UNISINOS

Programa Interdisciplinar de Pós-Graduação em

Computação Aplicada

Mestrado Acadêmico

Fausto Girola Junior

UDeal: Um Modelo Descentralizado para Identificação de Oportunidades de Negócio Usando Trilhas

São Leopoldo, 2014

Fausto Girola Junior

U-DEAL:UM MODELO DESCENTRALIZADO PARA IDENTIFICAÇÃO DE OPORTUNIDADES DE NEGÓCIO USANDO TRILHAS

Dissertação apresentada como requisito parcial para a obtenção do título de Mestre pelo Programa de Pós-Graduação em Computação Aplicada da Universidade do Vale do Rio dos Sinos — UNISINOS

Orientador:
Prof. Dr. Jorge Luis Victória Barbosa

São Leopoldo
2014

G526u Girola Junior, Fausto

U-Deal: um modelo descentralizado para identificação de oportunidades de negócio usando trilhas / Fausto Girola Junior -- 2014.

86 f. : il. color. ; 30cm.

Dissertação (mestrado em Computação Aplicada) -- Universidade do Vale do Rio dos Sinos, Programa de Pós-Graduação em Computação Aplicada, São Leopoldo, RS, 2014.

Orientador: Prof. Dr. Jorge Luis Victória Barbosa.

1. Ciência da computação. 2. Computação ubíqua - Descentralização. 3. Comércio ubíquo. 4. Negociação ubíqua. I. Título. II. Barbosa, Jorge Luis Victória.

CDU 004

Catalogação na Publicação: Bibliotecária Eliete Mari Doncato Brasil - CRB 10/1184

AGRADECIMENTOS

Ao meu orientador, Professor Jorge Luis Victória Barbosa, por sua dedicação e apoio nesses dois anos.

À coordenação, aos professores, às secretárias e aos colegas do Programa de Pós-Graduação em Computação Aplicada da Unisinos, em especial aos membros do Mobilab.

Aos meus pais, Fausto e Rosângela, e à minha esposa, Maristela, pela compreensão, pela paciência, pelo amor e pelo apoio incondicionais.

RESUMO

Em um cenário de popularização de dispositivos móveis com alto poder de processamento e conectividade e de aumento na disponibilidade de redes sem fio e serviços em nuvem, é cada vez maior o número de aplicações distribuídas, nas mais diversas áreas. Esses sistemas precisam lidar com novos tipos de problemas advindos da natureza móvel e/ou ubíqua de seus nodos computacionais. Cada vez mais, os aplicativos em dispositivos móveis precisam examinar as informações de seu contexto e histórico de ações, a fim de permitir uma rápida adaptação, exigindo mínima distração do usuário e grande usabilidade. Além disso, o modelo cliente-servidor demonstra limitações para atender os requerimentos dos novos sistemas. Este trabalho apresenta o U-Deal, um modelo multiagente descentralizado para comércio ubíquo. O modelo descreve um sistema *Peer-to-peer* sensível ao contexto, em que se destaca um *engine* para análise de trilhas e um gerador de perfil dinâmico. O objetivo do modelo é suportar a recomendação automática de oportunidades de negócio, com base no perfil, preferências e histórico de atividades dos usuários. Neste trabalho, são apresentados uma especificação do sistema, utilizando a metodologia Prometheus, e um protótipo, incluindo uma avaliação através da execução de cenários simulados.

Palavras-chave: Descentralização. Peer-to-peer. Pervasivo. Comércio Ubíquo. Negociação Ubíqua.

ABSTRACT

In a scenario of massive adoption of powerfull and highly connected mobile devices, cloud computing and improved wi-fi networks, is rapidly increasing the number of distributed application, in several areas. These systems need to handle new kind of problems, deriving from the mobile and/or ubiquitous nature of their computational nodes. Also, more and more mobile devices applications need to check context and historical information to adapt better to situations and provide minimum distraction and great usability to end user. Besides, the client-server model shows limitations to handle the new systems requirements. This work presents the U-Deal, a decentralized and multi-agent ubiquitous commerce model, that describes a Peer-to-peer context aware system, with a distributed historical context (trail) engine and a dynamic profile generator. The aim of the model is to provide an automatic deal opportunity recommendation based on user profile, preferences and activitiy history in a peer-to-peer environment. In this text, we present a system specification and a prototype, including simulated u-commerce scenarios.

Keywords: Decentralization. Peer-to-peer. Pervasive. Ubiquitous Commerce.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

1.1 Motivação

Vivemos numa época de mudança de paradigma na comunicação. A Internet alterou a relação das pessoas com a sociedade e com o conhecimento. Assim como o rádio e a televisão, em meados dos século XX, a Internet provocou (e ainda provoca) um profundo impacto nas relações sociais ao conectar o mundo, acelerando a troca de informações, modificando usos e costumes e democratizando a produção e o acesso ao saber.

Um dos desdobramentos mais recentes do impacto da Internet na sociedade é o aparecimento das redes sociais. Essas, além de aprofundar a influência da Internet sobre a vida das pessoas, impelem milhões de usuários todos os anos não apenas a compartilhar informações na rede, mas também a consumir produtos e serviços *online*. É possível mesmo afirmar que as redes sociais estão entre os principais fatores de inclusão digital nos dias de hoje, pois permitem a criação de novos espaços sociais de grandes proporções, frequentados por todo o tipo de usuários, não necessariamente profissionais ou entusiastas da tecnologia.

