7)	4,7
04	0,0%(0)	0,0%(0)	0,0%(0)	20,0%(2)	80,0%(8)	4,8
05	0,0%(0)	0,0%(0)	0,0%(0)	30,0%(3)	70,0%(7)	4,7
06	0,0%(0)	0,0%(0)	0,0%(0)	30,0%(3)	70,0%(7)	4,7
07	0,0%(0)	0,0%(0)	0,0%(0)	40,0%(3)	60,0%(7)	4,7
08	0,0%(0)	0,0%(0)	0,0%(0)	40,0%(4)	60,0%(6)	4,6
09	0,0%(0)	0,0%(0)	20,0%(2)	20,0%(2)	60,0%(6)	4,4
10	0,0%(0)	0,0%(0)	30,0%(3)	10,0%(1)	60,0%(6)	4,3

5.3 Considerações do Capítulo

Este capítulo apresentou a avaliação das funcionalidades da Arquitetura SAUI através de cenários de uso das áreas de agropecuária e saúde. Estes cenários foram considerados em decorrência de projetos de pesquisa associados a esta tese. A utilização de cenários em diferentes áreas também aponta a possibilidade de utilização da Arquitetura SAUI em diferentes domínios de aplicação.

Neste capítulo foram apresentadas as tecnologias de software empregadas na implementação dos componentes da Arquitetura SAUI e na prototipação das aplicações que utilizam a arquitetura, bem como a tecnologia de hardware definida para os equipamentos servidores de contexto e de borda.

Na perspectiva destes cenários de uso foram desenvolvidos testes das funcionalidades da Arquitetura SAUI, abrangendo desde a coleta dos dados de contexto até o processamento destes para a identificação de situações e a consequente notificação da ocorrência destas a consumidores interessados. O processo, descrito nesse capítulo, de coleta, armazenamento e processamento dos dados de contexto, bem como de identificação de situações caracterizou a abordagem distribuída, híbrida, dirigida por eventos e regras, que foi definida nesta tese para a Arquitetura SAUI.

Ainda, este capítulo apresentou uma avaliação de usabilidade das funcionalidades da Arquitetura SAUI, utilizando o método TAM. A aplicação prototipada para o cenário de uso da área de agropecuária foi avaliada considerando sua facilidade de uso e utilidade. Os resultados obtidos mostraram uma elevada aprovação da aplicação, o que caracteriza a capacidade da arquitetura em atender os requisitos da mesma.

O próximo capítulo contém as principais conclusões deste trabalho, bem como as contribuições da pesquisa, publicações realizadas e trabalhos futuros.

6 CONSIDERAÇÕES FINAIS

Este capítulo apresenta as considerações finais desta tese, revisitando o problema de pesquisa considerado no desenvolvimento deste trabalho, bem como destacando as principais contribuições da tese e as oportunidades de trabalhos futuros na frente de pesquisa relacionada à Arquitetura SAUI.

6.1 Conclusão

Esta tese tem como problema central de pesquisa "Como deve ser uma arquitetura para provimento de Ciência de Situação direcionada às aplicações ubíquas, considerando a infraestrutura da Internet das Coisas?". Nesse sentido, é concebida a Arquitetura SAUI que pode ser empregada em vários domínios de aplicação, caracterizando-se pelo suporte ao tratamento autônomo dos dados contextuais, de forma distribuída, baseado em eventos e regras, bem como pela definição de uma abordagem híbrida para representação e processamento do contexto e a decorrente identificação de situações em diferentes cenários de uso.

Para tanto, como etapa inicial do desenvolvimento desta tese, é realizada uma revisão de conceitos sobre os seguintes temas: Computação Ubíqua, Ciência de Contexto, Ciência de Situação e Internet da Coisas. Esta etapa do esforço de pesquisa está registrada no Capítulo 2. Ainda, neste Capítulo 2, são apresentadas as premissas de pesquisa do *middleware* EXEHDA, bem como seus principais aspectos arquiteturais e funcionais, considerando o objetivo de instanciar a Arquitetura SAUI no âmbito do projeto ISAM/EXEHDA.

Como etapa seguinte do esforço de pesquisa, no Capítulo 3 é sistematizada uma seleção de trabalhos relacionados, tendo por base as premissas de concepção da Arquitetura SAUI. De modo mais específico, os trabalhos relacionados são discutidos considerando aspectos relacionados ao: (i) emprego da arquitetura em diferentes domínios de aplicação; (ii) uso de uma abordagem distribuída para coleta dos dados de contexto; (iii) emprego de um modelo híbrido para representação do contexto; (iv) suporte à aquisição autônoma dos dados de contexto e à atuação, na perspectiva da infraestrutura da IoT; (v) uso de eventos e regras para coleta e processamento de contexto; e (vi) emprego de abordagem híbrida para processamento de contexto na identificação de situações. A Arquitetura SAUI é então comparada com os trabalhos relacionados, sendo caracterizadas as principais contribuições científicas desta tese, as quais abrangem aspectos relacionados a um suporte concomitante para operação distribuída, tratamento autônomo dos dados contextuais baseado em eventos e regras, e processamento híbrido do contexto, visando à identificação de situações.

Como consequência das contribuições científicas identificadas, a concepção da

Arquitetura SAUI é apresentada no Capítulo 4. As características da arquitetura e as funcionalidades previstas para os diferentes módulos contemplam uma abordagem: (i) distribuída para coleta e processamento dos dados de contexto; (ii) dirigida por eventos que permite a associação de regras do tipo evento-condição-ação aos contextos de interesse; e (iii) híbrida que possibilita o uso combinado tanto de modelos de contexto, como de técnicas para identificação de situações.

As funcionalidades da Arquitetura SAUI são avaliadas através de cenários de uso, em diferentes domínios de aplicação, no Capítulo 5. Considerando a premissa de uso da arquitetura em diferentes áreas são apresentadas aplicações relacionadas à agropecuária e saúde. Também, uma avaliação de usabilidade é realizada com o emprego do método TAM. Para atender o processamento das informações contextuais nestes cenários, destaca-se o emprego da abordagem híbrida, a qual é constituída por um modelo semântico e um modelo sintático para representação do contexto. Esta modelagem híbrida serve de base para o armazenamento das informações de contexto utilizadas no processo de identificação de situações, o qual também adota uma estratégia de processamento híbrido, com o uso de técnicas baseadas em especificação de regras e em aprendizagem de máquina. O emprego desta abordagem híbrida na Arquitetura SAUI mostra-se bastante adequado em ambientes com elevada dinamicidade em sua composição e organização, característica típica dos ambientes ubíquos e da infraestrutura da IoT.

6.2 Contribuições da Pesquisa

Esta seção sintetiza as contribuições da pesquisa realizada ao longo do desenvolvimento desta tese, as quais estão registradas em diferentes pontos do texto.

A contribuição científica central deste trabalho é a concepção de uma arquitetura para suporte à Ciência de Situação das aplicações ubíquas na perspectiva da infraestrutura provida pela Internet das Coisas, a qual pode ser empregada em diferentes domínios de aplicação.

A Arquitetura SAUI possui suporte concomitante para operação distribuída e tratamento autônomo dos dados contextuais baseado em eventos e regras, bem como para processamento híbrido do contexto, visando à identificação de situações. Registre-se que estas características não estão presentes de forma conjunta nos trabalhos relacionados estudados nessa tese.

