Consciência de Contexto na UbiComp: Fundamentos e Revisão de Estratégias de Modelagem

Roger da Silva Machado, UFPel; Patrícia Teixeira Davet, UFPel

Resumo—Com a disseminação das redes de computadores, e o aumento do emprego de dispositivos portáteis de elevado poder computacional e amplo espectro de uso, teve início a terceira era na cadeia evolutiva da computação moderna, a Computação Ubíqua. Uma das principais características da computação ubíqua é a consciência de contexto, a qual pode ser dividida em três etapas, sendo elas, aquisição, modelagem e processamento. A sinergia entre estas três etapas potencializa a utilização do contexto por parte das aplicações. Entre outros aspectos, o emprego de contexto proporciona uma melhor interação do usuário com a infraestrutura computacional. Este trabalho realiza um levantamento sobre as diferentes estratégias utilizadas nas etapas empregadas para provisão da consciência de contexto, com destaque as técnicas de modelagem. Neste sentido, é realizada uma análise de projetos em consciência de contexto, comparando os métodos empregados nas diferentes etapas.

Index Terms—Computação Ubíqua, Consciência de Contexto, Modelagem de Contexto.

1 Introdução

Com o passar dos anos, os computadores tornaram-se cada vez mais presentes nas tarefas cotidianas e mais próximos dos usuários. Por volta da década de 40 teve início à primeira era da computação com o surgimento dos *mainframes*, computadores que geralmente ocupavam um grande espaço e possuíam acesso restrito. Nas décadas de 70 e 80 surge à segunda era da computação com o advento dos primeiros computadores pessoais (PCs), possibilitando uma maior popularização dos computadores e aproximação com os usuários, além da realização de tarefas gerais como processamento de texto, navegação na Internet, jogos e programação.

Entretanto, foi com a popularização da Internet e dos ambientes distribuídos que os computadores passaram a se interconectar, compartilhando recursos e informações, ampliando os serviços oferecidos e a complexidade das aplicações desenvolvidas, levando a computação para a terceira era da computação moderna, a era da Computação Ubíqua ou UbiComp. Esta é a fase dos dias atuais, onde há uma variedade de dispositivos eletrônicos com diferentes perfis computacionais, que fornecem conjuntamente serviços para o usuário [1].

O paradigma da computação ubíqua tem como premissa prover computação de forma transparente, estando o modelo computacional integrado as demandas do usuário [2]. Ao se construir e executar aplicações ubíquas conscientes de contexto há uma série de funcionalidades que devem ser providas, envolvendo desde a aquisição de informações contextuais, a partir do conjunto de fontes heterogêneas e distribuídas, até a representação dessas informações, seu processamento, armazenamento, e a realização de inferências para seu uso em tomadas de decisões [3].

Conhecer o contexto em que ocorre uma interação, embora constitua uma tarefa usual para os seres humanos, mostra-se uma tarefa complexa em sistemas computacionais. A construção de sistemas conscientes de contexto não é uma tarefa trivial, onde inicialmente, é preciso definir o que considerar como contexto, onde este se aplica e que informações são necessárias para descrevê-lo. É preciso viabilizar formas de realizar a aquisição do contexto o mais automática possível. Após a coleta de dados brutos a partir do sensoriamento,

- ***Roger da Silva Machado**: Programa de Pós-Graduação em Computação, Universidade Federal de Pelotas - UFPel, Centro de Desenvolvimento Tecnológico - CDTec.*
E-mail: rdsmachado@inf.ufpel.edu.br

- ***Patrícia Teixeira Davet**: Programa de Pós-Graduação em Computação, Universidade Federal de Pelotas - UFPel, Centro de Desenvolvimento Tecnológico - CDTec.*
E-mail: ptdavet@inf.ufpel.edu.br

tornam-se necessários mecanismos de processamento de contexto (raciocínio, inferência, etc.) que tratem as informações coletadas, produzindo informações contextualizadas.

O processamento das informações contextuais coletadas prescinde que as mesmas sejam organizadas segundo um modelo. Há diversas estratégias para modelagem de contexto presentes na literatura, cada uma considerando cenários específicos de uso. Assim, a compreensão da natureza da aplicação é essencial para escolher ou projetar o modelo mais apropriado para realizar a etapa de modelagem de contexto. Por outro lado, a existência de modelos bem definidos é oportuna para a verificação da consistência dos dados contextuais coletados.

Neste trabalho, optou-se por realizar uma sumarização das principais técnicas utilizadas no gerenciamento das informações contextuais, mostrando as características das técnicas utilizadas em cada etapa presente na consciência de contexto, como também vantagens e desvantagens em sua utilização. E ainda são apresentados trabalhos que utilizam diferentes tipos de estratégias para realizar a modelagem de contexto.

Este artigo está organizado da seguinte forma: primeiramente, na seção 2, são apresentadas as principais características da Computação Ubíqua. A seção 3 apresenta os principais aspectos presentes na Consciência de Contexto, explorando as etapas de aquisição, modelagem e processamento. A seção 4 apresenta diferentes estratégias de modelagem utilizadas para a representação do contexto. A seção 5 apresenta trabalhos que utilizam a consciência de contexto, onde foi selecionado um trabalho para cada modelo discutido na seção anterior, juntamente com um que utiliza uma estratégia híbrida, e ao final é realizada uma análise comparativa destes trabalhos. E a seção 6 apresenta as considerações finais sobre o artigo.

2 Computação Ubíqua

Com o advento de dispositivos portáteis, inicialmente *handhelds*, PDAs (*Personal Digital Assistants*) e mais recentemente os *smartphones* que possuem capacidade considerável de processamento, além de recursos para comunicação sem fio e armazenamento de dados, consolidou-se um novo paradigma na computação, a Computação Ubíqua. Na UbiComp é introduzido um cenário que combina o emprego de dispositivos portáteis e fixos com a perspectiva de fornecer serviços de forma o mais transparente possível aos usuários, levando em consideração as características do mundo físico. Assim, caso um sistema ubíquo seja capaz de utilizar como entrada informações relevantes sobre as entidades (pessoas, lugares, objetos) relacionadas à aplicação, os mesmos poderão prover serviços de maneira mais precisa, dinâmica e otimizada, aumentando a satisfação dos usuários e minimizando o consumo de recursos, tais como, energia, processamento, comunicação, entre outros.

As aplicações ubíquas possuem basicamente três princípios, sendo eles [4]:

- descentralização: na computação ubíqua, alguns dispositivos ficam encarregados de executar tarefas específicas, fazendo com que as responsabilidades sejam distribuídas. Com isso é preciso que os equipamentos trabalhem e cooperem entre si, criando então uma rede dinâmica de relações entre os dispositivos e os servidores do ambiente, caracterizando um sistema distribuído;
- diversificação: um aspecto interessante do princípio da diversidade é que os sistemas computacionais devem saber gerenciar as diferentes capacidades dos mais diversos equipamentos escolhendo automaticamente aquele que melhor se adequar à determinada situação;
- conectividade: na ubiquidade computacional existe a visão da conectividade ilimitada e sem fronteiras, em que os equipamentos e as aplicações que neles executam estão se movendo com o usuário, entrando e saindo de redes heterogêneas de forma sempre transparente. Para isso, destaca-se a importância da padronização das redes e das interfaces dos dispositivos para que seja possível atingir a conectividade e interoperabilidade desejadas.

Como exemplo de infraestrutura neste sentido destaca-se o ambiente ubíquo gerenciado pelo *middleware* EXEHDA, o qual é mostrado na figura 1. O ambiente apresenta várias células, onde nestas células podem existir inúmeros Servidores de Borda (SB), servidores estes responsáveis pela comunicação com o meio através de sensores e

atuadores, além de um servidor base, chamado Servidor de Contexto (SC), o qual é responsável por armazenar as informações coletadas e permitir a manipulação (processamento, visualização, etc.) destas informações.

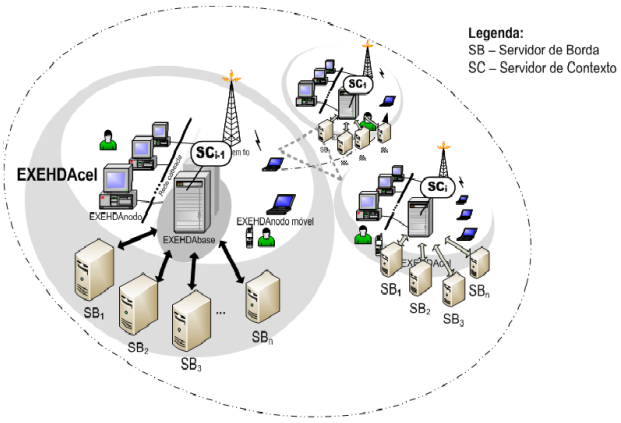

Figura 1. Ambiente ubíquo [5]

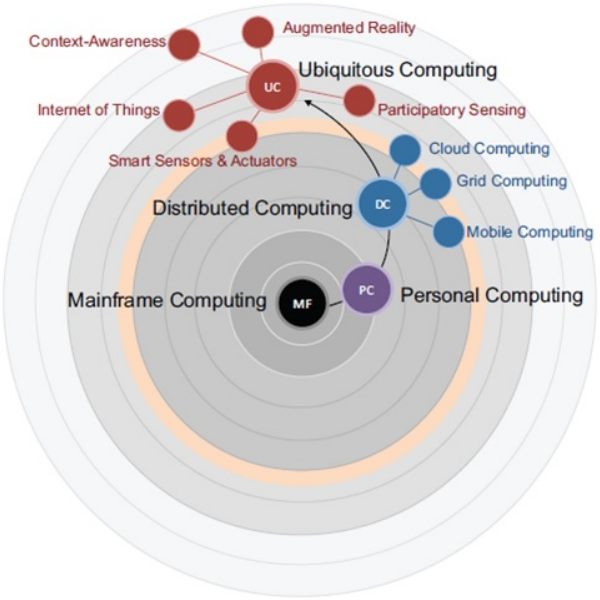

Figura 2. Cadeia Evolutiva e Características da Computação Ubíqua [6]

Sob outra perspectiva, a Computação Ubíqua consolida uma nova era na cadeia evolutiva da computação moderna. Partindo da computação pessoal (segunda geração) e ampliando as perspectivas da computação distribuída (terceira geração), a partir das raízes da computação baseada em mainframes [6]. A figura 2 resume esta cadeia de evolução e apresenta algumas características pertinentes a computação ubíqua.

Dentre as características inerentes a um ambiente ubíquo, está a necessidade da consciência de contexto, que é a capacidade de coletar informações relevantes para o contexto de interesse das aplicações, armazenar os dados coletados, mantendo um histórico que pode ser utilizado para estabelecer tendências sobre os valores de informações do contexto, ou ainda, direcionar ações e comportamentos que interfiram no estado do ambiente do usuário [7].