Em relação ao *hardware*, a partir dos anos 80 e 90, o crescimento da Internet foi impulsionado, entre outros fatores, pela popularização do PC (*Personal Computer*). Este foi, por décadas, o dispositivo cliente de Internet disponível. Contudo, desde o início do século XXI, uma forte convergência entre telefonia móvel e computação culminou no desenvolvimento dos *smartphones* (ainda que desde os anos 90 do século XX já existissem celulares com certos recursos de *smartphones*). Ainda que outros dispositivos móveis como os notebooks e PDA's (*Personal Digital Assistant*) tenham popularizado-se, são os *smartphones* que demonstram uma maior penetração entre os usuários. Além disso, ao contrário dos notebooks, os *smartphones* herdaram dos celulares um caráter de dispositivo móvel e "pessoal", fazendo destes candidatos ideais para protagonizar uma revolução: a Computação Móvel. Antes dos *smartphones*, os dispositivos móveis disponíveis eram limitados, tanto em termos de hardware quanto de software. Por exemplo, possuíam limitado acesso a redes sem fio (quando possuíam algum), baixo poder de processamento e memória, restrita autonomia de bateria, pouquíssimos aparelhos possuíam GPS. Não estavam ainda prontos para a Computação Móvel.

Dessa forma, é possível observar duas tendências nos últimos anos: a Internet recebe milhares de novos usuários todos os anos, para acessar as redes sociais e outros serviços *online*; é crescente e já muito significativo o número de utilizadores que acessam a Internet e as redes sociais através de dispositivos móveis. Este cenário demonstra que os *smartphones* já são dispositivos eficientes para o suporte à Computação Móvel, preparando a sociedade para a Computação Ubíqua. De fato, Weiser et al. (1998) já haviam discutido sobre isso ao afirmarem que a ideia de "computador pessoal", na Computação Ubíqua, não era adequada e que a visão a respeito da utilização de *laptops*, *tablets* e outros "navegadores de conhecimento"seria apenas uma fase de transição na direção da verdadeira ubiquidade, pois estes dispositivos não tinham poten-

cial para tornarem-se invisíveis. A tendência é de que o foco da computação mude de aparelhos pessoais com alta conectividade para a integração de dispositivos "inteligentes"e com acesso a informações sobre seu próprio contexto em artefatos do dia a dia: nas roupas, nas casas, nos carros e nas ruas (SATYANARAYANAN, 2001).

Nesse quadro de proliferação de dispositivos móveis com alto poder de processamento e conectividade, de popularização dos serviços em nuvem e de aumento na disponibilidade de redes sem fio públicas ou privadas (incluindo os serviços 3G e 4G), abriu-se espaço para um cenário de computação distribuída nunca antes visto. Serviços que antes eram baseados em uma arquitetura cliente-servidor já podem ser implementados através de sistemas heterogêneos, onde os nós computacionais podem ter diferentes arquiteturas ou diferentes *vendors*. Além disso, existe um potencial para o desenvolvimento de sistemas distribuídos onde os dispositivos móveis podem trocar informações diretamente entre si (*peer-to-peer*) e mesmo obter dados de outros tipos de dispositivos mais simples, como sensores por exemplo. Essa arquitetura favorece a implementação de Sistemas Sensíveis ao Contexto (DEY, 2001).

Essa descentralização (que não implica na ausência de acesso a servidores ou serviços em nuvem quando necessário), aliada à utilização de dispositivos com maior capacidade de processamento e de comunicação, favorece a implementação de tarefas de escopo local. Por exemplo, a descoberta e o compartilhamento de recursos disponíveis em uma rede local ou *ad hoc* pode ser feita diretamente entre os dispositivos, sem a necessidade de que um serviço centralizado identifique, através de localização GPS, que dois dispositivos estão fisicamente próximos e dispare um alerta aos interessados. Essa é uma das propriedades da Computação Ubíqua e foi apresentada por Satyanarayanan (2001) como Escalabilidade Localizada. Segundo o autor, a Escalabilidade Localizada é a capacidade de priorização de interação com usuários mais próximos. Em ambientes descentralizados, baseados em redes *ad hoc*, a própria topologia da rede favorece a escalabilidade.

A relação entre Escalabilidade Localizada e poder de processamento dos dispositivos móveis é analisada no trabalho de Xiao et al. (2013). Os autores propõem um modelo descentralizado para suporte a *mobile crowdsensing*, baseado nas *cloudlets* propostas por Satyanarayanan (2010). O termo *mobile crowdsensing* aplica-se a sistemas computacionais onde usuários de dispositivos móveis, providos de sensores e câmeras, coletivamente (de forma ativa ou passiva) compartilham informações a fim de avaliar algum fenômeno de interesse comum. Um exemplo de aplicação é o *CreekWatch*, desenvolvido pelo IBM Almaden Research Center. O sistema monitora o nível e a qualidade da água de rios, agregando e analisando os relatos dos usuários, normalmente baseados em fotos e mensagens de texto (GANTI; YE; LEI, 2011).