A seguir estão resumidos os principais resultados atingidos durante o trabalho de estudo e pesquisa relacionados a esta tese de doutorado:

- identificação das principais pesquisas direcionadas à proposição de arquiteturas para suporte à Ciência de Situação;
- definição de uma abordagem para concepção da arquitetura direcionada à Ciência de Situação para aplicações ubíquas na infraestrutura provida pela IoT, sendo concebida a Arquitetura SAUI que contempla essa abordagem;
- proposição de uma abordagem distribuída para coleta e processamento das informações contextuais, bem como para atuação sobre o meio físico;
- definição de uma abordagem híbrida para representação e processamento do contexto, visando à identificação de situações;

- modelagem de componentes de software que viabilizam a coleta e o processamento dos dados contexto com base em regras definidas pelo desenvolvedor da aplicação, considerando eventos relacionados aos dados coletados;

- avaliação das funcionalidades da arquitetura através de cenários de uso e da comparação das mesmas com o estado da arte na área;

- provimento de serviços de Ciência de Situação, na perspectiva de uma abordagem híbrida para processamento contextual, para o Subsistema de Reconhecimento de Contexto e Adaptação do *middleware* EXEHDA;

- divulgação dos resultados do trabalho realizado durante o desenvolvimento da tese através de publicações. As principais estão relacionadas na Seção 6.3.

6.3 Publicações

As principais contribuições da pesquisa foram compartilhadas com a comunidade científica através de publicações. Esta seção apresenta as publicações realizadas em decorrência da pesquisa desenvolvida durante esta tese, abrangendo os anos de 2012 a 2016.

Periódicos

- LOPES, João L. B.; SOUZA, Alexandre; MOURA JUNIOR, Denis; SOUZA, Rodrigo; PERNAS, Ana M.; Yamin, Adenauer C.; GEYER, Cláudio F. R. A situation-aware ubiquitous approach for assessing therapeutic goals in hospital environment. Journal of Applied Computing Research, v. 4, p. 1-12, 2015. (Qualis B5)

- LOPES, João L. B.; SOUZA, Rodrigo; COSTA, Cristiano; BARBOSA, Jorge; PERNAS, Ana M.; YAMIN, Adenauer; GEYER, Cláudio F. R. A Middleware Architecture for Dynamic Adaptation in Ubiquitous Computing. Journal of Universal Computer Science, v. 20, p. 1327-1351, 2014. (Qualis B1)

- LOPES, João L. B.; SOUZA, Rodrigo; GADOTTI, Gizele I.; PERNAS, Ana M.; YAMIN, Adenauer; GEYER, Cláudio F. R. An Architectural Model for Situation Awareness in Ubiquitous Computing. IEEE Latin America Transactions, v. 12, p. 1113-1119, 2014. (Qualis B4)

- LOPES, João L. B.; GUSMAO, Márcia Z.; DUARTE, Cauê; DAVET, Patrícia; SOUZA, Rodrigo; PERNAS, Ana M.; YAMIN, Adenauer; GEYER, Cláudio F. R. Toward a Distributed Architecture for Context Awareness in Ubiquitous Computing. Journal of Applied Computing Research, v. 3, p. 19-33, 2013. (Qualis B5)

- LOPES, João L. B.; SOUZA, Rodrigo; GUSMAO, Márcia Z.; YAMIN, Adenauer Correa; GEYER, Cláudio Fernando Resin. Processamento de Contexto na Ubicomp: Uma Revisão Orientada a Aspectos Semânticos. Revista do CCEI, v. 16, p. 254-273, 2012. (Qualis B5)

Congressos

- LOPES, João L. B.; SCHEUNEMANN, Douglas; REISER, Renata; YAMIN, Adenauer; GEYER, Cláudio F. R. Middleware Architecture for Supporting a Hybrid Processing of Context Data Targeted to Detection of Situations in UbiComp. In: 12th International FLINS Conference - Conference on Uncertainty Modelling in Knowledge Engineering and Decision Making, 2016, Roubaix, France. (Qualis B4)

- LOPES, João L. B.; SOUZA, Alexandre; SOUZA, Rodrigo; DAVET, Patrícia; PERNAS, Ana M.; YAMIN, Adenauer C.; GEYER, Cláudio F. R. A Situation-Aware Pervasive Approach for Assessing Therapeutic Goals in Healthcare Environment. In: ACM Symposium on Applied Computing, 2016, Pisa, Italy. (Qualis A1)

- LOPES, João L. B.; SOUZA, Rodrigo; PERNAS, A. M.; YAMIN, Adenauer; GEYER, Cláudio Fernando Resin. A Distributed Architecture for Supporting Context-Aware Applications in UbiComp. In: International Conference on Advanced Information Networking and Applications, 2014, Victoria, Canada, 2014 v. 1. p. 584-590. (Qualis A2)

- LOPES, João L. B.; GUSMAO, Márcia Z.; SOUZA, Rodrigo; DAVET, Patrícia; PERNAS, Ana M.; YAMIN, Adenauer C.; GEYER, Cláudio F. R. Uma Arquitetura Distribuída Direcionada à Consciência de Contexto na Computação Ubíqua. In: Simpósio Brasileiro de Computação Ubíqua e Pervasiva, 2013, Maceió. XXXIII Congresso da Sociedade Brasileira de Computação, 2013. v. 1. p. 2022-2031. (Qualis B5)

- LOPES, João L. B.; SOUZA, Rodrigo; SOUZA, A.; DAVET, P.; PERNAS, A. M.; YAMIN, Adenauer; GEYER, Cláudio Fernando Resin. Uma Abordagem Autonômica Baseada em Regras para Consciência de Situação na Computação Ubíqua. In: Simpósio em Sistemas Computacionais, 2013, Porto de Galinhas. Simpósio em Sistemas Computacionais, 2013. v. 1. p. 136-143. (Qualis B4)

- LOPES, João L. B.; GUSMAO, Márcia Z.; SOUZA, Rodrigo; DAVET, Patrícia; SOUZA, Alexandre; COSTA, Cristiano; BARBOSA, Jorge; PERNAS, Ana M.; YAMIN, Adenauer; GEYER, Cláudio F. R. Towards a Distributed Architecture for Context-Aware Mobile Applications in UbiComp. In: Brazilian Symposium on Multimedia and the Web, 2013, Salvador. Brazilian Symposium on Multimedia and the Web, 2013. v. 1. p. 1-8. (Qualis B3)

- LOPES, João L. B.; GADOTTI, Gizele I.; DAVET, Patrícia; SOUZA, Alexandre; SOUZA, Rodrigo; PERNAS, Ana M.; YAMIN, Adenauer; GEYER, Cláudio Fernando Resin. Consciência de Situação na Ubicomp: Um Estudo de Caso Aplicado à Análise de Sementes. In: Congresso Brasileiro de Agroinformática, 2013, Cuiabá. IX Congresso Brasileiro de Agroinformática (SBIAgro), 2013. v. 1.

- LOPES, João L. B.; SOUZA, Rodrigo; YAMIN, Adenauer; GEYER, Cláudio Fernando Resin. A Distributed Architecture for Dynamic Context Awareness in UbiComp. In: XI Workshop de Processamento Paralelo e Distribuído, 2013, Porto Alegre. XI Workshop de Processamento Paralelo e Distribuído, 2013.

- LOPES, João L. B.; SOUZA, Rodrigo; GADOTTI, G.; GUSMAO, Márcia Z.; YAMIN, Adenauer Correa; GEYER, Cláudio Fernando Resin. Uma Proposta para Consciência de Contextos Dinâmicos na Ubicomp. In: Seminário Integrado de Software e Hardware, 2012, Curitiba, PR. Anais do Congresso da Sociedade Brasileira de Computação, 2012. (Qualis B4)

- LOPES, João L. B.; SOUZA, Rodrigo; COSTA, Cristiano; BARBOSA, Jorge; GUSMAO, Márcia Z.; GADOTTI, G.; YAMIN, Adenauer Correa; GEYER, Cláudio Fernando Resin. DynamiCC: A Framework for Dynamic Context Composition in Ubicomp. In: Brazilian Symposium on Multimedia and the Web, 2012, São Paulo. Proceedings of the 18th Brazilian symposium on Multimedia and the web. New York, NY, USA: ACM, 2012. p. 169-172. (Qualis B3)

- LOPES, João L. B.; SOUZA, Rodrigo; COSTA, Cristiano; BARBOSA, Jorge; GUSMAO, Márcia Z.; YAMIN, Adenauer Correa; GEYER, Cláudio Fernando Resin. A Model for Context Awareness in Ubicomp. In: Brazilian Symposium on Multimedia and the Web, 2012, São Paulo - Brasil. Proceedings of the 18th Brazilian symposium on Multimedia and the web. New York, NY, USA: ACM, 2012. p. 161-168. (Qualis B3)