3 Consciência de Contexto

Contexto é qualquer informação que pode ser usada para caracterizar a situação de uma entidade (pessoa, local ou objeto) que é considerada relevante para a interação entre o usuário e a aplicação, incluindo o próprio usuário e a aplicação [8] [9].

Também pode ser considerado como contexto uma descrição complexa de conhecimento compartilhado sobre circunstâncias físicas, sociais, históricas, entre outras, onde ações ou eventos ocorrem. É o que restringe a interpretação de uma ação ou evento, sem, no entanto, ser parte dessa ação/evento. Portanto o contexto é uma coleção de condições relevantes e influências que tornam uma situação única e compreensível [10].

A consciência de contexto é a capacidade de um sistema em usar o contexto para prover serviços e/ou informações relevantes para o usuário [9].

Os sistemas conscientes de contexto devem ser flexíveis, adaptativos, e capazes de atuar automaticamente para ajudar o usuário na realização de suas atividades.

As áreas da Computação Ubíqua e Inteligência Artificial foram as pioneiras em estudar e utilizar o conceito de contexto, demonstrando o potencial da aplicação desse conceito nos sistemas computacionais. Pesquisas recentes vêm utilizando o conceito de contexto para beneficiar sistemas ligados a outras áreas tais como:

- sistemas colaborativos: utilização de contexto para melhorar a interação e produtividade do grupo, aplicado por exemplo, na área médica;
- hipermídia adaptativa: possibilidade de personalização e adaptação do conteúdo de sites Web a partir de contextos;
- integração de dados: facilitando a resolução

de conflitos semânticos com a utilização de contextos;
- interação humano-computador: o contexto é utilizado para adaptar as interfaces dos sistemas tornando mais intuitiva a sua interação com os usuários.

Algumas motivações para aplicação de consciência de contexto nos sistemas computacionais são:

- auxiliar na compreensão da realidade;
- facilitar na adaptação de sistemas;
- auxiliar no processo de transformação dos dados em informação;
- apoiar a compreensão de eventos;
- ajudar a identificar situações de interesse.

Um sistema consciente de contexto possui três etapas básicas, sendo elas: aquisição, modelagem e processamento. A figura 3 apresenta um exemplo resumido do ciclo de vida do contexto, onde na parte inferior da figura estão os sensores, utilizados pela camada Aquisição para capturar os dados sensorados. Após, estes dados são repassados para a camada de Modelagem, a qual é responsável por modelar e armazenar os dados coletados. Na sequência os dados contextuais são repassados para a camada Processamento, a qual irá processar os dados com o intuito de realizar alguma inferência nos dados, de forma a identificar determinadas situações que podem gerar uma ação a ser realizada. No topo da figura 3 tem a abstração das aplicações que utilizam os dados após a passagem pelas três camadas. Estas três etapas são apresentadas a seguir com maiores detalhes.

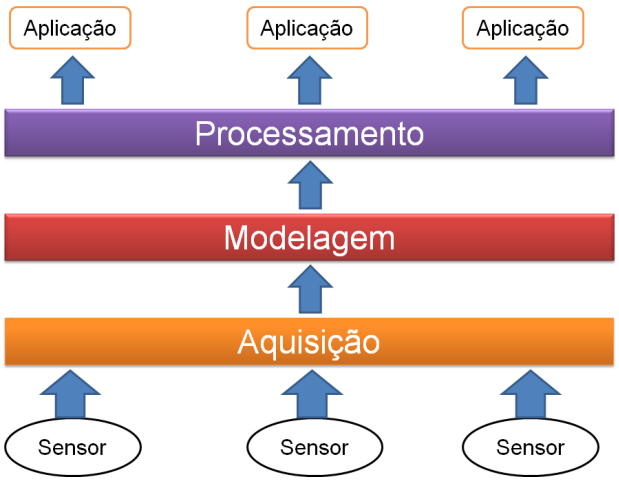

Figura 3. Fluxo de vida do contexto

3.1 Aquisição de Contexto

A aquisição do contexto refere-se ao processo de monitorar, capturar e/ou obter informações contextuais. A informação contextual pode ser adquirida a partir de diversas fontes, tais como [11]:

- sensores físicos: são o tipo mais comum de sensores, tangíveis e geram dados por si só. Os dados obtidos a partir destes sensores são chamados de contexto de baixo nível. Possuem menos significância e apresentam vulnerabilidade a pequenas mudanças. São responsáveis por capturarem informações sobre o ambiente físico, tais como luz, temperatura, umidade, gás. Alguns pontos positivos são: a detecção de erros e a possibilidade de identificar falta de valores de forma relativamente fácil; ser mais eficiente devido a ter acesso às configurações do sensor de baixo nível. Como pontos negativos destacam-se: Implantação e manutenção de hardware que pode ser custosa; tratar com sensores e programação de baixo nível, desenvolvimento, teste, depuração; fornecimento de dados brutos e de baixo nível.
- sensores virtuais: estes sensores não geram necessariamente dados por si mesmos, eles recuperam dados de outras fontes e publicam como dados de sensores. Podem ser utilizados para coletar informações que não podem ser medidas fisicamente, tais como detalhes de calendário, e-mail, bate-papo, mapas, dados de redes sociais, preferências do usuário. Alguns pontos positivos são: fornecimento de dados significativos; informações de contexto de alto nível; dados fornecidos não precisam de processamento; não é necessário lidar com tarefas de nível de hardware. Como pontos negativos destacam-se: dificuldade de encontrar erros nos dados; preenchimento de valores ausentes não é uma tarefa fácil, já que em sua maioria são dados não numéricos e imprevisíveis.
- sensores lógicos: combinam sensores físicos e sensores virtuais, a fim de produzir informações mais significativas, utilizados para coletar informações que não são possíveis de coletar diretamente através de um único sensor físico, e em situações que são ne-

cessárias o processamento e fusão de dados de vários sensores, como por exemplo, informações sobre o tempo, o reconhecimento de atividade, reconhecimento de localização. Alguns pontos positivos são: fornecimento de dados altamente significativos; provimento de informações de contexto de alto nível; normalmente informações mais precisas; não é necessário lidar com tarefas de nível de hardware. Como pontos negativos destacam-se: dificuldade de encontrar erros em dados; preenchimento de valores ausentes não é uma tarefa fácil, já que em sua maioria os dados são valores não numéricos; não se tem controle sobre processo de produção de dados.

As informações contextuais podem ser classificadas das seguintes formas [12]:

- informações percebidas: são provenientes de sensores físicos ou lógicos, possuem uma baixa persistência. As informações podem ser imprecisas, desconhecidas ou caducas, e suas fontes de imperfeição são erros na percepção, falhas do sensor ou desconexões da rede;
- informações provenientes de perfil: são fornecidas pelo próprio usuário, possuem uma persistência moderada. As informações tendem a caducar, e suas fontes de imperfeição são a omissão do usuário em atualizar mudanças ocorridas;
- informações derivadas: são aquelas obtidas por meio de mecanismos de derivação, possuem uma persistência variável. As informações estão sujeitas a erros e imperfeições, e suas fontes de imperfeição são entradas imprecisas, mecanismo de derivação imaturo ou simplificado.

É importante que a tarefa de aquisição de contexto permaneça em constante execução, e é desejável que seja implementada de forma independente das aplicações que a utilizem, possibilitando assim que diversas aplicações possam fazer uso das mesmas informações contextuais.

3.2 Modelagem de Contexto

Um sistema consciente de contexto requer que informações contextuais sejam trocadas e utilizadas por diferentes entidades, como agentes humanos e de software, dispositivos e serviços. E a quantidade de informação capturada e acessada num modelo de representação do contexto é significativa, com isso, os estudos de técnicas para representação de informações contextuais veem se tornando cada vez mais importantes.

O processo de modelagem de contexto consiste na concepção de um modelo de entidades do mundo real, suas propriedades, estado de seu ambiente e situações que podem ser usados como referência para a aquisição, interpretação e raciocínio de informações contextuais [6]. Um bom formalismo de modelagem de informações de contexto reduz a complexidade das aplicações conscientes ao contexto, facilita o acesso às informações realizando buscas de forma eficiente, e melhora a capacidade de manutenção e de evolução da aplicação [13]. Alguns requisitos que devem ser levados em consideração durante a modelagem das informações de contexto são [14]: heterogeneidade, mobilidade, dependências e relações entre os dados.

Existem diversos modelos para representação do contexto, tais como: chave-valor, linguagem de marcação, gráfico, orientado a objetos, lógico, ontológico, entre outros. Devido à relevância deste tópico para as aplicações conscientes de contexto, na seção 4 são apresentadas as principais estratégias presentes na literatura para realizar a etapa de modelagem de contexto, onde são apresentadas as principais características de cada modelo.

3.3 Processamento de Contexto

Processamento de Contexto pode ser definido como um mecanismo de raciocínio para inferir novos conhecimentos e melhorar a compreensão dos contextos adquiridos [15]. Também pode ser apresentado como um processo de realizar deduções de contexto de alto nível a um conjunto de contextos de baixo nível. A necessidade de raciocínio surgiu devido as características do contexto, tais como, imperfeição e incerteza dos dados.

Um número significativo de mecanismos provenientes dos campos da inteligência artificial e com base em sistemas de conhecimento pode ser adotado para realizar o processo de raciocínio de contexto. A seguir são apresentadas algumas das técnicas utilizadas para realizar o processamento e raciocínio sobre a informação contextual presentes nos trabalhos conscientes de contexto. A figura 4 apresenta uma distribuição das técnicas

de processamento de contexto utilizadas por trabalhos encontrados na literatura, onde destaca-se as técnicas baseadas em regras, aprendizagem e em lógica probabilística.