Esse cenário tecnológico vem gerando novas oportunidades em diversas áreas, tais como Educação (BARBOSA et al., 2011), Acessibilidade (TAVARES et al., 2012) e Entretenimento (SEGATTO et al., 2008). As relações comerciais e de negócios, assim como outras áreas de conhecimento, também usufruem dessas novas tecnologias para melhorar suas práticas e abordagens. A aplicação dessas tecnologias no aperfeiçoamento das estratégias comerciais ocasionou

o surgimento de uma frente de pesquisa denominada Comércio Ubíquo (GERSHMAN, 2004). Atualmente existem diversos modelos para suporte ao comércio ubíquo, como o GTTracker (CAZAROTTO, 2013), o MUCS (FRANCO et al., 2011), o PAM (LIN; YU; SHIH, 2005) e os modelos de Sanchez-Pi e Molina (2009) e Lin et al. (2011).

Os modelos de comércio ubíquo, em geral, estão baseados no modelo cliente-servidor (FRANCO et al., 2011). Contudo, tal abordagem apresenta limitações para a implementação de sistemas de acordo com a definição de Computação Ubíqua apresentada por Satyanarayanan (2001). Segundo o autor, a computação ubíqua requer redes móveis, alta disponibilidade, escalabilidade localizada e acesso móvel a informação. Já os modelos descentralizados suportam mais facilmente tais características, pois a comunicação é feita de forma direta entre os usuários, não necessitando de infraestruturas pré-configuradas e instaladas para suportar o serviço. Além disso, os sistemas descentralizados suportam redes *ad hoc*, onde é possível a criação de redes móveis, e possuem alta disponibilidade, já que se um nodo da rede ficar inoperante somente as tarefas que dependem desse nodo ficarão comprometidas.

Contudo, se por um lado a arquitetura descentralizada favorece determinados aspectos úteis à Computação Ubíqua, por outro lado traz desvantagens. Menor tolerância a falhas e escalabilidade limitada, gerenciamento e configuração não centralizados, dificuldade ou impossibilidade de executar balanceamento de carga, tomada de decisão com base em um volume limitado de dados, são algumas das desvantagens dos sistemas descentralizados.

Além da Escalabilidade Localizada, Satyanarayanan (2001) aborda ainda outros atributos desejáveis para a Computação Ubíqua. Dentre estes, é possível destacar a *invisibilidade* de um sistema. Para o autor, *invisibilidade* é o completo desaparecimento dos sistemas computacionais da consciência das pessoas. Idealmente, o que se deseja é a *distração mínima do usuário*. Quando um ambiente de Computação Ubíqua continuamente atende as expectativas, raramente apresentando alertas e surpresas, o usuário vai interagir com o sistema apenas na periferia da sua consciência, na maior parte do tempo.

Um sistema só pode se tornar invisível (atuar apenas na periferia da atenção do usuário) se for proativo e para tanto precisa conhecer as intenções do usuário. Para que um sistema possa inferir as intenções de um usuário ele tem que conhecer o perfil desse usuário e o contexto onde está inserido. Além disso, pode ser importante para o sistema conhecer o estado e as ações tomadas anteriormente por esse usuário no mesmo contexto ou em diferentes contextos. Isso requer que o sistema rastreie a localização e as ações do usuário e armazene essas informações em um histórico de contextos. Um histórico de contextos visitados por um usuário durante um período é uma Trilha (SILVA et al., 2009) . Dessa forma, um sistema sensível a trilhas complementa um sistema apenas sensível ao contexto, pois fornece novas dimensões para análise da intenção dos usuários.

De acordo com tal cenário social e tecnológico, existe muito espaço para a pesquisa de sistemas de Computação Ubíqua que encontrem alternativas para os problemas de escalabilidade, poder de processamento limitado dos dispositivos móveis, alta latência de rede (principalmente

em sistemas na nuvem através de WAN) e sensibilidade ao contexto. Além disso, trabalhos como de Satyanarayanan (2010) e Youssef (2013) procuram entender como a Computação Ubíqua pode beneficiar-se da disponibilidade de serviços em nuvem, inclusive em modelos distribuídos e descentralizados.

Sendo assim, este trabalho descreve o U-Deal, um modelo computacional sensível ao contexto e com suporte a trilhas descentralizadas para a identificação de oportunidades de negócios, através da negociação entre agentes.

1.2 Definição do Problema e Questão de Pesquisa

O número de usuários *online* cresce em larga escala, incluindo os usuários móveis. A computação baseada na modelo cliente-servidor estimula o uso de arquitetura em nuvem. Diversos sistemas e aplicativos para Computação Móvel tendem a centralizar a descoberta de serviços e o processamento principal (mineração de dados, recomendação) em servidores na nuvem (o MUCS (FRANCO et al., 2011), o PAM (LIN; YU; SHIH, 2005), os modelos de Sanchez-Pi e Molina (2009) e Lin et al. (2011), o UbiCloud (YOUSSEF, 2013)). Além disso, a crescente disponibilidade da Internet e de conexões *ad hoc* através de redes sem fio, juntamente com a sofisticação dos dispositivos móveis atuais permitem que uma pessoa esteja online 24h por dia, quase em qualquer lugar. Sendo assim, a questão de pesquisa que guia este trabalho é a seguinte: **Como seria um sistema para gerar oportunidades de negócio através de negociação entre agentes, utilizando Trilhas distribuídas?**

1.3 Objetivos

A principal contribuição científica deste trabalho é a modelagem de um sistema *peer-to-peer* sensível ao contexto que utilize trilhas distribuídas.