- LOPES, João L. B.; SOUZA, Rodrigo; COSTA, Cristiano; BARBOSA, Jorge; GUSMAO, Márcia Z.; GADOTTI, G.; YAMIN, Adenauer Correa; GEYER, Cláudio Fernando Resin. Uma Arquitetura Distribuída para Composição de Contextos Dinâmicos na Ubicomp. In: XXXVIII Conferencia Latinoamericana En Informatica (CLEI), 2012, Medellín - Colômbia. XXXVIII Conferencia Latinoamericana En Informatica (CLEI). Medellín - Colômbia, 2012. p. 1-9. (Qualis B4)

- LOPES, João L. B.; SOUZA, Rodrigo; COSTA, Cristiano; BARBOSA, Jorge; GUSMAO, Márcia Z.; YAMIN, Adenauer C.; GEYER, Cláudio F. R. Managing Adaptation in Ubicomp. In: XXXVIII Conferencia Latinoamericana En Informatica (CLEI), 2012, Medellín - Colômbia. XXXVIII Conferencia Latinoamericana en Informatica (CLEI). Medellín - Colômbia, 2012. p. 1-10. (Qualis B4)

Além destes artigos, outras publicações relacionadas ao trabalho desenvolvido aconteceram em regime de co-autoria.

Periódicos

- SOUZA, Rodrigo; LOPES, João L. B.; GADOTTI, Gisele; PERNAS, Ana M.; YAMIN, Adenauer C.; GEYER, Cláudio F. R. Gerenciamento proativo de redes de sensores na Ubicomp. Revista Brasileira de Computação Aplicada, v. 7, p. 53-64, 2015. (Qualis B5)

- RODRIGUES, Sérgio; VENECIAN, Luthiano; SOUZA, Alexandre; LOPES, João L. B.; YAMIN, Adenauer C.; GEYER, Cláudio F. R. Uma Proposta Baseada em Processamento Semântico para Consciência do Contexto na Medicina Ubíqua. Revista do CCEI, v. 16, p. 295-313, 2012. (Qualis B5)

Congressos

- SCHEUNEMANN, Douglas; LOPES, João L. B.; YAMIN, Adenauer C.; GEYER, Cláudio F. R. Uma Contribuição à Reabilitação Cardíaca Explorando a Identificação de Situações na IoT. In: Seminário Integrado de Software e Hardware, 2016, Porto Alegre, RS, 2016. (Qualis B4)

- SCHEUNEMANN, Douglas; LOPES, João L. B.; YAMIN, Adenauer C.; GEYER, Cláudio F. R. Identificação de Situações de Risco para Pacientes em Reabilitação Cardíaca Explorando uma Arquitetura de Software na Internet das Coisas. In: Workshop de Informática Médica, 2016, Porto Alegre, RS, 2016. (Qualis B4)

- SCHEUNEMANN, Douglas; LOPES, João L. B.; PARREIRA, Wemerson; YAMIN, Adenauer C.; GEYER, Cláudio F. R. Cardiac Rehabilitation: an IoT Architecture Exploring Situation Awareness Based on Open-Source Technologies. In: Simpósio Brasileiro de Telecomunicações e Processamento de Sinais, 2016, Santarém, PA, 2016. (Qualis B4)

- MACHADO, Roger ; ALMEIDA, Ricardo ; ROSA, Diórgenes ; LOPES, João L. B. ; PERNAS, A. M. ; YAMIN, Adenauer Correa . Explorando Modelos Contextuais Híbridos: uma Abordagem Composicional. In: Simpósio Brasileiro de Computação Ubíqua e Pervasiva, 2016, Porto Alegre, RS, 2016. (Qualis B5)

- SOUZA, Rodrigo; LOPES, João L. B.; CARDOZO, Anderson; CARVALHO, T. R.; DAVET, Patrícia; WOLF, Alexandre; BARBOSA, Jorge; YAMIN, Adenauer C.; GEYER, Cláudio F. R. Uma Arquitetura para IoT Direcionada à Ciência do Contexto Baseada em Eventos Distribuídos. In: Simpósio Brasileiro de Computação Ubíqua e Pervasiva, 2016, Porto Alegre, RS, 2016. (Qualis B4)

- SOUZA, Rodrigo; LOPES, João L. B.; GARCIA, Cleiton; DAVET, Patrícia; YAMIN, Adenauer C.; GEYER, Cláudio F. R. Context Awareness in UbiComp: An IoT Oriented Distributed Architecture. In: IEEE International Conference on Electronics, Circuits, and Systems - ICECS, 2015, Cairo. IEEE International Conference on Electronics, Circuits, and Systems, 2015. (Qualis B1)

- SOUZA, Rodrigo; LOPES, João L. B.; SOUZA, Alexandre; DAVET, Patrícia; PERNAS, Ana M.; YAMIN, Adenauer C.; GEYER, Cláudio F. R. Uma Arquitetura Distribuída para Internet das Coisas na Medicina Ubíqua. In: Seminário Integrado de Software e Hardware, 2015, Recife, PE - Brasil. XXXV Congresso da Sociedade Brasileira de Computação. SBC, 2015. (Qualis B4)

- SOUZA, Alexandre; LOPES, João L. B.; JOAO, Leonardo; DAVET, Patrícia; SOUZA, Rodrigo; PERNAS, Ana M.; YAMIN, Adenauer C.; GEYER, Cláudio F. R. Avaliação de Metas Terapêuticas: Uma Proposta de Arquitetura Direcionada à UbiComp. In: Workshop de Informática Médica, 2015, Recife, PE - Brasil. XXXV Congresso da Sociedade Brasileira de Computação. SBC, 2015. (Qualis B4)

- SOUZA, Alexandre; MESQUITTA, Francisco; LOPES, João L. B.; SOUZA, Rodrigo; PERNAS, Ana M.; YAMIN, Adenauer C.; GEYER, Cláudio F. R. Uma Abordagem Ubíqua Consciente de Situação para Avaliação de Metas Terapêuticas em Ambiente Hospitalar. In: Simpósio Brasileiro de Computação Ubíqua e

Pervasiva, 2014, Brasília. Anais do Congresso da Sociedade Brasileira de Computação. Porto Alegre: SBC, 2014. v. 1. p. 921-930. (Qualis B5)

- SOUZA, Rodrigo; LOPES, João L. B.; PERNAS, Ana M. ; GADOTTI, Gizele; YAMIN, Adenauer C.; GEYER, Cláudio F. R. A Contribution to the Sensor Network Management for Context Awareness in Ubicomp. In: Brazilian Symposium on Multimedia and the Web, 2014, João Pessoa. Proceedings of the 20th Brazilian Symposium on Multimedia and the Web, 2014. (Qualis B3)

- PERNAS, Ana M.; MACHADO, Alencar; LOPES, João L. B.; YAMIN, Adenauer; GEYER, Cláudio F. R.; OLIVEIRA, José P. M. Ambientes Educacionais Ubíquos: explorando a consciência de situação. In: Simpósio Brasileiro de Computação Ubíqua e Pervasiva, 2013, Maceió. XXXIII Congresso da Sociedade Brasileira de Computação, 2013. v. 1. p. 1992-2001. (Qualis B5)

6.4 Trabalhos Futuros

A Arquitetura SAUI apresenta diversas oportunidades a serem exploradas. Neste sentido, destacam-se os seguintes temas de pesquisas que podem ser considerados em trabalhos futuros:

- Expandir as funcionalidades do Mecanismo de Raciocínio, em particular, avaliar o uso de ontologias probabilísticas para implementação do processo de predição de situações.

- Dotar o Componente Gerenciador de Regras de uma interface de programação visual com base em fluxo de dados, a exemplo da proposta Node-RED, atualmente empregada por empresas de grande porte, como a plataforma BlueMix da IBM.