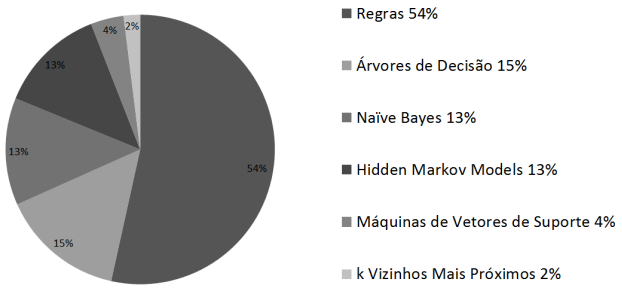

Figura 4. Técnicas de processamento de contexto encontradas nas aplicações conscientes de contexto [15]

- Baseado em regras: este é um dos métodos mais simples para realizar raciocínio sobre contexto, basicamente as regras seguem um formato do tipo se-então-senão. Permite a geração de informações de contexto de alto nível utilizando contextos de baixo nível, é uma técnica simples para se definir e estender, e não necessita de uma utilização intensiva de recursos computacionais. Como mostrado na figura 4 é a técnica mais utilizada em trabalhos conscientes de contexto. O raciocínio baseado em regras possui alguns pontos negativos, tais como: quando se utiliza uma grande base de regras facilmente se torna confuso e intratável; as regras devem ser definidas manualmente, o que é propenso a erros devido ao trabalho manual; e não possui mecanismo para realizar a validação e verificação de qualidade. O raciocínio baseado em regras só pode ser aplicado em sistemas de suporte de contexto com propagação baseado em eventos, e não suporta imprecisão, somente sendo aplicada para respostas do tipo booleana.
- Baseado em aprendizagem supervisionada: nesta categoria estão as técnicas que utilizam um conjunto de treinamento, onde neste conjunto os dados se encontram categorizados, ou seja, está presente o resultado esperado para cada caso que será utilizado para treinamento, e em seguida é possível classificar novos eventos com base no conjunto que foi utilizado durante o treinamento. Dentre as técnicas desta categoria destacam-se: Árvores de Decisão que é uma técnica de aprendizado supervisionado onde é construída uma árvore a partir de um conjunto de dados que podem ser utilizados para classificar os dados, e as Máquinas de Vetores de Suporte, as quais são amplamente utilizadas para reconhecimento de padrões em computação consciente de contexto. Os pontos positivos destas técnicas são que costumam alcançar um alto grau de precisão, a disponibilidade de modelos alternativos, além de possuir uma boa base matemática e estatística. Dentre as dificuldades encontradas na utilização destas técnicas pode-se citar: a exigência de quantidade significativa de dados para treinamento; pode ser necessário uma maior utilização de recursos computacionais, tais como, processamento, armazenamento; seleção de dados que serão utilizados, de forma a não tornar o processamento muito custoso.
- Baseado em aprendizagem não supervisionada: estas técnicas utilizam um conjunto de treinamento para aprender, mas estes dados não estão categorizados, eles não possuem o resultado esperado. Nesta categoria destacam-se as técnicas de agrupamento, como a técnica k Vizinhos Mais Próximo. Estas técnicas costumam ser utilizadas em redes de sensores, para realizar tarefas como de roteamento, e também nas operações de posicionamento e localização. As técnicas utilizadas possuem destaque por conseguirem aprender sem precisarem de um conjunto de treinamento com as respostas esperadas. Como pontos negativos destacam-se: dificuldade em realizar a validação; complexidade que os modelos podem alcançar; imprevisibilidade dos resultados; utilização de recursos computacionais podem se tornar intensivos.
- Baseado em lógica descritiva: é aplicado em conjunto com a representação de contexto ontológico. A modelagem semântica de conceitos (classes), papéis (propriedades, relações) e indivíduos, permitem que o conhecimento a ser especificado seja interpretável por máquina. O raciocínio baseado em ontologias é computacionalmente intensivo e o tempo de resposta dependerá em grande

parte do tamanho do conjunto de dados e do conjunto de regras presente na ontologia. Além da complexidade de raciocínio, a concepção de uma ontologia é dita ser uma tarefa complexa que exige conhecimentos de domínio. Como pontos positivos do raciocínio baseado em lógica descritiva têm-se a possibilidade de raciocínio complexo, representação complexa, resultados significativos, e a possibilidade de validação e verificação da qualidade. Os pontos negativos na sua utilização destacam-se: a necessidade de os dados precisarem ser modelados em formatos compatíveis; o baixo desempenho, já que podem exigir um tempo maior para realizar o processamento; necessidade de um maior poder computacional.

- Baseado em lógica probabilística: estas técnicas permitem que as decisões sejam tomadas com base em probabilidades associadas aos eventos. Pode ser utilizado para combinar os dados dos sensores a partir de fontes diferentes. Além disso, pode ser usado para identificar as resoluções de conflitos entre contextos. Na maioria das vezes estas técnicas são usadas para entender ocorrência de eventos. O raciocínio probabilístico é especialmente aplicável em ambientes conscientes de contexto, devido as potenciais falhas temporárias na comunicação com os sensores, e a possibilidade de medições imprecisas de sensores físicos. As principais vantagens desta estratégia são a possibilidade de combinar eventos, lidar com a incerteza, e fornecer resultados moderadamente significativos. E como pontos negativos destacam-se a dificuldade de só trabalhar com valores numéricos e a necessidade de saber as probabilidades dos eventos. Dentre as técnicas utilizadas nesta estratégia destaca-se Naïve Bayes que é considerada a estratégia generativa mais simples para a classificação de variáveis de classe com uma única de pendência de valores, e os *Hidden Markov Models* que representa dados estruturados sequencialmente, permitindo que o estado seja representado usando evidências observáveis sem ler diretamente o estado.

4 Estratégias para Modelagem de Contexto

Atualmente, diversas estratégias para representação de contexto são encontradas na literatura, sendo que cada uma possui vantagens e desvantagens na sua utilização, sendo assim, ainda não se tem uma técnica que seja considerada ideal para representar qualquer tipo de contexto. A seguir são apresentadas algumas das estratégias utilizadas para realizar a modelagem de contexto levantadas pelos surveys [13], [6], [15].

4.1 Modelos Chave-Valor

Esta estratégia de modelagem é a que utiliza a estrutura de dados mais simples para representar a informação contextual. O contexto é representado através de pares compostos por uma chave, que identifica o atributo de contexto, e por um valor associado a essa chave. Os dados podem ser representados em diferentes formatos, como por exemplo, arquivos textos e binários. É um modelo simples de utilizar e manipular, não permite estruturações mais sofisticadas que habilitem algoritmos eficientes de recuperação de contexto. Esta é a forma mais simples para representação do contexto, mas a modelagem deste tipo não é escalável, e não é adequada para armazenar estruturas de dados complexas, além disso, não é possível modelar estruturas ou relações hierárquicas [15].

Pode ser utilizada para modelar quantidade limitada de dados, como as preferências do usuário e configurações de aplicativos. Em sua maioria dados que não estão relacionados e são independentes entre si. Um exemplo de modelagem baseada em pares chave-valor é apresentada na tabela 1. Alguns exemplos de trabalhos que utilizam a modelagem de chave-valor são: [16], [17], [18], [19], [20] e [21].

Tabela 1
Exemplo de modelagem chave-valor

Chave	Valor
Localização	Pelotas, RS
Idade	20 anos
Estado Civil	Solteiro

4.2 Modelos Baseados em Linguagens de Marcação

Esta estratégia de modelagem utiliza estruturas de dados hierárquicas, consistindo de *tags* com atributos e conteúdo para armazenar o contexto, sendo considerada uma melhoria sobre a estratégia de chave-valor. A vantagem de usar *tags* de marcação é que permite a recuperação de dados de forma eficiente. Além disso, a validação é suportada através de definições de esquema, e sofisticadas ferramentas de validação estão disponíveis para técnicas populares, tais como, a marcação XML (*eXtensible Markup Language*). Em contraste, as linguagens de marcação não oferecem recursos avançados de expressão que permitam raciocínio sobre as informações contextuais [15]. Uma aplicação comum de modelagem baseada em linguagens de marcação é a modelagem de perfis.

Pode ser utilizado como um formato intermediário de organização de dados, bem como, um modo de transferência de dados através da rede. E ainda pode ser usado para separar as estruturas de dados utilizadas por dois componentes em um sistema. Um exemplo é o CSCP (*Comprehensive Structured Context Profiles*) [22]. O CSCP é um modelo que representa o contexto como perfis de sessão, a figura 5 apresenta um exemplo desta estratégia. Alguns exemplos de trabalhos que utilizam a modelagem baseada em linguagens de marcação são: [23], [24], [25], [26], [27], [28], [29], [30], [31].

```
<?xml version="1.0"encoding="UTF-8"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:cscp="context-aware.org/CSCP/CSCPProfileSyntax#"
    xmlns:dev="context-aware.org/CSCP/DeviceProfileSyntax#"
    xmlns:net="context-aware.org/CSCP/NetworkProfileSyntax#"
    xmlns="context-aware.org/CSCP/SessionProfileSyntax#"
    <SessionProfile rdf:ID="Session">
        <cscp:default  rdf:resource=
            http://localContext/CSCPProfile/previous#Session"/>
        <device><dev:DeviceProfile>
            <dev:hardware><dev:Hardware>
                <dev:memory>9216</dev:memory>
            </dev:Hardware></dev:hardware></dev:DeviceProfile>
        </device>
    </SessionProfile>
</rdf:RDF>
```

Figura 5. Exemplo CSCP [13]

4.3 Modelos Gráficos

Esta estratégia de modelagem utiliza elementos gráficos para a representação de contexto, tais como, grafos contextuais, ORM (*Object Role Modeling*), UML (*Unified Modeling Language*). A modelagem utilizando grafos contextuais [32] são baseados em redes semânticas e suportam a representação de modelos de tarefas. As estratégias baseadas em ORM [33] envolvem a identificação de fatos e papéis executados pelas entidades, a figura 6 apresenta um exemplo de representação utilizando o modelo ORM. E as estratégias baseadas em UML utilizam extensões dessa linguagem, como perfis e estereótipos, para representar informações de contexto, um exemplo desta estratégia é CMP (*Context UML Profile*) [34], a seguir são apresentados alguns diagramas da UML e suas finalidades na modelagem de contexto.

- Diagrama de Casos de Uso: modelar os atores envolvidos, relacionados ou influenciados pelo contexto e suas possibilidades de interação.
- Diagrama de Componentes: modelar os sistemas envolvidos que contêm informações relacionadas ao contexto, como bancos de dados, fontes de contexto, entre outros.
- Diagrama de Classes: representar a informação estrutural do domínio e modelar como o contexto está estruturado.
- Diagrama de Sequência: modelar cenários de ativação do contexto. O diagrama detalha o fluxo da informação entre os sistemas envolvidos e exibe a sequência de disseminação da informação.

Os modelos gráficos possibilitam um melhor entendimento da estrutura das informações contextuais, mas possuem um baixo grau de formalismo [13]. Podem ser utilizados para armazenamento em longo prazo de grandes volumes de dados, e os contextos históricos podem ser armazenados em bancos de dados. Alguns exemplos de trabalhos que utilizam modelos gráficos são: [35], [36], [37].

4.4 Modelos Orientados a Objetos

As estratégias orientadas a objetos modelam os dados utilizando hierarquias e relações entre classes, e tentam explorar os benefícios do paradigma de programação orientado a objetos, tais como, encapsulamento e reusabilidade. Os detalhes de processamento de contexto são encapsulados no nível de objetos, e o acesso às informações de contexto é realizado apenas por meio de interfaces.