Sendo assim, o objetivo deste trabalho é propor um modelo computacional aplicado à identificação de oportunidades de negócios, utilizando trilhas distribuídas. A identificação dessas oportunidades deve ser feita através de negociação entre agentes, utilizando informações do perfil dos usuários e de suas trilhas. Essa negociação deve ser executada de forma distribuída e descentralizada, diretamente entre os dispositivos conectados em uma rede, não utilizando uma arquitetura cliente/servidor.

Além disso, este trabalho propõe-se a:

- especificar um modelo de sistema multiagente descentralizado utilizando metodologia de engenharia de software orientada a agentes;
- desenvolver um protótipo, implementando os principais conceitos propostos no modelo;
- avaliar a aplicabilidade funcional do modelo, usando o protótipo como base, através da simulação de cenários.

1.4 Metodologia

A fim de atingir os objetivos deste trabalho, a primeira etapa foi um estudo geral sobre Computação Ubíqua e Comércio Ubíquo.

A segunda etapa foi o estudo de modelos já propostos para sistemas de Comércio Ubíquo. O objetivo desta etapa foi realizar um estudo comparativo entre eles, analisando sua arquitetura e mecanismos de recomendação de oportunidades.

A terceira etapa foi descrever o modelo U-Deal, apresentando sua arquitetura geral, principais elementos e como estes interagem. Além disso, foi analisado como cada um desses elementos contribuiu para que o modelo alcançasse seus objetivos.

Após especificação do modelo, foi implementado um protótipo a fim de validar o mesmo.

Por fim, com base no modelo e no protótipo, a terceira etapa foi a avaliação do modelo, incluindo testes funcionais e de desempenho, através da simulação de cenários.

1.5 Organização do Trabalho

Esta dissertação está organizada da seguinte maneira. O Capítulo 2 descreve os principais conceitos referentes às áreas relacionadas a este trabalho. O Capítulo 3 descreve trabalhos relacionados à área de comércio ubíquo, apresentando como o trabalho proposto se enquadra nesta área. No Capítulo 4, é apresentado o modelo U-Deal para comércio ubíquo descentralizado. O Capítulo 5 apresenta os aspectos de implementação e avaliação. Por fim, o capítulo 6 apresenta as considerações finais.

2 CONCEITOS BÁSICOS

Neste capítulo, serão abordados os principais temas relacionados ao modelo proposto como Computação Ubíqua e Sensibilidade ao Contexto, bem como o conceito de arquitetura descentralizada e Comércio Ubíquo. Além disso, o capítulo trata brevemente sobre Sistemas de Recomendação para Computação Móvel e de que forma esse tema relaciona-se com a dissertação.

2.1 Computação Ubíqua

A Computação Ubíqua representa um conjunto de tendências relacionadas não apenas à arquitetura de hardware e software, mas, principalmente, a uma nova visão de mundo e de relacionamento das pessoas com a tecnologia. Nos anos 90, Weiser (1991) e sua equipe da XEROX PARC já observavam que o poder de processamento dos computadores aumentava e seu tamanho diminuía a cada ano, assim como a proporção de computadores por pessoa também crescia. Estava claro que a miniaturização dos computadores e a proliferação de redes sem fio levaria a uma quebra de paradigma na computação.

Segundo Weiser et al. (1998), a Computação Ubíqua representaria a terceira era na computação, sendo o Mainframe o representante da primeira era e o PC o representante da segunda era. Para os autores, a Computação Ubíqua deveria popularizar-se entre 2005 e 2020, após um período de transição dominado pela Internet e Computação Distribuída.