- Aprofundar as pesquisas referentes a Qualidade de Contexto, prevendo componentes arquiteturais específicos para o tratamento das informações contextuais.

- Portar os mecanismos de armazenamento e processamento contextual para uma infraestrutura de nuvem, explorando aspectos de escalabilidade, bem como potencializando a disponibilização destes serviços às diferentes células de execução gerenciadas pelo *middleware*.

REFERÊNCIAS

ABDOLI, F.; KAHANI, M. **Ontology-based distributed intrusion detection system**. 2009.

ALEGRE, U.; AUGUSTO, J. C.; CLARK, T. Engineering context-aware systems and applications: a survey. **Journal of Systems and Software**, [S.l.], v.117, p.55 – 83, 2016.

ALLEN, J. F.; FERGUSON, G. Actions and Eventis in Interval Temporal Logic. **Journal of Logic and Computation**, [S.l.], 1994.

ANTONY, J.; ANJU, K.; MATHEW, A. A Tablet PC based System for Ubiquitous Patient Monitoring and Smart Alert Generation in an Intensive Care Unit. **International Journal of Computer Applications**, [S.l.], 2013.

ATZORI, L.; IERA, A.; MORABITO, G. The Internet of Things: a survey. **Computer Networks**, [S.l.], v.54, n.15, p.2787 – 2805, 2010.

AUGUSTO, J. C. et al. Management of Uncertainty and Spatio-Temporal Aspects for Monitoring and Diagnosis in a Smart Home. **International Journal of Computational Intelligence Systems**, [S.l.], 2008.

BALDAUF, M.; DUSTDAR, S.; ROSENBERG, F. A survey on context-aware systems. **Int. J. Ad Hoc Ubiquitous Comput.**, Inderscience Publishers, Geneva, SWITZERLAND, v.2, n.4, p.263–277, June 2007.

BAUER, J. S.; NEWMAN, M. W.; KIENTZ, J. A. Thinking About Context: design practices for information architecture with context-aware systems. In: CONFERENCE 2014 PROCEEDINGS. **Anais...** [S.l.: s.n.], 2014. p.398–411.

BELLAVISTA, P. et al. A survey of context data distribution for mobile ubiquitous systems. **ACM Comput. Surv.**, New York, NY, USA, v.44, n.4, p.24:1–24:45, Sept. 2012.

BETTINI, C. et al. A survey of context modelling and reasoning techniques. **Pervasive Mob. Comput.**, Amsterdam, The Netherlands, The Netherlands, v.6, n.2, p.161–180, Apr. 2010.

BIBRI, S. E. Context Modeling, Representation, and Reasoning: an ontological and hybrid approach. In: **The Human Face of Ambient Intelligence**. [S.l.]: Atlantis Press, 2015. p.197–257. (Atlantis Ambient and Pervasive Intelligence, v.9).

BOUZEGHOUB, A.; DO, K. N.; LECOCQ, C. A situation-based delivery of learning resources in pervasive learning. In: SECOND EUROPEAN CONFERENCE ON TECHNOLOGY ENHANCED LEARNING: CREATING NEW LEARNING EXPERIENCES ON A GLOBAL SCALE, Berlin, Heidelberg. **Proceedings...** Springer-Verlag, 2007. p.450–456. (EC-TEL'07).

BROCK, J. D.; BRUCE, R. F.; CAMERON, M. E. Changing the world with a Raspberry Pi. **Journal of Computing Sciences in Colleges**, [S.l.], v.29, n.2, p.151–153, 2013.

CACERES, R.; FRIDAY, A. Ubicomp Systems at 20: progress, opportunities, and challenges. **Pervasive Computing, IEEE**, [S.l.], v.11, n.1, p.14–21, 2012.

CHEN, G.; KOTZ, D. A Survey of Context-Aware Mobile Computing Research. **Technical Report TR2000-381**, Dept. of Computer Science, Dartmouth College, 2000.

CHEN, L. et al. Semantic Smart Homes: towards knowledge rich assisted living environments. **Intelligent Patient Management**, [S.l.], 2009.

CIMINO, M. G. C. A. et al. An adaptive rule-based approach for managing situation-awareness. **Expert Syst. Appl.**, [S.l.], v.39, n.12, p.10796–10811, 2012.

COSTA, C. A. et al. A Primer of Ubiquitous Computing Challenges and Trends. In: NETO, F. M. M.; NETO, P. F. R. (Ed.). **Designing Solutions-Based Ubiquitous and Pervasive Computing**: new issues and trends. Hershey: IGI Global Publishing, 2010. v.1, p.282–303.

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

COSTA, P. D. et al. A Model-Driven Approach to Situations: situation modeling and rule-based situation detection. In: EDOC. **Anais...** IEEE, 2012. p.154–163.

DA, K.; DALMAU, M.; ROOSE, P. A Survey of adaptation systems. **International Journal on Internet and Distributed Computing Systems**, [S.l.], v.2, n.1, p.1–18, 2011.

DAVIS, F. D. Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. **MIS Quarterly**, [S.l.], v.13, n.3, p.319–340, 1989.

DELICATO, F. C.; PIRES, P. F.; BATISTA, T. V. **Middleware Solutions for the Internet of Things.** [S.l.]: Springer, 2013. i-x, 1-78p. (Springer Briefs in Computer Science).

DEY, A. Context-aware computing: the cyberdesk project. In: SPRING SYMPOSIUM ON INTELLIGENT ENVIRONMENTS. **Anais...** AAAI, 1998.

DEY, A. K. Understanding and Using Context. **Personal and Ubiquitous Computing**, [S.l.], v.5, p.4–7, 2001.

ELEKAR, K.; WAGHMARE, M.; PRIYADARSHI, A. Use of rule base data mining algorithm for intrusion detection. In: PERVASIVE COMPUTING (ICPC), 2015 INTERNATIONAL CONFERENCE ON. **Anais...** [S.l.: s.n.], 2015. p.1–5.

ETZION, O.; NIBLETT, P. **Event Processing in Action**. 1st.ed. Greenwich, CT, USA: Manning Publications Co., 2010.

FENSEL, D.; WAHLSTER, W.; LIEBERMAN, H. (Ed.). **Spinning the Semantic Web**: bringing the world wide web to its full potential. Cambridge, MA, USA: MIT Press, 2005.

FERRY, N. et al. WComp, Middleware for Ubiquitous Computing and System Focused Adaptation. In: CALVARY, G. et al. (Ed.). **Computer Science and Ambient Intelligence**. [S.l.]: ISTE Ltd and Wiley Sons Inc, 2013. v.1, p.89–120.

FIELDING, R. T. **Architectural Styles and the Design of Network-based Software Architectures**. 2000. PhD thesis — University of California, California-USA.

FLEISCHMANN, A. M. P. **Sensibilidade à Situação em Sistemas Educacionais na Web**. 2012. 164p. Tese (Doutorado em Computação) — UFRGS, Porto Alegre, RS.

FLOCH, J. et al. Playing MUSIC: building context-aware and self-adaptive mobile applications. **Software: Practice and Experience**, [S.l.], v.43, n.3, p.359–388, 2013.

FRIESS, O. V. P. **Internet of Things - Global Technological and Societal Trends From Smart Environments and Spaces to Green ICT**. [S.l.]: River Publishers, 2011. (River Publishers Series in Communications).

GALLACHER, S. et al. Dynamic context-aware personalisation in a pervasive environment. **Pervasive and Mobile Computing**, [S.l.], v.10, Part B, p.120 – 137, 2014.

GUBBI, J. et al. Internet of Things (IoT): a vision, architectural elements, and future directions. **Future Generation Computer Systems**, [S.l.], v.29, n.7, p.1645 – 1660, 2013.

GUBBI, J. et al. Internet of Things (IoT): a vision, architectural elements, and future directions. **Future Generation Computer Systems**, [S.l.], v.29, n.7, p.1645–1660, 2013.