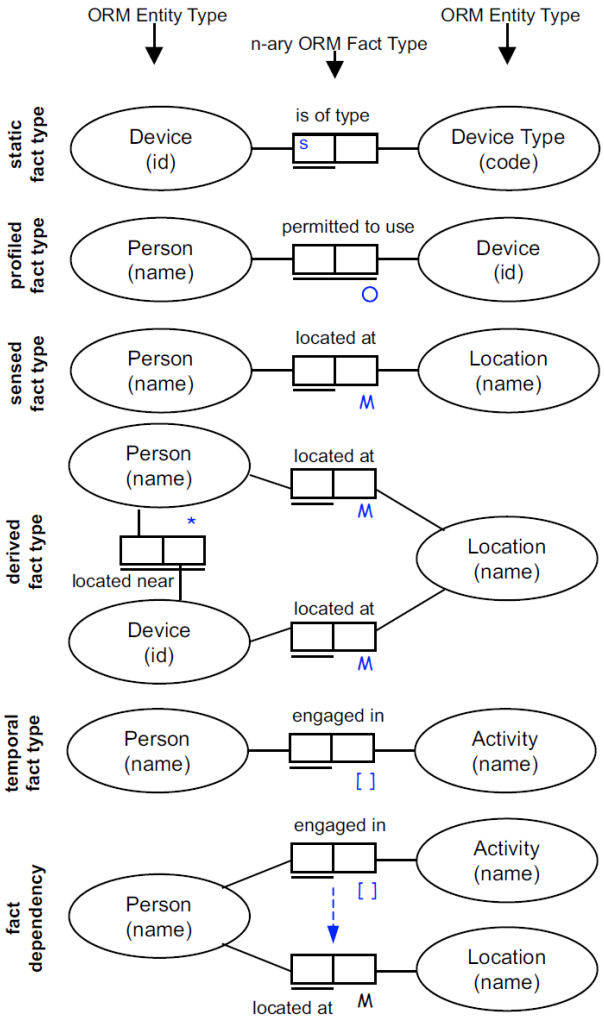

Figura 6. Modelo ORM [13]

Os modelos orientados a objetos são fortes em relação à possibilidade de composição distribuída, pois novos tipos de informações de contexto podem ser adicionados e instâncias podem ser atualizadas de forma distribuída. Porém, normalmente, as infraestruturas de execução que suportam esses modelos exigem muitos recursos dos dispositivos computacionais, o que nem sempre pode ser atendido em ambientes de computação ubíqua [13]. E ainda esta estratégia não fornece capacidade de raciocínio sobre os contextos, e a validação de projetos é difícil, devido a falta de normas e especificações [15].

Esta estratégia pode ser utilizada para representar contexto no nível de código de programação, permite a manipulação do contexto em tempo de execução. Um exemplo dessa estratégia é apresentado na figura 7, onde as informações de contexto são representadas por um conjunto de entidades, que, por sua vez, descrevem objetos físicos ou conceituais, como um canal de comunicação ou uma pessoa. Alguns exemplos de trabalhos que utilizam modelos orientados a objetos são: [38], [39], [40], [41], [42].

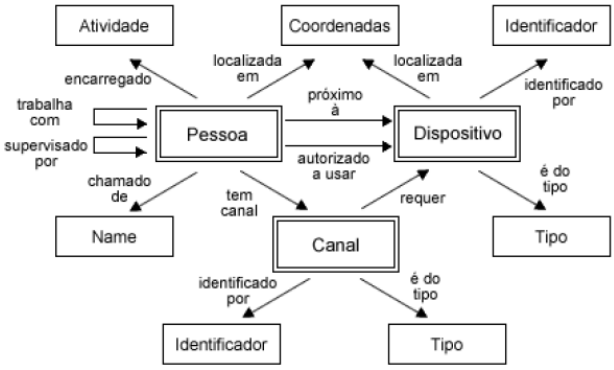

Figura 7. Exemplo de modelo orientado a objetos [43]

4.5 Modelos Baseados em Lógica

Nas estratégias baseadas em lógica a informação contextual é modelada como fatos, expressões e regras. Além disso, um processo de inferência pode ser utilizado para derivar novos fatos com base nas regras modeladas, com isso a informação contextual é adicionada, atualizada e removida em termos de fatos ou inferências a partir de regras. Essa estratégia possui um alto grau de formalismo, no entanto, a falta de normalização reduz a possibilidade de reutilização e aplicabilidade, e pode implicar em uma maior dificuldade de manutenção e compreensão [15]. A modelagem baseada em lógica permite que novas informações de contexto de alto nível possam ser extraídas utilizando informações de contexto de baixo nível, adicionando assim a capacidade de melhorar outras técnicas de modelagem de contexto.

Esta estratégia pode ser utilizada para gerar eventos, ações no modelo, e definir restrições ou limitações. A figura 8 ilustra parte de um modelo baseado em lógica relacionado à localização do usuário de uma determinada aplicação. Alguns trabalhos que utilizam a modelagem baseadas em lógica são: [44], [45], [46].

```
(H_UserAtLocation (uid ?uid)(rid ?rid)
    (start-time ?start-time))
(H_UserColocation (uid-list ?uid_1 … ?uid_n)
    (rid ?region-id)(start-time ?time-value))
(H_UserIsPresent (uid ?uid)(start-time ?time-value))
(L_UserAtLocation (uid ?uid)(x ?x) (y ?y) (z ?z))
```

Figura 8. Exemplo de parte de um modelo baseado em lógica [47]

4.6 Modelos Baseados em Ontologias

Nesta estratégia o contexto é organizado em ontologias. Uma ontologia é uma especificação explícita, formal, de uma conceitualização compartilhada em que objetos, conceitos, entidades e relacionamentos do mundo real são definidos em uma determinada área de interesse ou domínio de conhecimento [48]. Um dos grandes interesses na construção e uso de ontologias é tornar o conhecimento sobre o mundo real processável por máquinas. As ontologias consistem de vários componentes chaves, tais como, indivíduos, classes, atributos, relações, termos de funções, restrições, regras, axiomas e eventos. A figura 9 ilustra parte de uma ontologia para modelagem de contexto.

O desenvolvimento de ontologias pode ser dividido em duas etapas [15], onde na primeira etapa o domínio e o escopo são definidos, e na segunda etapa é analisada ontologias já existentes que possam ser reutilizadas, já que um dos principais objetivos das ontologias é a reutilização do conhecimento compartilhado.

Uma das vantagens da utilização das ontologias é a possibilidade de raciocínio lógico, o qual pode ser utilizado pelas aplicações para inferir contextos de alto nível a partir de contextos de baixo nível, e para checar e resolver inconsistências no conhecimento contextual. O raciocínio é realizado por um motor de inferência o qual permite: juntar diferentes fragmentos ontológicos; deduzir conhecimento a partir de axiomas codificados simbolicamente; consultar instâncias e seus valores; consultar nomes de conceitos e atributos baseados em ontologias conhecidas; validar a consistência de uma ou mais ontologias; atribuir relacionamentos inter-ontológicos; e completar as ontologias através do processamento de hierarquia implícita e relacionamentos baseados em regras existentes [49].

Como desvantagens pode-se destacar a complexidade que a representação pode alcançar, e a grande utilização de recursos computacionais necessários para realizar a recuperação das informações.

Esta estratégia pode ser utilizada para modelar o conhecimento do domínio e a estrutura do contexto baseada nos relacionamentos definidos pela ontologia. Alguns exemplos de trabalhos que utilizam a modelagem baseada em ontologias são: [50], [51], [49], [52], [53], [54], [55], [56].

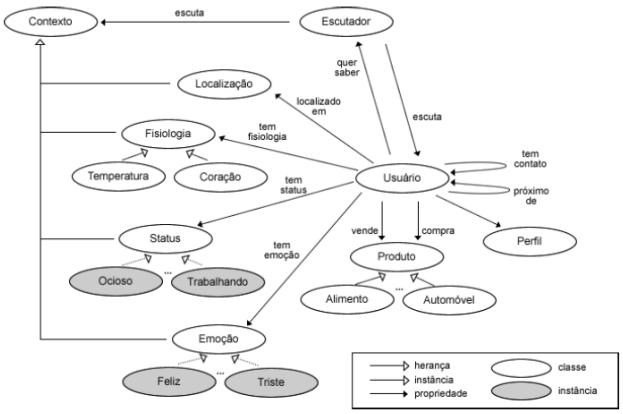

Figura 9. Exemplo de ontologia para modelagem de contexto [57]

4.7 Análise das Estratégias de Modelagem

Como apresentado anteriormente há diversas estratégias que podem ser utilizadas para realizar a modelagem de informações contextuais, com isso torna-se necessário métricas que possam ser utilizadas para realizar a comparação entre as estratégias de modelagem. Com base em [13] a tabela 2 apresenta uma comparação entre alguns modelos de contexto, onde o sinal de (-) significa limitação do modelo, (–) limitação ainda maior, o sinal de (+) o atendimento da métrica e o (++) o atendimento de uma forma mais satisfatória. A análise dos modelos é realizada tendo como base as seguintes características [13]:

- composição distribuída (dc): a composição e administração dos modelos de contexto são extremamente dinâmicas em termos do tempo, topologia da rede e recursos;
- validação parcial (pv): capacidade para validar conhecimento parcial. Em determinados momentos, devido a composição distribuída,

não é possível validar todo o conhecimento de contexto;
- qualidade da informação (qua): a qualidade da informação muda de acordo com o sensor utilizado, entre outros fatores. Os métodos devem suportar o tratamento de informação com destintos níveis de qualidade;
- incompleteza e ambiguidade (inc): o método deve ser capaz de tratar informações incompletas e ambíguas;
- nível de formalidade (for): modelos com sintaxe e semântica bem definidas.
- aplicabilidade em ambientes já existentes (app): utilização de modelos em aplicações já existentes.

Tabela 2
Comparação entre estratégias de modelagem de contexto

Modelo	dc	pv	qua	inc	for	app
Chave-Valor	-	-	- -	- -	- -	+
Linguagem de Marcação	+	++	-	-	+	++
Gráfico	- -	-	+	-	+	+
Orientado a Objetos	++	+	+	+	+	+
Lógico	++	-	-	-	++	- -
Ontológico	++	++	+	+	++	+

Como pode ser observado na tabela 2 cada técnica possui vantagens e desvantagens em relação a sua utilização, sendo que ainda não foi encontrado uma técnica que seja considerada a ideal, com isso tem surgido as estratégias híbridas. Os modelos híbridos de modelagem de contexto são considerados os mais promissores, pois combinam diferentes técnicas de modelagem, com diferentes níveis de interpretação, para diferentes aspectos. Alguns exemplos de trabalhos que utilizam modelos híbridos são: [58] [59], [60], [61], [62].