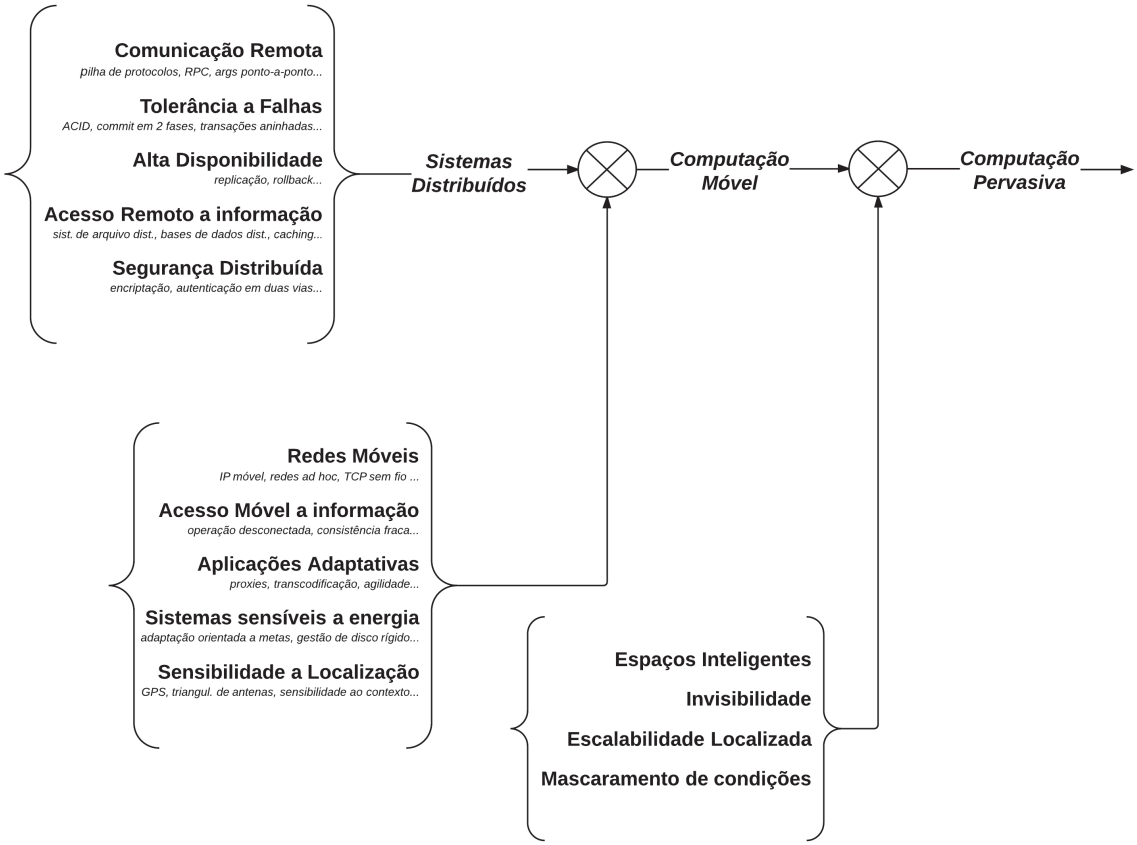
A Computação Ubíqua está baseada em dois aspectos principais: a ubiquidade e a transparência (WEISER et al., 1998). Por Ubiquidade pode-se entender que o sistema não depende de um terminal específico, mas está disponível em qualquer lugar, quando se necessite. Tanto o usuário quando o dispositivo podem ser móveis. Por transparência é possível entender que a interação com os computadores deve ocorrer na periferia da atenção do usuário e apenas requisitando a atenção total do mesmo em caso de urgência. Essa definição é também sustentada por SALBER, DEY e ABOWD (1998). Coen et al. (1999), por sua vez, descreve a Computação Ubíqua em termos de ambientes inteligentes, com computadores integrados nos objetos do dia a dia, enriquecidos com sensores capazes de obter informações sobre seu contexto. De fato, Weiser já havia alertado para o fato de que a Computação ubíqua, ao embutir computadores nos objetos do dia a dia como paredes, carros, semáforos, eletrodomésticos, não precisaria de mini clientes (*thin clients*), mas sim mini servidores (*thin servers*).

O impacto social deste paradigma pode ser comparado a outros exemplos de tecnologias que foram incorporadas ao nosso dia a dia que, apesar de essenciais, não mais nos damos conta de sua presença - são transparentes. Como exemplos o autor nos dá a escrita, massivamente disseminada, e a eletricidade, que só é lembrada na sua ausência. Tanto a eletricidade quanto a escrita não são naturais mas tecnologias desenvolvidas pelo homem e incorporadas à realidade, com as quais interagimos de forma ubíqua (em todos os lugares) e transparente (sem nos

darmos conta).

Satyanarayanan (2001) apresenta a Computação Ubíqua (ou pervasiva para o autor) de uma maneira mais sistemática, inserindo a Computação Ubíqua como um ramo precedido pela Computação distribuída e a Computação Móvel (Figura 1). Para o autor, a Computação Ubíqua

Figura 1: Taxonomia de problemas de pesquisa para sistemas computacionais na computação ubíqua



Fonte: Satyanarayanan (2001)

herda muitos conceitos dos seus antecessores e apresenta alguns novos:

- Espaços inteligentes (*Smart Spaces*): um espaço inteligente é uma área bem definida (uma sala, um corredor, um auditório) que contém infraestrutura computacional embutida para, por exemplo, automaticamente ajustar algum parâmetro de acordo com a presença ou preferência dos usuários presentes (temperatura, linguagem de comunicação);

- Invisibilidade: A invisibilidade preconizada por Weiser na prática significa "distração mínima do usuário"onde o ambiente computacional se adapta às expectativas do usuário, oferecendo pouca surpresa, antecipando problemas, interagindo com o usuário de maneira quase subconsciente.

- Escalabilidade localizada: conforme o número de usuários e interações aumentam, assim como a sofisticação dos aplicativos e dos espaços inteligentes, é necessário que a arquitetura dos sistemas para Computação Ubíqua sejam escaláveis em dois sentidos. Por um lado, os sistemas deve poder absorver mais usuários e interações, por outro estas devem ser locais, ou seja, devem ser relevantes para aquele contexto computacional, a fim de evitar sobrecarregar o sistema com interações distantes, com pouca relevância. Segundo o autor: "Assim como as leis naturais do inverso do quadrado da distância, um bom *design* de sistema deve ser escalável através da redução das interações entre entidades. Isso contradiz diretamente o pensamento corrente sobre a Internet, a qual muitos acreditam anuncia a 'morte da distância'. "(SATYANARAYANAN, 2001)