GUESGEN, H.; MARSLAND, S. Spatio-Temporal Reasoning and Context Awareness. In: NAKASHIMA, H.; AGHAJAN, H.; AUGUSTO, J. (Ed.). **Handbook of Ambient Intelligence and Smart Environments**. [S.l.]: Springer US, 2010. p.609–634.

HORA, H. R. M. d.; MONTEIRO, G. T. R.; ARICA, J. Confiabilidade em Questionários para Qualidade: um estudo com o coeficiente alfa de cronbach. **Produto e Produção**, [S.l.], v.11, n.2, p.85–103, 2010.

HORRIDGE, M.; BECHHOFER, S. The OWL API: a java api for owl ontologies. **Semant. web**, Amsterdam, The Netherlands, The Netherlands, v.2, n.1, p.11–21, Jan. 2011.

HUEBSCHER, M. C.; MCCANN, J. A.; HOSKINS, A. Context as autonomic intelligence in a ubiquitous computing environment. **Int. J. Internet Protoc. Technol.**, Inderscience Publishers, Geneva, SWITZERLAND, v.2, n.1, p.30–39, Dec. 2007.

HULL, R.; NEAVES, P.; BEDFORD-ROBERTS, J. Towards situated computing. In: INTERNATIONAL SYMPOSIUM ON WEARABLE COMPUTERS. **Anais...** ., 1997.

ISAZA, G. A. et al. **Towards Ontology-Based Intelligent Model for Intrusion Detection and Prevention**. 2010.

JENA, A. **Apache Jena - Java framework for building Semantic Web and Linked Data applications**. Disponível em: <http://jena.apache.org/>. Acesso em março de 2016.

JESS. **Jess - The Rule Engine for the Java Platform**. Disponível em: < http://www.jessrules.com/jess/index.shtml >. Acesso em novembro de 2016.

KAKOUSIS, K.; PASPALLIS, N.; PAPADOPOULOS, G. A. A survey of software adaptation in mobile and ubiquitous computing. **Enterprise Information Systems**, [S.l.], v.4, n.4, p.355–389, 2010.

KHAIRKAR, A. D. **Intrusion Detection System based on Ontology for Web Applications**. 2013. Master of Technology, Computer Engineering — Department of Computer Engineering and Information Technology College of Engineering, Pune.

KHARGHARIA, B.; HARIRI, S.; YOUSIF, M. S. Autonomic power and performance management for computing systems. **Cluster Computing**, [S.l.], v.11, n.2, p.167–181, 2008.

KNAPPMEYER, M. et al. Survey of Context Provisioning Middleware. **Communications Surveys Tutorials, IEEE**, [S.l.], v.15, n.3, p.1492–1519, 2013.

KNUBLAUCH, H. et al. **OWL Pizzas-Practical Experience of Teaching OWL-DL**: common errors & common patterns. [S.l.]: Springer, 2004. 63–81p. (Lecture Notes in Computer Science, v.3257).

KRUMM, J. **Ubiquitous Computing Fundamentals**. 1st.ed. [S.l.]: Chapman & Hall/CRC, 2010.

KWAPISZ, J. R.; WEISS, G. M.; MOORE, S. a. Activity recognition using cell phone accelerometers. **ACM SIGKDD Explorations Newsletter**, [S.l.], v.12, p.74, 2011.

LALANDA, P.; MCCANN, J. A.; DIACONESCU, A. **Autonomic Computing. Principles, Design and Implementation**. 1st.ed. [S.l.]: Springer-Verlag London, 2013.

LARA, O. D.; LABRADOR, M. a. A Survey on Human Activity Recognition using Wearable Sensors. **IEEE Communications Surveys & Tutorials**, [S.l.], p.1–18, 2012.

LEE, W.; STOLFO, S.; MOK, K. A data mining framework for building intrusion detection models. In: SECURITY AND PRIVACY, 1999. PROCEEDINGS OF THE 1999 IEEE SYMPOSIUM ON. **Anais...** [S.l.: s.n.], 1999. p.120–132.

LI, S.; XU, L.; ZHAO, S. The internet of things: a survey. **Information Systems Frontiers**, [S.l.], v.17, n.2, p.243–259, 2015.

LI, X. et al. Context Aware Middleware Architectures: survey and challenges. **Sensors**, [S.l.], v.15, n.8, p.20570, 2015.

LOKE, S. W. Incremental awareness and compositionality: a design philosophy for context-aware pervasive systems. **Pervasive and Mobile Computing**, [S.l.], 2010.

LOPES, J. et al. A Middleware Architecture for Dynamic Adaptation in Ubiquitous Computing. **Journal of Universal Computer Science**, [S.l.], v.20, n.9, p.1327–1351, sep 2014.

LOPES, J. et al. A Distributed Architecture for Supporting Context-Aware Applications in UbiComp. In: ADVANCED INFORMATION NETWORKING AND APPLICATIONS (AINA), 2014 IEEE 28TH INTERNATIONAL CONFERENCE ON. **Anais...** [S.l.: s.n.], 2014. p.584–590.

LOPES, J. et al. A Situation-aware Pervasive Approach for Assessing Therapeutic Goals in Healthcare Environment. In: ANNUAL ACM SYMPOSIUM ON APPLIED COMPUTING, 31., New York, NY, USA. **Proceedings...** ACM, 2016. p.125–130. (SAC '16).

LOPES, J. et al. Middleware Architecture for Supporting a Hybrid Processing of Context Data Targeted to Detection of Situations in UbiComp. In: INTERNATIONAL FLINS CONFERENCE ON UNCERTAINTY MODELING IN KNOWLEDGE ENGINEERING AND DECISION MAKING, 12., Singapore. **Proceedings...** World Scientific, 2016.

LOUREIRO, A. A. F. et al. Computação Ubíqua Ciente de Contexto: desafios e tendências. In: SIMPóSIO BRASILEIRO DE REDES DE COMPUTADORES E SISTEMAS DISTRIBUíDOS - SBRC 2009 - MINICURSO, 27. **Anais...** [S.l.: s.n.], 2009. p.99–149.

MAXIM. **Maxim Integrated**. 2016.

MIORANDI, D. et al. Internet of things: vision, applications and research challenges. **Ad Hoc Networks**, [S.l.], v.10, n.7, p.1497 – 1516, 2012.

NAZARIO, D. C. et al. Toward assessing Quality of Context parameters in a ubiquitous assisted environment. In: IEEE SYMPOSIUM ON COMPUTERS AND COMMUNICATIONS (ISCC), 2014. **Anais...** [S.l.: s.n.], 2014. p.1–6.

NEGRÃO, C. E.; BARRETO, A. C. P. **Cardiologia do Exercício: do atleta ao cardiopata**. 3.ed. Barueri, SP - Brazil: Manole, 2010. 725p.

PEREIRA, I.; COSTA, P.; ALMEIDA, J. A rule-based platform for situation management. In: COGNITIVE METHODS IN SITUATION AWARENESS AND DECISION SUPPORT (COGSIMA), 2013 IEEE INTERNATIONAL MULTI-DISCIPLINARY CONFERENCE ON. **Anais...** [S.l.: s.n.], 2013. p.83–90.

PERERA, C. et al. Context aware computing for the internet of things: a survey. **IEEE Communications Surveys and Tutorials**, [S.l.], v.16, n.1, p.414 – 454, 2014.

PERTTUNEN, M.; RIEKKI, J.; LASSILA, O. Context representation and reasoning in pervasive computing: a review. **International Journal of Multimedia and Ubiquitous Engineering**, [S.l.], p.1–28, 2009.

PIRES, P. F. et al. A platform for integrating physical devices in the Internet of Things. **Proceedings of the 12th IEEE International Conference on Embedded and Ubiquitous Computing**, USA, p.234–241, 2014.

RABELO, D.; GIL, C.; ARAÚJO, S. D. Reabilitação cardíaca com ênfase no exercício: uma revisão sistemática. **Revista Brasileira de Medicina do Esporte**, [S.l.], v.12, n.5, p.279–285, 2006.