A utilização de modelos híbridos traz um novo desafio, que é como realizar o gerenciamento das informações contextuais, já que se pode combinar diferentes modelos e diferentes formas de armazenar as informações contextuais. Com isso é importante ter mecanismos que facilitam o acesso as informações contextuais capturadas, de forma a garantir as atualizações das informações e mantendo a consistência dos dados contextuais.

5 Trabalhos Conscientes de Contexto

Nesta seção foram selecionados trabalhos que utilizam a consciência de contexto, com o objetivo de mostrar as diferentes abordagens utilizadas em sistemas conscientes de contexto. Onde apresenta-se um trabalho para cada tipo de modelagem discutida na seção anterior, e outro que utiliza uma estratégia híbrida. Na análise dos projetos serão explorados aspectos referentes à sua arquitetura, técnicas empregadas, principais funcionalidades e aplicabilidade. Ao final será traçado um comparativo tendo como parâmetros as etapas empregadas para provisão de consciência de contexto.

5.1 MidSen

MidSen [21] é um *middleware* consciente de contexto para Redes de Sensores Sem Fio (RSSF). Utiliza como estratégia de modelagem o modelo Chave-Valor e baseia-se em regras de Evento-Condição-Ação (ECA). O projeto destaca a importância da detecção de eventos eficientes através do processamento de dois algoritmos, Algoritmo de Detecção de Eventos e Algoritmo de Descoberta de Serviços Conscientes de Contexto. Na figura 10 é apresentada a arquitetura do *middleware* MidSen, onde destaca-se os seguintes componentes: *Knowledge Base*; *Application Interface*; *Knowledge Manager*; *Application Notifier*; *Inference Engine*; *Network Interface*; *Working Memory*.

5.2 Aura

Aura [24] é um projeto baseado em um *middleware* orientado a tarefas que possui uma arquitetura distribuída e que tem como propósito criar uma infraestrutura de serviços para a Computação Ubíqua.

O projeto parte do princípio que a atenção humana é o recurso mais escasso em ambientes computacionais ao invés dos recursos de hardware como: velocidade do processador, memória principal, largura de banda de rede e capacidade de disco. A atenção humana refere-se a capacidade de um usuário em atender às suas tarefas primárias, ignorando distrações geradas pelo sistema, tais como baixo desempenho e falhas. Aura tem por objetivo minimizar as distrações sobre a atenção do usuário, criando a visão de uma "Aura de Informações Pessoais" através do desenvolvimento

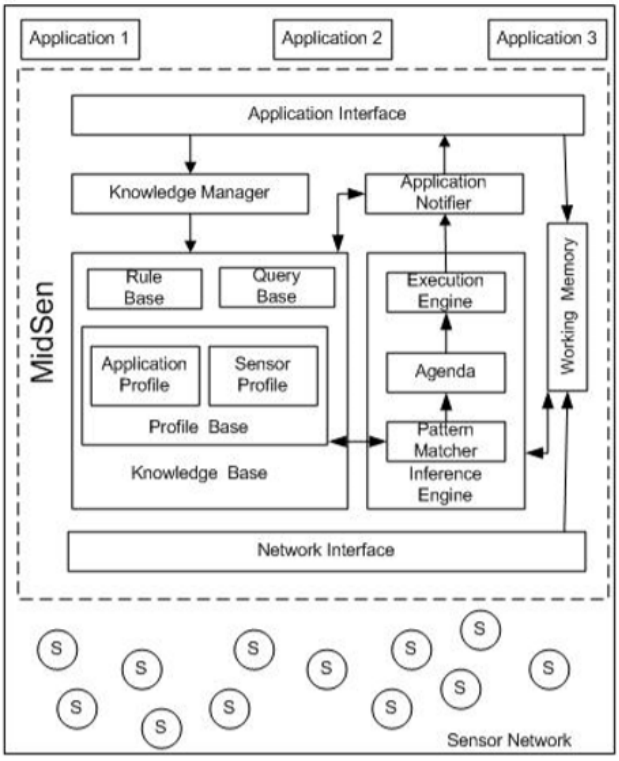

Figura 10. Arquitetura MidSen [21]

de arquiteturas, algoritmos, interfaces e demais técnicas necessárias, propiciando um ambiente que se adapta ao contexto e às necessidades do usuário. A estratégia de modelagem que descreve os serviços é o modelo Baseado em Linguagem de Marcação, mais especificamente XML.

A arquitetura do projeto Aura pode ser visualizada na figura 11 onde são destacados os seguintes componentes:

- *Odyssey*, que suporta adaptação e monitoramento dos recursos;
- *Coda*, que provê acesso a arquivos de forma distribuída e adaptável à largura de banda e conectividade;
- *Spectra*, que é um mecanismo adaptativo de execução remota baseado em contextos;
- *Prism*, que captura e gerencia a intenção do usuário atuando de forma proativa, isto é, se antecipando às requisições do mesmo.

O projeto aborda dois grandes desafios. Primeiro, permitir que um usuário preserve a continuidade de seu trabalho ao se deslocar em diferentes ambientes. Segundo, ser capaz de adaptar-se a computação em curso de um determinado ambiente, na presença de variabilidade dinâmica de recursos. Aura mostra a importância de ter um middleware voltado para a IoT (*Internet of Things*) que funcione sobre diversas plataformas e dispositivos com diferentes limitações de recursos, tais como, *tablets*, *smartphones*, computadores.

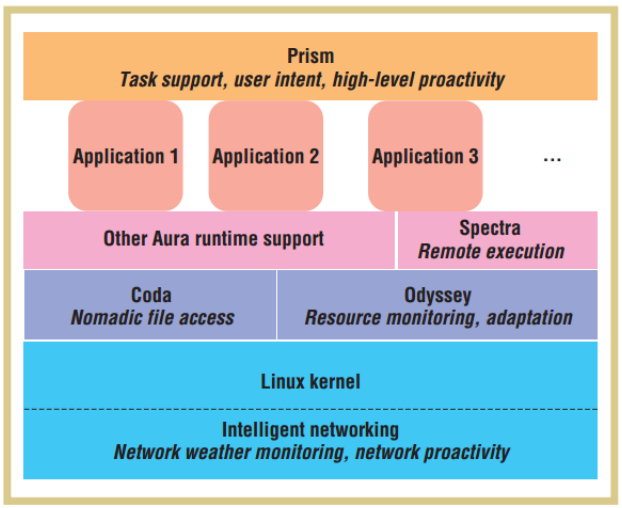

Figura 11. Arquitetura Aura [24]

5.3 Spatial CONtext STREAm Management

SCONSTREAM (*Spatial CONtext STREAm Management*) [63] é um sistema de processamento de fluxo de contextos espaciais que utiliza a estratégia de modelagem gráfica. O trabalho destaca como desafio o processamento em tempo real de fluxo de contextos espaciais, pois enquanto o fluxo de dados brutos espaciais fornecidos por sensores devem ser manuseados em tempo real, o fluxo de dados espaciais conscientes de contexto requerem muitas vezes uma análise complexa e com alto custo de processamento, tornando difícil a integração entre as etapas de fluxo de dados espaciais brutos, e a de fluxo de dados espaciais conscientes de contexto.

O sistema SCONSTREAM apresenta como alternativa de solução a conversão do fluxo de dados brutos espaciais sensorados em fluxo de contextos espaciais menores e mais adequados para serem processados pelo módulo de consciência de contexto. Esta conversão é realizada através da utilização de operadores, como por exemplo, *Inside(x,m)*. Este operador denota que "m"é um objeto dentro do espaço x, portanto a conversão é realizada de forma a deixar passar para o Módulo Espacial Consciente de Contexto

somente os objetos que estejam dentro do espaço que consta no operador.

A arquitetura do SCONSTREAM é apresentada na figura 12, onde pode-se notar os componentes presentes na mesma. O componente *Spatial Data Stream Management System* armazena temporariamente em um buffer, denominado janela deslizante, os dados espaciais continuamente recebidos dos sensores. Estes dados armazenados são convertidos em fluxo de contexto espacial mediante consulta contínua gerenciada pelo componente *Continuous Query Manager*, utilizando para tal operadores presentes no componente *Spatial Operators for Data Stream* e com base em informações espaciais mantidas em um catálogo de sensores gerenciados pelo componente *Sensor Catalog Manager*. *Data Archive* armazena os dados contextuais. Quando um novo contexto espacial é produzido o componente *Trigger* aciona a operação de análise do *Spatial Context-awareness Module* que pode requisitar o contexto gerado, ou ainda obter outras informações mais detalhadas que se tornem necessárias e que ficam armazenadas no *Spatial DBMS*.

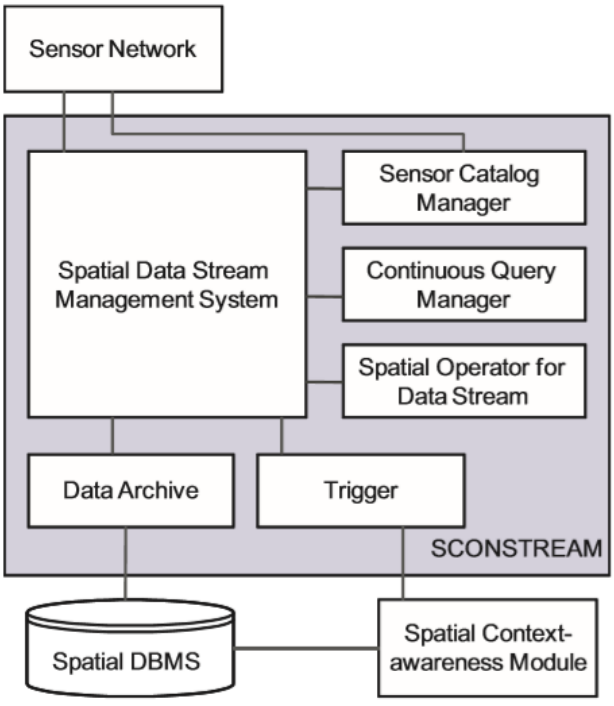

Figura 12. Arquitetura do sistema SCONSTREAM [63]

5.4 COntext entitieS coMpositiOn and Sharing

COSMOS (*COntext entitieS coMpositiOn and Sharing*) [40] é um *framework* baseado em componentes para o gerenciamento de dados de contexto em ambientes ubíquos. Para realizar a modelagem das informações de contexto foi escolhida a estratégia orientada a objetos, onde o contexto é definido como um nó de contexto, o qual é organizado em hierarquias. Possui como aplicações alvo, por exemplo, guias turísticos baseados em computador com navegação, ou aplicações com anotações contextuais, tais como jogos multiplayer. O gerenciamento de contexto fornecido por COSMOS é centrado no usuário, e no aplicativo, com o objetivo de fornecer informações que podem ser facilmente processadas.