- Mascaramento de condições desiguais: é a capacidade de um sistema de se adaptar a uma taxa de penetração de tecnologia ubíqua variável. Essa taxa de penetração da tecnologia em uma infraestrutura pode ser entendida como o nível ou complexidade de serviços que um espaço ou sistema inteligente pode oferecer. Mas não apenas isso, mas principalmente um sistema ubíquo deve ser capaz de se adaptar ao nível de serviço oferecido, naquele ambiente, dado que um sistema móvel pode interagir com diversos outros ao longo de um dado espaço de tempo. Ao se adaptar ao nível de interoperabilidade oferecido esse sistema ubíquo diminui a necessidade de interações desnecessárias com o usuário. Por fim, um dispositivo móvel pode interagir com ambientes mais ricos ou mais pobres de recursos e isso deve ficar transparente (o máximo possível) para o usuário.

- Interoperação espontânea: é a capacidade de um dispositivo em se comunicar com outros dispositivos em um ambiente dinâmico, ou seja, em um ambiente onde os dispositivos participantes mantêm associações passageiras. Isso caracteriza o princípio da volatilidade, onde dado que a interação entre os dispositivos é dinâmica e imprevisível, deve haver um tipo de regra que dirija a execução do sistema.

Além dos textos clássicos de Weiser e Satyanarayanan, os autores Costa, Yamin e Geyer (2008) elaboraram uma análise dos trabalhos mais recentes na área da Computação Ubíqua, a fim de identificar os atributos básicos em uma Infraestrutura de software para aplicações ubíquas. A Tabela 1 resume os resultados dessa análise, complementando o estudo dos problemas e das propriedades da Computação Ubíqua feito por Satyanarayanan (2001). Dentre os novos problemas de pesquisa abordados por Costa, Yamin e Geyer (2008), deve-se citar:

- Heterogeneidade: atributo derivado da Computação distribuída; Os programas na Computação Ubíqua devem abstrair para os usuários as diferenças na infraestrutura e gerenciar as conversões necessárias de um ambiente para o outro. Neste cenário, os desenvolvedores, independentemente da plataforma, devem criar a lógica apenas uma vez;

- Tolerância a falhas e segurança: esse conceito engloba atributos como disponibilidade, confiabilidade, segurança, integridade e capacidade de manutenção. Idealmente, o que

se deseja é evitar as falhas mais severas e frequentes; Segurança está ligada a Tolerância a falhas. Um sistema é seguro se existem medidas que garantam a disponibilidade, integridade e disponibilidade. É possível utilizar muitos mecanismos de segurança da Computação distribuída, mas estes precisam ser "leves"a fim de preservar a espontaneidade de interações e a limitação de determinados dispositivos;

- Privacidade e confiança: Privacidade está relacionada às regras de uso da informação - como ela será acessada e distribuída. Para os autores essa é uma questão importante e um desafio para os sistemas na Computação Ubíqua;

- Mobilidade: é a capacidade que os sistemas de Computação Ubíqua devem ter para prover acesso às aplicações e dados onde quer que o usuário esteja. Essa mobilidade pode ser física ou lógica. As aplicações e dados devem estar disponíveis em diferentes dispositivos (aplicações "Siga-me");

- Sensibilidade ao contexto: esse é um atributo herdado da Computação Móvel e representa a capacidade de um sistema em inferir o contexto do usuário e/ou da aplicação a fim de fornecer informações para que o mesmo adapte-se a nova situação;

- Gestão do contexto: Está ligado à Sensibilidade ao Contexto; é a ação em resposta a dados de sensoriamento do contexto, com o objetivo de adaptar o sistema. Além disso, deve ajudar o sistema a adaptar-se a maior ou menor disponibilidade de recursos de um ambiente.

Por fim Weiser et al. (1998) alertam para um efeito colateral da ubiquidade: a sobrecarga de informação sobre os usuários. Para os autores, faz-se necessário que sistemas baseados em Computação Ubíqua informem sem sobrecarregar. TVs, rádios, telefones e mesmo a Internet, em excesso, podem ser fontes de distração e irritação, o que contribuem para deteriorar a qualidade de vida. Já que os computadores estarão em toda a parte, informando, a Computação Ubíqua deve seguir o exemplo de outras tecnologias que atuam na periferia da atenção do usuário. Esse conjunto de orientações ficou conhecido como Tecnologia Calma.

2.1.1 Sensibilidade ao Contexto

Desde os primórdios da computação, o contexto dos sistemas computacionais era definido pelo local onde os computadores estavam instalados. Mesmo os PC's eram usados apenas em escritórios, laboratórios ou em chão de fábrica. Esse contexto era estático, com pouca variação das situações que circundavam os computadores - não havia a necessidade de adaptação a diferentes ambientes. Isso mudou radicalmente com a chegada e popularização da Computação Móvel, em que os usuários levam os computadores consigo, utilizando-os em diversas situações e ambientes.