RAITOHARJU, R. **Information Technology Acceptance in the Finnish Social and Healthcare Sector**: exploring the effects of cultural factors. [S.l.]: Turku School of Economics, 2007. (Turun kauppakorkeakoulun julkaisuja).

RANDELL, D. A.; CUI, Z.; COHN, A. G. A Spatial Logic based on Regions and Connection. **3rd Int. Conf. on Knowledge Representation and Reasoning**, [S.l.], 1992.

RAYCHOUDHURY, V. et al. Middleware for pervasive computing: a survey. **Pervasive and Mobile Computing**, [S.l.], v.9, n.2, p.177 – 200, 2013. Special Section: Mobile Interactions with the Real World.

RAZZAQUE, M. A. et al. Middleware for Internet of Things: a survey. **Internet of Things Journal, IEEE**, [S.l.], v.3, n.1, p.70–95, 2016.

RYAN, N.; PASCOE, J.; MORSE, D. Enhanced reality fieldwork: the contextaware archaeological assistant. **Computer Applications in Archaeology**, [S.l.], 1997.

SATYANARAYANAN, M. Mobile Computing: the next decade. In: ACM WORKSHOP ON MOBILE CLOUD COMPUTING & SERVICES: SOCIAL NETWORKS AND BEYOND, 1., New York, NY, USA. **Proceedings...** ACM, 2010. p.5:1–5:6. (MCS '10).

SCHILIT, B.; THEIMER, M. Disseminating Active Map Information to Mobile Hosts. **IEEE Network**, [S.l.], 1994.

SEABORNE, A. **RDQL - A Query Language for RDF**. Disponível em: <http://www.w3.org/Submission/RDQL/>. Acesso em novembro de 2016.

SEHILI, S. et al. IoT Efficient Design Using WComp Framework and Discrete Event Modeling and Simulation. In: SIMULATION AND MODELING METHODOLOGIES, TECHNOLOGIES AND APPLICATIONS: INTERNATIONAL CONFERENCE. **Anais...** Springer International Publishing, 2016. p.49–67.

BLUCHER (Ed.). **Controle e Modelagem Fuzzy**. 2.ed. [S.l.: s.n.], 2007. 200p.

SILVA, T. et al. Overview of Ubicomp Research Based on Scientic Publications. In:. In: SIMPóSIO BRASILEIRO DE COMPUTAçãO UBíQUA E PERVASIVA, Curitiba, PR. **Anais...** SBC, 2012.

SIRIN, E. et al. Pellet: a practical owl-dl reasoner. **Web Semantics: Science, Services and Agents on the World Wide Web**, Amsterdam, The Netherlands, v.5, n.2, p.51–53, June 2007.

SOBREVILLA, P.; MONTSENY, E. Fuzzy Sets in Computer Vision : an overview. **Mathw. Soft Computing**, [S.l.], v.10, p.71–83, 2003.

SPARQL. **SPARQL 1.1 Overview - W3C Recommendation**. Disponível em: <https://www.w3.org/TR/sparql11-overview//>. Acesso em março de 2016.

UNDERCOFFER, J.; JOSHI, A.; PINKSTON, J. Modeling Computer Attacks: an ontology for intrusion detection. In: VIGNA, G.; KRUEGEL, C.; JONSSON, E. (Ed.). **Recent Advances in Intrusion Detection**. [S.l.]: Springer Berlin Heidelberg, 2003. p.113–135. (Lecture Notes in Computer Science, v.2820).

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

WHITMORE, A.; AGARWAL, A.; DA XU, L. The Internet of Things - A survey of topics and trends. **Information Systems Frontiers**, [S.l.], v.17, n.2, p.261–274, 2015.

YAMIN, A. **Arquitetura para um Ambiente de Grade Computacional Direcionado às Aplicações Distribuídas, Móveis e Conscientes do Contexto da Computação Pervasiva**. 2004. 195p. Tese (Doutorado em Ciência da Computação) — Instituto de Informática, UFRGS, Porto Alegre, RS.

YAMIN, A. et al. Towards Merging Context-Aware, Mobile and Grid Computing. **International Journal of High Performance Computing Applications**, Londres, v.17, n.2, p.191–203, 2003.

YE, J.; DOBSON, S.; MCKEEVER, S. Review: situation identification techniques in pervasive computing: a review. **Pervasive Mob. Comput.**, Amsterdam, The Netherlands, The Netherlands, v.8, n.1, p.36–66, Feb. 2012.

YE, J. et al. Semantic web technologies in pervasive computing: a survey and research roadmap. **Pervasive and Mobile Computing**, [S.l.], v.23, p.1 – 25, 2015.

YOON, C.; KIM, S. Convenience and TAM in a ubiquitous computing environment: the case of wireless lan. **Electron. Commer. Rec. Appl.**, Amsterdam, The Netherlands, The Netherlands, v.6, n.1, p.102–112, Jan. 2007.

APÊNDICE A ESTILO ARQUITETURAL REST

REST (*Representational State Transfer*) é um termo que foi utilizado pela primeira vez por FIELDING (2000). Como um estilo arquitetural de *software* o REST não descreve protocolos específicos, formatos de dados ou sequências de interação, mas sim apresenta os princípios de arquitetura e componentes que levaram a ampla utilização de sistemas distribuídos, como o da *World Wide Web*.

O REST trabalha em conjunto com o protocolo HTTP, embora seja possível desenvolver um sistema de software baseado nas definições do estilo arquitetural REST sem usar HTTP e sem interagir com a Web. Os sistemas que seguem os princípios REST são referenciados como "*RESTful*".

O estilo arquitetural REST é construído sobre determinados princípios básicos da Internet. Existem cinco princípios que devem ser utilizados para a criação de serviços REST:

- Tudo é recurso: um recurso é qualquer componente de uma aplicação que pode ser unicamente identificável e que possa ser transmitido na rede através das suas representações (binárias ou textuais). Em REST o pensamento não está focado em arquivos, mas sim em recursos;

- Todos os recursos são identificados através de um identificador único: os identificadores são utilizados para permitir a fácil manipulação do recurso. Em REST utilizam-se *Uniform Resource Identifiers* (URIs) para identificar recursos;

- Utilização de uma interface simples e uniforme: REST utiliza HTTP como protocolo base, sendo este um protocolo bem conhecido e aceito;

- Comunicação efetuada através de representações: cada recurso pode possuir várias representações diferentes. Quando se realiza uma determinada operação sobre um recurso, o que acontece na realidade é uma troca de representações desse recurso;

- Interações *stateless* (sem estado): todas as interações realizadas sobre um recurso são independentes, isto é, a comunicação deve ser feita sem o armazenamento de qualquer tipo de estado no servidor, ou seja, cada requisição do cliente para o servidor deve conter todas as informações necessárias para que ela seja entendida. Portanto, estados de sessão, quando necessários, devem ser totalmente mantidos no cliente.

Um dos pontos chave deste estilo arquitetural tem a ver com a utilização de URIs para identificação de recursos na Web. Segundo este estilo, os serviços são abstraídos através de uma interface uniforme (HTTP e respectivos métodos) que fornece mecanismos para

que as aplicações cliente possam escolher a melhor representação possível das respostas que recebem do serviço. Estes são alguns dos motivos que posicionam o estilo REST no cenário da Internet das Coisas para o desenvolvimento de uma arquitetura e APIs para dispositivos inteligentes.

O protocolo de comunicação base associado ao estilo REST é, como referido anteriormente, o protocolo HTTP. A arquitetura base deste protocolo segue o modelo de comunicação cliente-servidor. Este modelo é amplamente utilizado na Internet e de fácil compreensão: um cliente que deseje consultar um determinado recurso envia ao servidor um pedido HTTP, ao qual o servidor responde enviando uma mensagem ao cliente com a respectiva resposta.

Em relação ao formato da resposta enviada pelo servidor ao cliente e uma vez que os dispositivos inteligentes possuem recursos limitados, a informação pode ser enviada ao cliente tomando a forma de um documento estruturado XML ou ainda a forma de um documento JSON.