Na figura 13 é apresentada a arquitetura do *framework* COSMOS, onde pode-se notar a divisão em três camadas da arquitetura, sendo elas:

Context collector, define a noção de um coletor de contexto. Coletores de contexto são entidades de software que fornecem dados brutos sobre o ambiente. Esses dados podem ser provenientes do sistema operacional, dispositivos de rede, ou qualquer outro tipo de equipamento de hardware. A noção de um coletor de contexto também abrange informações provenientes de preferências do usuário.

Context processing, define a noção de um processador de contexto. Processadores contexto filtram e reúnem dados brutos provenientes dos colecionadores de contexto. O objetivo é calcular informações de alto nível, numéricas ou discretas, sobre o ambiente de execução. Os dados fornecidos pelo processador de contexto são transmitidos para a camada de adaptação.

Context adaptation, responsável pelo processo de tomada de decisão. O objetivo é ser capaz de realizar uma ação quando se verificar necessário. A camada de adaptação é um serviço que é fornecido para os aplicativos encapsulando as situações identificadas.

5.5 The Use of Situation Theory in Context Modeling

Uma estratégia de modelagem baseada em lógica foi proposta por Akman e Surav e é denominada Teoria da Situação Estendida [44], como

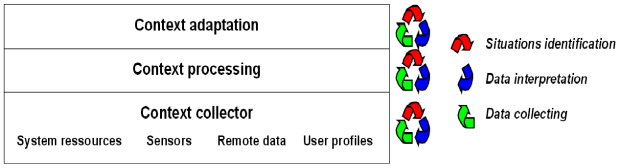

Figura 13. Arquitetura do gerente de contexto COSMOS [40]

uma extensão à Teoria da Situação, proposta por Barwise e Perry, os quais tentaram cobrir semânticas de linguagem natural em um sistema de lógica formal. O modelo da Teoria da Situação Estendida modela o contexto como tipos de situações comuns. A variedade de diferentes contextos é tratada sob a forma de regras e pressuposições relacionadas a um ponto de vista em particular. Eles representam os fatos relativos a um dado contexto com expressões livres de parâmetros e suportadas pelo tipo de situação correspondente ao contexto. A Figura 14 mostra um exemplo de como as regras de um contexto são representadas como restrições nessa estratégia.

$$S_1 = [\dot{s} \mid \dot{s} \models\ll bird, \dot{a}, 1 \gg]$$
$$S_2 = [\dot{s} \mid \dot{s} \models\ll flies, \dot{a}, 1 \gg]$$
$$B \models\ll present, air, 1 \gg \wedge \ll penguin, \dot{a}, 0 \gg \wedge \ldots$$
$$C = S_1 \Rightarrow S_2 \mid B$$

Figura 14. Exemplo de regras de contexto na Teoria da Situação Estendida [44]

5.6 Context Broker Architecture

CoBrA (*Context Broker Architecture*) é uma arquitetura baseada em agentes cujo objetivo é apoiar sistemas conscientes de contexto, em espaços inteligentes, em particular, salas de reuniões inteligentes em um campus universitário. O elemento principal dessa arquitetura é um agente inteligente chamado *context broker* que mantém e gerencia um modelo compartilhado de contexto, a ontologia *CoBrA-Ont*, provendo serviços de proteção de privacidade para os usuários [64].

CoBrA-Ont é uma ontologia desenvolvida em OWL (*Web Ontology Language*) que tem por objetivo auxiliar os agentes na aquisição, raciocínio e compartilhamento do conhecimento, bem como apoiar a detecção e resolução de conhecimento contextual inconsistente e ambíguo. A representação gráfica dos conceitos ontológicos da *CoBrA-Ont*, é categorizada em quatro temas distintos e relacionados:

- conceitos que definem lugares físicos e suas relações espaciais associadas;
- conceitos que definem agentes (humanos e de software);
- conceitos que descrevem o contexto de localização de um agente em um campus universitário;
- conceitos que descrevem os contextos de atividade de um agente, incluindo papéis, desejos e intenções associadas em um evento de apresentação.

A arquitetura do CoBrA é ilustrada na figura 15, onde pode se notar alguns dos requisitos para gerenciamento de contexto tratados, tais como [65]:

- *Context knowledge base*: gerencia o armazenamento do conhecimento do *context broker's*. Este conhecimento inclui as ontologias, as quais descrevem vários tipos de contextos, os dados instanciados da ontologia, e os meta-dados que descrevem a estrutura de armazenamento do conhecimento representado;
- *Context Reasoning Engine*: motor de inferência lógica para o raciocínio sobre a informação contextual adquirida. A função deste motor inclui a interpretação de contexto com base nos dados sensorados adquiridos, a agregação de informação contextual a partir de múltiplas fontes, a utilização de ontologias e heurísticas de domínio, e a detecção e resolução de inconsistências nas informações capturadas;
- *Context Acquisition Module*: conjunto de procedimentos para a aquisição de informações contextuais a partir de sensores, agentes e da Web. Tem como objetivo melhorar a reutilização dos procedimentos de detecção de contexto;
- *Privacy Management Module*: gerencia as políticas de privacidade dos usuários e, controla o compartilhamento de suas informações privadas, e ajuda a orientar o raciocínio lógico para ajustar a granularidade das informações.

Para realizar o raciocínio sobre contexto, Co-

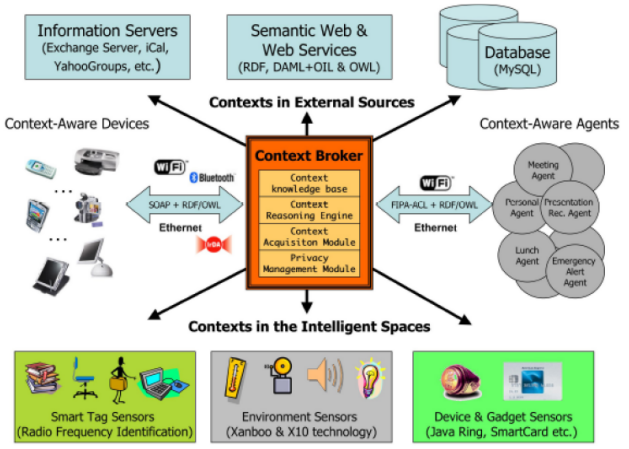

Figura 15. Visão geral da arquitetura CoBrA [65]

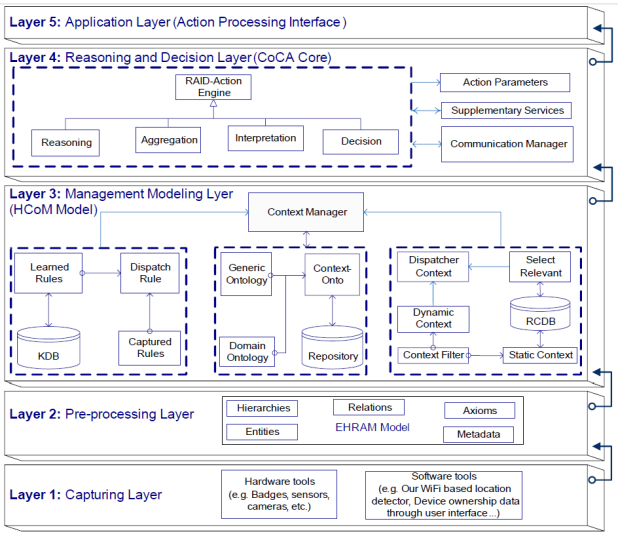

Figura 16. Arquitetura da plataforma CoCA [69]

BrA utiliza um número de diferentes sistemas baseados em regras, como Jena [66], usado para inferência sobre a ontologia OWL, JESS (*Java Expert System Shell*) [67], usado para interpretação de contexto utilizando regras específicas do domínio, e Theorist [68], um raciocinador baseado em sentenças Prolog utilizado para apoiar as inferências lógicas para resolver conhecimento inconsistente [65].

5.7 Collaborative Context-Aware Service Platform

CoCA (*Collaborative Context-Aware Service Platform*) [60] é uma plataforma que realiza as tarefas de representação, raciocínio, agregação e interpretação de dados de contexto. As ações e decisões são realizadas com base nos dados de contexto adquiridos, e a plataforma suporta a colaboração e o compartilhamento de recursos computacionais entre os dispositivos. A figura 16 apresenta a arquitetura da plataforma CoCA dividida em camadas, mostrando suas principais funcionalidades.

Camada 1: camada de aquisição, responsável por lidar com as ferramentas de aquisição de dados, tanto as ferramentas de hardware, sensores e câmeras, como as de software, onde desta-se o rastreador local, que converte um sinal Wi-Fi (*Wireless Fidelity*) em um nome de localização significativa. A interface da plataforma é construída com base em APIs (*Application Programming Interface*), devido ao fato de ter sido concebida para ser utilizada por aplicações de diferentes domínios.

Camada 2: a camada de pré-processamento, é usada para formalizar e preparar os dados capturados para posterior processamento. Trata-se de uma modelagem conceitual dos dados capturados de acordo com o formalismo de representação do contexto EHRAM (*Entities Hierarchies Relations Axioms Metadata*), que realiza a separação de dados em entidades, hierarquias, relações, axiomas e metadados. Isto permite organizar e processar os dados de contexto e dados semânticos de contexto separadamente.

Camada 3: a camada de modelagem de gerenciamento de contexto, trata de como são organizados os contextos úteis para o raciocínio. A representação formal destes dados é realizado por meio do modelo HCoM (*Hybrid Context Management*) [70], que é uma estratégia híbrida que combina a característica semântica das ontologias e os esquemas relacionais. Onde é detacado que sistemas de gerenciamento de banco de dados por si só não podem ser usados para gerenciar contexto, e que as ontologias podem não ter um bom desempenho em termos de eficiência e de processamento de consultas com grandes volumes de dados, tornando-se assim, necessária a estratégia híbrida. Os principais componentes desta camada são:

- *Context Manager*: agrega os resultados e envia os dados para o motor de raciocínio;
- *Collaboration Manager*: responsável por tentar reunir mais dados de outras fontes de contexto possíveis, caso seja necessário;

- *Context Filter*: responsável por realizar a validação e decidir se o contexto precisa ser armazenado em *RCDB*;
- *Context Selector*: com base na solicitação do usuário decide o contexto que deve ser usado no processamento baseado na precisão, tempo e recursos computacionais exigidos;
- *Context-onto*: gerencia as ontologias e funciona como um repositório;
- *Rules and Policy*: permite aos usuários adicionar regras para o sistema;
- RCDB (Relational Context Database): armazena o contexto capturado em um sistema de gerenciamento de banco de dados;
- *Rule-Mining*: uma base de dados que consiste em regras que dizem quando as ações devem ser executadas;
- Interfaces: fornece interfaces para os consumidores de contexto.

Camada 4: camada de raciocínio e decisão, o núcleo da plataforma CoCA, é o lugar onde o raciocínio e as decisões conscientes de contexto são executadas. Fornece o núcleo de serviço de consciência a contexto após o raciocínio sobre os componentes. Trata-se do motor de ação RAID (Reasoning, Aggregation, Interpretation, Decision and Action) que preenche a ontologia com os dados de contexto e, em seguida, aplica-se as regras e axiomas para o raciocínio e decisão sobre as ações a serem executadas. Outra tarefa que é realizada nesta camada é a agregação, através da combinação de dois ou mais contextos de baixo nível para gerar um contexto de alto nível, o qual é mais significativo. Um exemplo de agregação realizada poder ser a combinação das informações referentes à temperatura corporal, frequência cardíaca e pressão arterial de um paciente para conseguir decidir o estado de saúde do paciente. Nesta camada também se encontram alguns serviços suplementares, tais como, serviços de descoberta de conhecimento, o processo de adicionar recursos de forma a aumentar à capacidade da plataforma, as tarefas relacionadas à privacidade e gerenciamento de segurança de serviço da plataforma.

Camada 5: camada de aplicação, é nesta camada onde as ações são desencadeadas sendo que elas podem ser de forma reativa ou proativa.

5.8 Análise dos Trabalhos

A tabela 3 apresenta uma análise comparativa entre os trabalhos discutidos nesta seção, mostrando as técnicas empregadas nas etapas de aquisição e processamento para as diferentes estratégias de modelagem.

Na etapa de Aquisição foi realizada a seguinte classificação de acordo com as fontes de contexto utilizadas, ou seja, os tipos de sensores suportados por cada solução: (F) denota que a solução suporta apenas sensores físicos, (T) denota que a solução oferece suporte a todos os tipos de fontes de dados (sensores físicos, lógicos e virtuais), e (-) representa que não foi identificado o tipo de sensor suportado.

Para caracterização das técnicas de raciocínio de contexto utilizadas por cada projeto, na etapa de Processamento, foram utilizadas as seguintes abreviaturas: regras (R), aprendizagem (A), e baseado em ontologias (O).

Na etapa de modelagem foram caracterizadas as técnicas utilizadas, na tabela as abreviaturas possuem o seguinte significado: (C) modelo chave-valor, (M) modelo baseado em linguagem de marcação, (G) modelo gráfico, (OO) modelo orientado a objetos, (L) modelo baseado em lógica, (On) modelo baseado em ontologia e (Re) modelo relacional.

Tabela 3
Análise comparativa dos trabalhos em relação as etapas de um sistema consciente de contexto

Projeto	Aquisição	Processamento	Modelagem
MidSen	F	R	C
Aura	T	R	M
SCONSTREAM	F	R	G
COSMOS	F	R	OO
The Use of Situation Theory in Context Modeling	-	R	L
CoBrA	T	R, O	On
CoCA	T	R, A, O	On, Re

Com base na tabela 3 é constatado que na etapa de Aquisição de Contexto as informações contextuais utilizam como fonte de dados todos os tipos de sensores (físicos, virtuais e lógicos), ou somente sensores físicos denotando que há uma maior demanda pela aquisição de dados oriundos do ambiente físico.

Em uma outra análise destaca-se o fato de todos os trabalhos examinados utilizarem na sua etapa de Processamento de Contexto a técnica baseada em regras, principalmente devido as suas características como, simplicidade e baixa utilização de recursos computacionais.

Sob outra perspectiva verifica-se que os trabalhos com suporte apenas para sensores físicos são mais focados em aplicações específicas, enquanto que os outros que utilizam diferentes tipos de sensores podem atender aplicações de diferentes naturezas.

6 Considerações Finais

Este artigo apresentou uma revisão sobre sistemas conscientes de contexto, mostrando as principais etapas da consciência de contexto, sendo elas aquisição, modelagem, e processamento de contexto. Na etapa de aquisição foram apresentadas as principais fontes utilizadas de forma a adquirir os dados de contexto, destacando os pontos positivos e negativos de cada fonte.

A etapa de processamento de contexto possui grande importância para aplicações conscientes de contexto, já que é responsável por realizar raciocínio sobre os dados de contexto adquiridos, e assim deduzir informações significativas, que auxiliem as aplicações conscientes de contexto. Foram apresentadas as principais técnicas utilizadas, mostrando suas principais características e vantagens em sua utilização, onde se destaca a utilização da técnica baseada em regras, devido a sua simplicidade, e por ser a técnica mais utilizada nesta etapa.

Na etapa de modelagem de contexto, destacaram-se as principais estratégias utilizadas, mostrando suas principais características como também as vantagens e desvantagens em sua utilização. Destacam-se os modelos híbridos que vêm ganhando destaque, combinando dois ou mais modelos, de forma a aproveitar os pontos positivos de cada estratégia, tendo como intuito o aprimoramento da etapa de modelagem de contexto, e com isso, facilitar a utilização das informações contextuais.

Foram apresentados trabalhos que utilizam diferentes estratégias para realizar a modelagem do contexto, mostrando suas arquiteturas, e características. Onde foi realizada uma comparação entre os trabalhos, analisando as três etapas presentes na consciência de contexto, destacando as técnicas empregadas em cada trabalho.

Referências

[1] John Krumm. *Ubiquitous Computing Fundamentals*. Chapman & Hall/CRC, 1st edition, 2009.

[2] Mark Weiser. The computer for the 21st century. *Scientific American*, 265(3):66–75, January 1991.

[3] P. Bellavista, A. Corradi, M. Fanelli, and L. Foschini. A survey of context data distribution for mobile ubiquitous systems. *ACM Comput. Surv.*, 44(4):24:1–24:45, September 2012.

[4] Alan José de Moura Silva Filho. Estudo sobre a computação ubíqua e aplicações para tv digital. Monografia de graduação bacharelado em ciência da computação, Universidade Federal de Pernambuco/UFPE, Recife/PE, 2010.

[5] Márcia Zechlinski Gusmão. Uma arquitetura de software direcionada à consciência de contexto na ubicomp. Dissertação de mestrado em ciência da computação, PPGC/UFPel, Pelotas-RS, 2013.

[6] M. Knappmeyer, S.L. Kiani, E.S. Reetz, N. Baker, and R. Tonjes. Survey of context provisioning middleware. *Communications Surveys Tutorials, IEEE*, 15(3):1492–1519, Third 2013.

[7] C. A. COSTA. *Continuum: A Context-aware Service-based Software Infrastructure for Ubiquitous Computing*. PhD thesis, UFRGS, Porto Alegre-RS, 2008.

[8] A. K. Dey. Understanding and using context. *Personal and Ubiquitous Computing*, 5:4–7, 2001.

[9] A. M. F. Pernas. *Sensibilidade à Situação em Sistemas Educacionais na Web*. Tese de doutorado em ciência da computação, Instituto de Informática-UFRGS, Porto Alegre-RS, 2012.

[10] P. Brézillon. Context in problem solving: a survey. *Knowl. Eng. Rev.*, 14(1):47–80, May 1999.

[11] V. Vieira, D. Souza, A. C. Salgado, and P. Tedesco. *Uso e Representação de Contexto em Sistemas Computacionais*, chapter 1, pages 127–166. UFSCAR, 2006.

[12] K. Henricksen and J. Indulska. Developing context-aware pervasive computing applications: Models and approach. *Pervasive and Mobile Computing, In*, 2:2005, 2005.

[13] Thomas Strang and Claudia Linnhoff-Popien. A context modeling survey. In *In: Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing, Nottingham/England*, 2004.

[14] Claudio Bettini, Jadwiga Indulska, Daniela Nicklas, Anand Ranganathan, Daniele Riboni, and et al. A survey of context modelling and reasoning techniques, 2008.

[15] Charith Perera, Arkady B. Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. Context aware computing for the internet of things: A survey. *CoRR*, abs/1305.0982, 2013.

[16] Daniela Petrelli, Elena Not, Carlo Strapparava, Oliviero Stock, and Massimo Zancanaro. Modeling context is like taking pictures, 2000.

[17] Roberto De Virgilio and Riccardo Torlone. A general methodology for context-aware data access. In Vijay Kumar, Arkady B. Zaslavsky, Ugur Çetintemel, and Alexandros Labrinidis, editors, *MobiDE*, pages 9–15. ACM, 2005.

[18] Daniel H. Wilson, Anna C. Long, and Chris Atkeson. A context-aware recognition survey for data collection using ubiquitous sensors in the home. In Gerrit C. van der Veer and Carolyn Gale, editors, *CHI Extended Abstracts*, pages 1865–1868. ACM, 2005.

[19] Chia-Hsing Hou, Hung-Chang Hsiao, Chung-Ta King, and Chun-Nan Lu. Context discovery in sensor networks. In *ITRE*, pages 2–6. IEEE, 2005.

[20] Sungjin Ahn and Daeyoung Kim 0001. Proactive context-aware sensor networks. In Kay Römer, Holger Karl, and Friedemann Mattern, editors, *EWSN*, volume 3868 of *Lecture Notes in Computer Science*, pages 38–53. Springer, 2006.

[21] Pankesh Patel, Sunil Jardosh, Sanjay Chaudhary, and Prabhat Ranjan. Context aware middleware architecture for wireless sensor network. In *IEEE SCC*, pages 532–535. IEEE Computer Society, 2009.

[22] Albert Held, Sven Buchholz, Alexander Schill, and Er Schill. Modeling of context information for pervasive computing applications, 2002.

[23] G. Klyne, F. Reynolds, C. Woodrow, H. Ohto, J. Hjelm, M. H. Butler, and L. Tran. *Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0. W3C Recommendation, W3C, January 2004.* 2004.

[24] David Garlan, Daniel P. Siewiorek, and Peter Steenkiste. Project aura: Toward distraction-free pervasive computing. *IEEE Pervasive Computing*, 1:22–31, 2002.

[25] L. Capra, W. Emmerich, and C. Mascolo. Carisma: context-aware reflective middleware system for mobile applications. *Software Engineering, IEEE Transactions on*, 29(10):929 – 945, oct. 2003.

[26] L. Sanchez, J. Lanza, R. Olsen, M. Bauer, and M. Girod-Genet. A generic context management framework for personal networking environments. *Mobile and Ubiquitous Systems, Annual International Conference on*, 0:1–8, 2006.