Por sua vez, caso um usuário final queira consultar informações relacionadas a um determinado recurso, é preferível obter essa informação, por exemplo, através de um *browser* que renderize a resposta dada pelo servidor na forma de uma página Web. Uma vantagem inerente a estas várias representações reside na possibilidade de poderem ser interpretadas e processadas por máquinas e ainda por pessoas permitindo uma maior flexibilidade e eficiência no processo de comunicação.

Dados dinâmicos acerca do mundo real podem ser apresentados em páginas Web e depois processados com o auxílio de ferramentas Web 2.0. Por exemplo, os dispositivos podem ser indexados como páginas Web através das suas representações para que os usuários possam acessar a estas páginas através de um *browser* com o auxílio de um motor de busca. Estes endereços podem também ser enviados para outros usuários (através de *e-mail* por exemplo) além de também poderem ser adicionados aos favoritos do *browser* pelo usuário final. A ideia base passa pela utilização da Web como sistema de informação descentralizado para que seja fácil expor novos serviços e aplicações, direta ou indiretamente por dispositivos inteligentes.

A ideia central de REST está intimamente relacionada com a noção de recurso. Cada recurso possui um identificador global (URI em HTTP). Estes recursos são acessados por componentes na rede que comunicam através de um protocolo (ex. HTTP) e trocam conteúdo (representações) desses recursos. Para interagir com um recurso, uma aplicação tem que possuir o identificador do recurso bem como o método que quer aplicar sobre o mesmo.

Portanto, no REST não é necessário conhecer a implementação e a configuração do sistema ou seja, não interessa saber se existem *proxies, firewalls* ou qualquer outro componente entre a aplicação e o servidor que aloja esses recursos. A aplicação deve ser capaz de interpretar os dados enviados como resposta pelo servidor, sejam eles texto simples, XML, imagens, documentos, etc. Para isso, é possível a qualquer cliente especificar no pedido qual o formato da representação que deseja receber por parte do servidor.

De forma simples, o primeiro passo para permitir ligar dispositivos inteligentes na web passa por criar a rede de recursos. Dois aspectos centrais são o identificador de um recurso e as suas relações com outros recursos.

Em REST, a interação com os recursos e a recepção das representações por parte dos clientes acontece através de uma interface uniforme que especifica um contrato de serviço entre clientes e servidores. Existem três partes distintas e fundamentais para que se realize

esta interação: operações, negociação de conteúdo e estado da resposta.

Quanto às operações, REST utiliza quatro métodos principais do protocolo HTTP para permitir a interação com recursos: GET, PUT, POST e DELETE. O método GET é utilizado para obtenção de representações de recursos. O método PUT serve para atualizar o estado de um recurso ou para criar um recurso utilizando um identificador. O método POST cria um novo recurso sem ser necessário especificar o identificador. Por fim, o método DELETE serve para remover um recurso. Um exemplo do uso das operações através de métodos HTTP identificadas por URIs pode ser visto na Tabela A.1.

Tabela A.1: Exemplo de Interações com Recursos Através de Métodos HTTP

URI	Método HTTP	Descrição
/user	GET	Lista de usuários.
/user/id	GET	Recupera um usuário com todas as suas informações.
/user/id	PUT	Altera os dados do usuário informado.
/user/id	DELETE	Remove o usuário informado.
/user/	POST	Cria um novo usuário.

Em relação à negociação de conteúdo, ou seja, o mecanismo que permite aos clientes e servidores comunicarem para chegarem a um acordo sobre qual a representação a ser devolvida pelo servidor, esta negociação está embutida na interface HTTP subjacente à arquitetura REST. No que diz respeito ao estado da resposta, são utilizados os códigos *standard* do protocolo HTTP.

A flexibilidade e facilidade de compreensão do estilo arquitetural REST, torna-o um estilo atrativo e de ampla utilização a nível mundial nos mais variados cenários, inclusive o da IoT.

APÊNDICE B PROCESSAMENTO SEMÂNTICO DAS INFORMAÇÕES DE CONTEXTO

O processamento semântico, na perspectiva dos sistemas computacionais, requer a utilização de padrões que sejam interpretáveis por máquina. Nesse sentido, o suporte ao processamento semântico das informações de contexto pode ser construído com o uso de padrões da Web Semântica. Esses padrões propostos pela W3C (*World Wide Web Consortium*) são organizados em uma arquitetura em camadas.

A primeira camada serve de base para a Web Semântica, abrangendo o conjunto de caracteres Unicode e a URI que é utilizada para referenciar os recursos na rede. Na segunda camada, a linguagem de marcação XML associada aos recursos de *NameSpaces* e XML *Schema* servem para descrever documentos que podem ser compreendidos por pessoas e software. A partir deste ponto, todas as camadas seguintes estão estruturadas sobre a camada XML e tiram proveito dela. Assim, a estrutura seguinte (RDF + RDF *Schema*) é uma camada de metadados que é construída sobre a de XML e que visa a interoperabilidade entre as aplicações, representando os dados sob a forma de triplas através de descrições de recursos, propriedades e valor.

A próxima estrutura corresponde à camada de ontologias, sendo de particular interesse para o processamento das informações de contexto, já que as ontologias vêm se constituindo na principal abordagem para modelagem semântica do contexto. Essa camada define um vocabulário comum para que o conteúdo semântico possa ser compartilhado entre as aplicações. Neste nível, a linguagem OWL, usada para representar as ontologias, possibilita agregar maior conteúdo semântico, utilizando recursos do RDF e do XML das camadas inferiores.

As últimas camadas, baseadas na estrutura já existente na parte inferior da arquitetura, implementam a definição de regras e possibilitam a prova e validação de inferências realizadas por agentes de software que fazem uso dos recursos de toda a estrutura existente para a representação do conhecimento.

Ontologias

As ontologias têm sido largamente utilizadas em diversas áreas da Ciência da Computação. Particularmente, vêm sendo empregadas para lidar com alguns dos principais desafios relacionados à construção de ambientes ubíquos. De modo geral, ontologias têm sido usadas para representar ambientes ubíquos, descrevendo as entidades envolvidas e suas respectivas propriedades.

As aplicações em um ambiente ubíquo necessitam ser cientes do contexto, assim elas podem adaptar-se rapidamente às mudanças de situações. Aplicações em um

ambiente ubíquo utilizam diferentes tipos de contexto, por exemplo: localização das pessoas, tarefas individuais ou em grupo, informações sobre tempo. Os diversos tipos de informações de contexto que podem ser utilizados precisam estar muito bem definidos, assim as diferentes entidades componentes do ambiente ubíquo terão um entendimento comum do contexto. Também, precisam atuar como mecanismos para que os usuários possam especificar como as diferentes aplicações e serviços devem comportar-se em diferentes contextos. Portanto, estes mecanismos precisam ser baseados em estruturas muito bem definidas, já que existem diferentes tipos de informações de contexto que podem ocorrer em um ambiente ubíquo. Nesse sentido, as ontologias podem ser aplicadas com sucesso para esta tarefa, definindo descrições padronizadas para os diversos tipos de informações de contexto relevantes.

A definição mais aceita e citada pelos autores da área de Computação é a que caracteriza ontologia como uma especificação formal e explícita de uma conceituação compartilhada (FENSEL; WAHLSTER; LIEBERMAN, 2005). Conceituação se refere a modelagem abstrata do mundo real. Explícita significa que os conceitos e seus requisitos são definidos explicitamente, definições de conceitos, instâncias, relações, restrições e axiomas. Formal indica que a ontologia é processável por máquina, permite raciocínio automático e possui semântica formal. Compartilhada significa que uma ontologia captura o conhecimento apresentado não apenas por um único indivíduo, mas por um grupo, sendo uma terminologia comum da área modelada ou acordada entre os desenvolvedores.

Uma ontologia não se resume somente a um vocabulário, também possui relacionamentos e restrições (axiomas) entre os conceitos definidos pelo vocabulário e, através de regras de inferência, é possível derivar novos fatos baseando-se em fatos existentes.