[27] John Herbert, John O'Donoghue, and Xiang Chen. A context-sensitive rule-based architecture for a smart building environment. In *FGCN (2)*, pages 437–440. IEEE Computer Society, 2008.

[28] Eike S. Reetz, Ralf Tönjes, and Nigel Baker. Towards global smart spaces: Merge wireless sensor networks into context-aware systems. In *Proceedings of the 5th IEEE International Conference on Wireless Pervasive Computing*, ISWPC'10, pages 337–342, Piscataway, NJ, USA, 2010. IEEE Press.

[29] Antonio Corradi, Mario Fanelli, and Luca Foschini. Implementing a scalable context-aware middleware. In *ISCC*, pages 868–874. IEEE, 2009.

[30] Adel Shaeib, Paolo Cappellari, and Mark Roantree. A framework for real-time context provision in ubiquitous sensing environments. In *ISCC*, pages 1083–1085. IEEE, 2010.

[31] Fei Li, Sanjin Sehic, and Schahram Dustdar. Copal: An adaptive approach to context provisioning. In *WiMob*, pages 286–293. IEEE, 2010.

[32] Patrick Brézillon. Task-realization models in contextual graphs. In Anind K. Dey, Boicho N. Kokinov, David B. Leake, and Roy M. Turner, editors, *CONTEXT*, volume 3554 of *Lecture Notes in Computer Science*, pages 55–68. Springer, 2005.

[33] Karen Henricksen and Jadwiga Indulska. A software engineering framework for context-aware pervasive computing. In *In: PerCom, IEEE Computer Society*, pages 77–86. IEEE Computer Society, 2004.

[34] Christof Simons. Cmp: A uml context modeling profile for mobile distributed systems. In *HICSS*, page 289. IEEE Computer Society, 2007.

[35] Karen Henricksen, Jadwiga Indulska, and Andry Rakotonirainy. Modeling context information in pervasive computing systems. In *Pervasive Computing*, pages 167–180. Springer Berlin Heidelberg, 2002.

[36] Manasawee Kaenampornpan and Eamonn O'Neill. An integrated context model: Bringing activity to context. *Proc. Workshop on Advanced Context Modelling, Reasoning and Management*, 2004.

[37] Joëlle Coutaz, James L. Crowley, Simon Dobson, and David Garlan. Context is key. *Commun. ACM*, 48(3):49–53, March 2005.

[38] Keith Cheverst, Keith Mitchell, and Nigel Davies. Design of an object model for a context sensitive tourist guide. *Computers & Graphics*, 23(6):883–891, 1999.

[39] Jakob E. Bardram. The java context awareness framework (jcaf) - a service infrastructure and programming framework for context-aware applications. In Hans-Werner Gellersen, Roy Want, and Albrecht Schmidt, editors, *Pervasive*, volume 3468 of *Lecture Notes in Computer Science*, pages 98–115. Springer, 2005.

[40] Denis Conan, Romain Rouvoy, and Lionel Seinturier. Scalable processing of context information with cosmos. In Jadwiga Indulska and Kerry Raymond, editors, *DAIS*, volume 4531 of *Lecture Notes in Computer Science*, pages 210–224. Springer, 2007.

[41] Pravin Pawar, Hanga Boros, Fei Liu, Geert J. Heijenk, and Bert-Jan van Beijnum. Bridging context management systems in the ad hoc and mobile environments. In *ISCC*, pages 882–888. IEEE, 2009.

[42] Waskitho Wibisono, Arkady B. Zaslavsky, and Sea Ling. Comihoc: A middleware framework for context management in manet environment. In *AINA*, pages 620–627. IEEE Computer Society, 2010.

[43] Paula Cibele Cavalcante Fernandes. Ubifex: Uma abordagem para modelagem de características de linha de produtos de software sensíveis ao contexto. Master's thesis, UFRJ/ COPPE/ Programa de Engenharia de Sistemas e Computação, 2009.

[44] Varol Akman and Mehmet Surav. The use of situation theory in context modeling. *Computational Intelligence*, 13(3):427–438, 1997.

[45] Jean Bacon, John Bates, and David Halls. Location-oriented multimedia. *IEEE Personal Commun.*, 4(5):48–57, 1997.

[46] Philip D. Gray and Daniel Salber. Modelling and using sensed context information in the design of interactive applications. In Murray Reed Little and Laurence Nigay, editors, *EHCI*, volume 2254 of *Lecture Notes in Computer Science*, pages 317–336. Springer, 2001.

[47] Eleftheria Katsiri and Alan Mycroft. Knowledge-representation and scalable abstract reasoning for sentient computing using first-order logic. First Workshop on Challenges and Novel Applications for Automated Reasoning, 2003.

[48] Thomas R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, June 1993.

[49] Thomas Strang, Claudia Linnhoff-Popien, and Korbinian Frank. Cool: A context ontology language to enable contextual interoperability. In Jean-Bernard Stefani, Isabelle Demeure, and Daniel Hagimont, editors, *Distributed Applications and Interoperable Systems*, volume 2893 of

Lecture Notes in Computer Science, pages 236–247. Springer Berlin / Heidelberg, 2003.

[50] Harry Chen. An intelligent broker for context-aware systems. In *In Adjunct Proceedings of Ubicomp*, pages 183–184, 2003.

[51] Tao Gu, Hung Keng Pung, and Da Qing Zhang. A service-oriented middleware for building context-aware services. *J. Netw. Comput. Appl.*, 28(1):1–18, January 2005.

[52] Carsten Jacob, David Linner, Stephan Steglich, and Ilja Radusch. Bio-inspired context gathering in loosely coupled computing environments. In *Proceedings of the 1st International Conference on Bio Inspired Models of Network, Information and Computing Systems*, BIONETICS '06, New York, NY, USA, 2006. ACM.

[53] Fano Ramparany, Remco Poortinga, Maja Stikic, Joerg Schmalenstroeer, and Thorsten Prante. An open context information management infrastructure the ist-amigo project. In *3rd IET International Conference on Intelligent Environments (IE 2007)*, pages 398–403, sept. 2007.

[54] S. Pietschmann, A Mitschick, R. Winkler, and K. Meissner. Croco: Ontology-based, cross-application context management. In *Semantic Media Adaptation and Personalization, 2008. SMAP '08. Third International Workshop on*, pages 88–93, Dec 2008.

[55] Daniele Riboni and Claudio Bettini. Context-aware activity recognition through a combination of ontological and statistical reasoning. In Daqing Zhang, Marius Portmann, Ah-Hwee Tan, and Jadwiga Indulska, editors, *UIC*, volume 5585 of *Lecture Notes in Computer Science*, pages 39–53. Springer, 2009.

[56] Xiaoming Zhou, Xinhuai Tang, Xiaozhou Yuan, and Delai Chen. Spbca: Semantic pattern-based context-aware middleware. In *ICPADS*, pages 891–895. IEEE, 2009.

[57] Damiao Ribeiro de Almeida, Cláudio de Souza Baptista, and Fabio Gomes de Andrade. Using ontologies in context-aware applications. In *DEXA Workshops*, pages 349–353. IEEE Computer Society, 2006.

[58] Karen Henricksen, Steven Livingstone, and Jadwiga Indulska. Towards a hybrid approach to context modelling, reasoning and interoperation. In *Proceedings of First International Workshop on Advanced Context Modelling, Reasoning And Management*, 2004.

[59] R.C.A. da Rocha, M.A. Casanova, and M. Endler. Promoting efficiency and separation of concerns through a hybrid model based on ontologies for context-aware computing. In *Pervasive Computing and Communications Workshops, 2007. PerCom Workshops '07. Fifth Annual IEEE International Conference on*, pages 9–13, March 2007.

[60] D. Ejigu, M. Scuturici, and L. Brunie. Coca: A collaborative context-aware service platform for pervasive computing. In *Information Technology, 2007. ITNG '07. Fourth International Conference on*, pages 297–302, April 2007.

[61] Deirdre Lee and René Meier. A hybrid approach to context modelling in large-scale pervasive computing environments. In *Proceedings of the Fourth International ICST Conference on COMmunication System softWAre and middlewaRE*, COMSWARE '09, pages 14:1–14:12, New York, NY, USA, 2009. ACM.

[62] Igor Kotenko, Olga Polubelova, and Igor Saenko. The ontological approach for siem data repository implementation. In *Proceedings of the 2012 IEEE International Conference on Green Computing and Communications*, GREENCOM '12, pages 761–766, Washington, DC, USA, 2012. IEEE Computer Society.

[63] Jae-Hun Kim Oje Kwon, Yong-Soo Song and Ki-Joune Li. Sconstream: A spatial context stream processing system. *IEEE Computer Society*, 1:165–170, 2010.

[64] H. CHEN, T. FININ, A. JOSHI, and Y PENG. Umbc ebiquity project: Context broker architecture (cobra). acesso em outubro de 2014, 2005. Disponível em: <http://ebiquity.umbc.edu/project/html/id/1//>.

[65] H CHEN. *An Intelligent Broker Architecture for Pervasive Context-Aware Systems*. Phd thesis, Faculty of the Graduate School of the University of Maryland, 2004.

[66] Jena. A free and open source java framework for building semantic web and linked data applications. acesso em outubro de 2014, 2014. Disponível em: <http://jena.apache.org/>.

[67] Jess. Jess - the expert system shell for the java platform. acesso em outubro de 2014, 2014. Disponível em: <http://herzberg.ca.sandia.gov/jess/>.

[68] D. POOLE, R. GOEBEL, and ALELIUNAS. Theorist. acesso em outubro de 2014, 1998. Disponível em: <http://www.cs.ubc.ca/~poole/theorist>.

[69] D. Ejigu. *Context Modeling and Collaborative Context-Aware Services for Pervasive Computing*. Doctoral school of computer and information sciences (ediis) affiliated area: Computer science, National Institute of Applied Sciences, Insa de Lyon, 2007.

[70] Dejene Ejigu, Marian Scuturici, and Lionel Brunie. Hybrid approach to collaborative context-aware service platform for pervasive computing. *Journal of computers*, page 40, 2008.

Roger Machado Formado em Ciência da Computação na Universidade Federal de Pelotas no ano de 2013. Atualmente é aluno do Programa de Pós-Graduação em Computação da Universidade Federal de Pelotas e bolsista CAPES. Esforços de estudo e pesquisa voltados para as áreas de Computação Ubíqua, Consciência de Contexto, Mineração de Dados e Segurança da Informação.

Patrícia Davet Possui graduação em Engenharia Eletrônica pela Universidade Católica de Pelotas (2012). Atualmente é aluna do Programa de Pós-Graduação em Computação da Universidade Federal de Pelotas e bolsista CAPES.Tem interesse na área de Engenharia Eletrônica, com ênfase em Computação Autonômica e Instrumentação Virtual Ubíqua.