Entre as potencialidades apresentadas pelo uso de ontologias, destacam-se (KNUBLAUCH et al., 2004): (i) são otimizadas para a representação de conhecimento estruturado em um nível elevado de abstração; (ii) os modelos de domínio podem ser disponibilizados e compartilhados entre múltiplas aplicações; (iii) é baseada em dialetos não ambíguos da lógica formal, permitindo a utilização de serviços inteligentes baseados no raciocínio, possibilitando em tempo de execução, a definição dinâmica de classes, a reclassificação de instâncias e a execução de consultas lógicas complexas.

Para definir e instanciar ontologias, o W3C recomendada como padrão a linguagem OWL. Essa linguagem foi projetada para disponibilizar uma forma comum para o processamento de conteúdo semântico da informação. Ela foi desenvolvida para aumentar a facilidade de expressar semântica disponível em XML, RDF e RDFs. Consequentemente, pode ser considerada uma evolução destas linguagens em termos de sua habilidade de representar conteúdo semântico interpretável por máquinas. Sendo a OWL baseada em XML, a informação pode ser facilmente trocada entre diferentes tipos de computadores usando diferentes sistemas operacionais e linguagens de programação.

A OWL tem três sub-linguagens que possuem um nível crescente de expressividade: (i) OWL Lite, dá suporte aos usuários que necessitam principalmente de uma classificação hierárquica e restrições simples. (ii) OWL DL, viabiliza suporte a usuários que querem a máxima expressividade, mantendo a computabilidade (garante que todas as conclusões sejam computáveis) e a decidibilidade (todas as computações terminarão em tempo finito). OWL DL é assim denominada devido a sua correspondência com a lógica de descrição; (iii) OWL Full, é direcionada aos usuários que querem a máxima expressividade e a liberdade sintática do RDF sem nenhuma garantia computacional.

Pesquisa e Inferência sobre o Contexto

O emprego de uma abordagem semântica com o uso ontologias para representar e processar o contexto viabiliza a realização de pesquisas e inferências sobre as informações contextuais. A pesquisa trata somente da extração de dados da base ontológica, sem realizar qualquer tipo de dedução de novos fatos a partir dos existentes, o que é pertinente apenas ao processo de inferência.

Para o processo de pesquisa sobre as informações de contexto, duas linguagens de consulta se destacam: SPARQL e RDQL (*RDF Data Query Language*) (SEABORNE, 2007). Ambas linguagens possuem uma sintaxe similar a SQL. Também, são linguagens orientadas a dados, permitindo apenas extrair dados de documentos RDF, disponíveis em rede ou armazenados em um meio físico qualquer, não possuindo capacidade de inferência. Considera-se a SPARQL como uma evolução da RDQL, sendo recomendada pelo W3C como padrão para realização de consultas em ontologias descritas em linguagens como OWL, que tem como base o RDF.

O processo de inferência pode ser baseado em ontologias ou em regras. Em um processo de inferência baseado em ontologias, inferem-se informações de contexto a partir da combinação semântica definida pelos construtores da linguagem em que uma ontologia é especificada e de um conjunto de fatos instanciados na ontologia. Por sua vez, o processo de inferência baseado em regras utiliza uma sintaxe de construção de regras, de acordo com o mecanismo de inferência que estiver sendo utilizado. Essa sintaxe incorpora construtores específicos que podem ser utilizados como predicados e os demais predicados correspondem a termos do vocabulário definido na ontologia em relação a qual as regras estão sendo aplicadas.

Um raciocinador (*reasoner*) é um componente chave para trabalhar com inferência sobre o contexto modelado com ontologias, pois o conhecimento em uma ontologia pode não ser explícito e, nesse caso, um *reasoner* é requerido para deduzir conhecimento implícito. Nesse sentido, ambos processos de inferência necessitam empregar *reasoners* para sua execução, tais como: (i) Pellet (SIRIN et al., 2007) é um *reasoner* para inferência sobre ontologias descritas em OWL-DL; (ii) JESS (JESS, 2013) é um *engine* para processar regras; (iii) *Reasoners* da API Jena abrangem raciocinadores baseados tanto em ontologias como em regras.

Implementação do Processamento Semântico do Contexto

A implementação de uma abordagem semântica, baseada em ontologias, para o processamento das informações de contexto requer o suporte de ferramentas para criação e edição de ontologias, raciocínio sobre ontologias, bem como uso de ontologias em aplicações. De modo geral, essas ferramentas necessitam algum tipo de API que permita que as ontologias sejam lidas, manipuladas e consultadas.

API Jena

A Jena é uma API desenvolvida na linguagem de programação Java para construção de aplicações baseada em abordagem semântica. Inicialmente, a Jena foi desenvolvida nos laboratórios de pesquisa da HP (*Hewlett-Packard*) e posteriormente disponibilizada como projeto de software livre.

A API Jena oferece vários recursos para o processamento semântico, tais como: (i) mecanismos para manipulação de modelos RDF em memória, bases de dados

relacionais e arquivos; (ii) suporte às linguagens de consulta de dados RDF, como SPARQL e RDQL; (iii) um conjunto de APIs para manipulação de ontologias codificadas em OWL; (iv) motores de inferência baseados em ontologias e regras, bem como um mecanismo que permite integrar *reasoners* de terceiros.

A Jena é constituída por três módulos base destinados ao processamento de ontologias. O primeiro módulo é o *Ontology Model*, que contém todas as classes necessárias para trabalhar com ontologias descritas em OWL, DAML, OIL ou RDFS.

O outro módulo presente na arquitetura é o *Reasoner*. Este permite fazer inferências sobre modelos OWL. O uso das inferências sobre modelos semânticos permite obter informação adicional (inferida) sobre as ontologias. A arquitetura do *Reasoner* possibilita que novos motores de inferência externos venham a ser adicionados.

Os *reasoners* de OWL próprios da API Jena funcionam aplicando regras sobre instâncias da ontologia. Outro *reasoner* disponível é o de uso geral, denominado *Generic Rule Reasoner*. Este permite inferência sobre grafos RDF através de regras que podem ser definidas externamente. Estas regras são formadas basicamente por uma lista de termos de corpo (premissas), uma lista de termos de cabeça (conclusões) e um nome opcional que identifica a regra. Cada termo pode ser uma tripla padrão, uma tripla estendida padrão ou uma chamada a uma primitiva embutida.

O OWL está definido sobre RDFS, assim a Jena utiliza suas APIs de RDF para manipular as ontologias. Disto deriva a necessidade do terceiro módulo (*Base RDF Graph*).

API OWL

A API OWL (HORRIDGE; BECHHOFER, 2011) é uma API Java para criação e manipulação de ontologias especificadas em OWL. A API está fortemente alinhada com a especificação da OWL 2, suportando o modelo orientado a axiomas sobre o qual essa versão da OWL está baseada. Ela inclui interfaces para *reasoner* de propósito geral, validadores para diversos perfis de OWL 2 e suporte para análise e serialização de ontologias em diversas sintaxes. O projeto da API também possibilita a utilização de soluções alternativas desenvolvidas por terceiros.

Em seu núcleo, a API OWL consiste de um conjunto de interfaces para inspeção, manipulação e raciocínio com ontologias OWL. A API suporta leitura e gravação de ontologias em diversas sintaxes. Entretanto, nenhuma das interfaces reflete ou tende para qualquer modelo ou sintaxe particular. Diferente de outras APIs, como Jena, a representação de classes e axiomas não ocorre no nível de triplas RDF. Ao invés disso, o projeto da API OWL está diretamente baseado na especificação estrutural da OWL 2. Isso significa que uma ontologia é vista como um conjunto de axiomas e anotações. Os nomes e hierarquias das interfaces das entidades, classes e axiomas na OWL API correspondem exatamente à especificação estrutural da OWL.

Ainda, a API OWL inclui várias interfaces para acessar *reasoners* OWL, por exemplo, o Pellet, com o intuito de verificar a consistência de ontologias, computar hierarquias de classes e propriedades e verificar se os axiomas estão definidos pela ontologia.