Tabela 1: Problemas e desafios da Computação Ubíqua

Problemas	Área de Interesse	Motivação
Heterogeneidade	Sist.Distribuídos	Suportar uma variedade de serviços, dispositivos, redes, sistemas e ambientes.
Escalabilidade	Sist.Distribuídos	Permitir deployments em larga escala; Aumentar o número de recursos e usuários.
Tolerância a Falhas e Segurança	Missão crítica / Sist.Distribuídos	Evitar falhas mais graves e frequentes que o aceitável; Oferecer disponibilidade, confidencialidade, confiabilidade, segurança, integridade e capacidade de manutenção.
Privacidade e Confiança	Internet e Computação Móvel	Proteger contra mau uso de dados pessoais; Definir os níveis de confiança de um sistema.
Interoperação Espontânea	Computação Móvel	Suportar componentes que podem mudar identidade e funcionalidade;Permitir associação e interação.
Mobilidade	Computação Móvel	Suportar aplicações e acesso a dados em qualquer lugar e hora; Permitir que o ambiente acompanhe o usuário.
Sensibilidade ao Contexto	Computação Móvel	Perceber o estado do usuário e arredores; Inferir informações de contexto.
Gerenciamento do Contexto	Computação Móvel	Modificar o comportamento do sistema com base no contexto; Adaptar-se à situação corrente.
Interação Transparente com usuário	Computação Ubíqua	Mesclar a interface do usuário com o mundo real; Permitir ao usuário focar nas tarefas com um mínimo de distração.
Invisibilidade	Computação Ubíqua	Permitir ao usuário focar nas tarefas, não nas ferramentas; Fazer com que os computadores desapareçam da consciência.

Fonte: Costa, Yamin e Geyer (2008)

No início dos anos 90, os pesquisadores causaram um grande impacto ao expandir a noção de Computação Móvel, como até aquele momento era entendida, lançando as bases de uma nova área de pesquisa: a Computação Ubíqua ou Ubicomp (WEISER, 1991) .

Naquela época, a principal preocupação era tornar a Computação Móvel transparente, onde o usuário tivesse automaticamente acesso aos mesmos serviços onde quer que estivesse. As pesquisas iam em direção da melhoria da conectividade, a fim de prover os serviços de maneira automática e eficiente, sem interferência do usuário. É notável que a análise do contexto não era ainda vista como relevante. Ao contrário, o dispositivo móvel deveria ter o mesmo comportamento e serviços independentemente do ambiente onde estivessem.

Contudo, SCHILIT, ADAMS e WANT (1994) publicaram o que veio a ser o primeiro estudo sobre sensibilidade ao contexto (DEY; ABOWD; SALBER, 2001). Neste trabalho, o foco muda e a computação sensível ao contexto deveria produzir sistemas que se adaptassem à localização do usuário e à presença de outras entidades (pessoas, sistemas, servidores, dispositivos móveis). A computação não mais seria indiferente ao ambiente mas deveria ser capaz de examinar, reagir

e se adaptar ao mesmo. Dessa forma, a noção de sensibilidade ao contexto passou a ser central para os futuros trabalhos relacionados com a Computação Ubíqua, pois ela permite rápida adaptabilidade dos sistemas, pouca atenção ou intervenção do usuário e grande usabilidade, encaixando-se na ideia de desaparecimento ou transparência da tecnologia de Weiser.

Na definição clássica, contexto é qualquer informação que possa ser usada para caracterizar a situação de uma entidade (pessoas, lugares, objetos) e que possa ser relevante para a interação entre um usuário e uma aplicação, incluindo o usuário e a própria aplicação. Um modelo de contexto, por sua vez, representa uma parte do contexto que pode ser realisticamente obtido de sensores, da aplicação e dos usuários, e que possa ser explorado na execução de uma tarefa (HENRICKSEN, 2003). Essas informações podem ser discretas ou contínuas, representando, tipicamente, a localização, a identidade e o estado de pessoas, grupos e objetos físicos e computacionais.

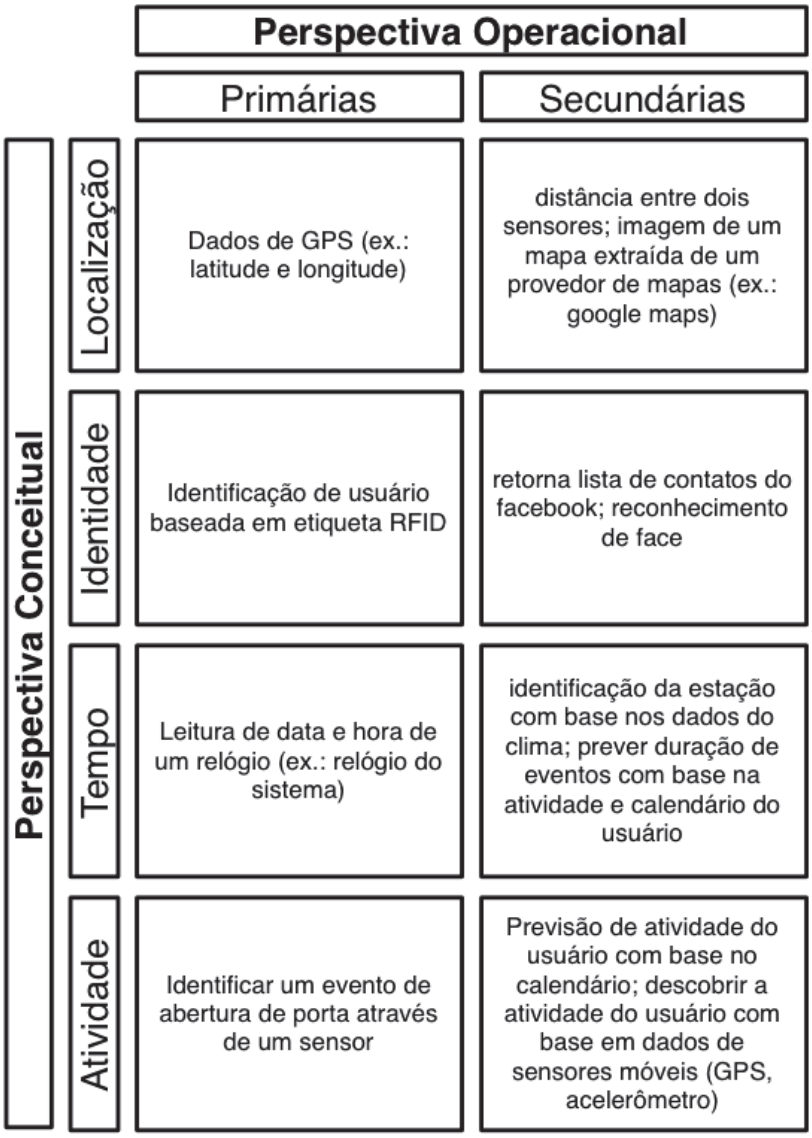
Dey, Abowd e Salber (2001) propuseram 4 categorias básicas de informação, a fim de descrever o contexto:

- Identidade: capacidade de atribuir um único identificador a uma entidade;
- Localização: toda informação que pode ser usada para deduzir a relação espacial entre duas entidades; não são apenas as informações de posicionamento num espaço bidimensional, mas co-locação, proximidade, contenção, orientação;
- Tempo: informação de contexto que auxilia a caracterização de uma situação; usado em conjunto com outros atributos, o tempo pode usado para avaliar a validade de uma característica, ordenar eventos, analisar causas e efeito;
- Atividade: identifica uma característica intrínseca de uma entidade, que pode ser avaliada (status) ou medida;

Perera et al. (2014), na Figura 2, ampliam as categorias básicas de contexto apresentadas por Dey, Abowd e Salber (2001), introduzindo uma dimensão operacional aos dados de contexto, de acordo com o seu nível de abstração. Uma informação pode ser classificada conceitualmente como Localização e operacionalmente pode ser uma informação secundária, ou seja, derivada ou calculada a partir dos dados brutos de sensores. De fato, as categorias básicas de contexto contêm pedaços simples de informação que combinadas e analisadas devem levar a um entendimento mais eficiente e completo da situação onde a entidade se encontra. Por exemplo, uma série histórica contendo localização (e tempo) permite obter a velocidade média de uma entidade em dado período, inferir sobre o comportamento (que pode denotar um status) da entidade ou ainda prever a posição de uma entidade em um momento futuro. Enfim, aplicações sensíveis ao contexto devem estar preparadas para receber (ou responder) as seguintes perguntas sobre uma entidade: quem? onde? quando? como? o quê?

Dessa forma, um sistema é sensível ao contexto quando utiliza o mesmo para prover o usuário com informações ou serviços relevantes, onde esta relevância depende da tarefa do usuário

Figura 2: Categorias de Contexto



Fonte: Perera et al. (2014)

(DEY; ABOWD; SALBER, 2001). Esse sistema deve suportar a utilização de dados de contexto para decidir as informações a serem apresentadas ao usuário (*Presentation*), execução automática de serviços (*Execution*) e classificação dos dados de contexto (*Tagging*). Para Barkhuus et al. (2003), a sensibilidade ao contexto apresenta três níveis, com base na interação com usuário:

- Personalização: permite ao usuário setar preferências (ex.: utiliza a aplicação para modificar a temperatura de um ambiente climatizado);
- Sensibilidade ao contexto passiva: monitora o contexto e notifica ou recomenda ao usuário algum produto ou serviço baseado em localização e preferências;
- Sensibilidade ao contexto ativa: monitora o contexto e inicializa automaticamente algum

serviço quando determinada condição ou cenário é identificada.

A Figura 3 apresenta um diagrama com um resumo do Ciclo de Vida do Contexto (PERERA et al., 2014). O ciclo de vida do contexto consiste em quatro fases . Na primeira, o contexto pode ser adquirido de diversas fontes, como, por exemplo, sensores físicos ou virtuais (aquisição de dados). Na segunda fase, os dados coletados precisam ter sido modelados e representados de forma adequada (modelagem). Na terceira fase, os dados precisam ser processados de forma que permita a extração de informações de contexto de alto nível, a partir de dados brutos de sensores e outras fontes (modelos de decisão). Por fim, os dados de alto e baixo níveis precisam ser disponibilizados aos consumidores interessados no contexto (distribuição).

Dey, Abowd e Salber (2001) descrevem ainda os requisitos principais para um *framework* analisador de contexto:

- Separação de conceitos: pode ser entendido como um padrão ou princípio de design que orienta a computação da informação de maneira modularizada. No âmbito do trabalho dos autores, a separação de conceitos significa separar a aquisição do contexto do seu uso na aplicação. Um exemplo simples dado pelo autor é de que um *framework* precisa saber como consumir um dado de contexto proveniente de um sensor, mas não precisa saber como este funciona - isto deve estar transparente;

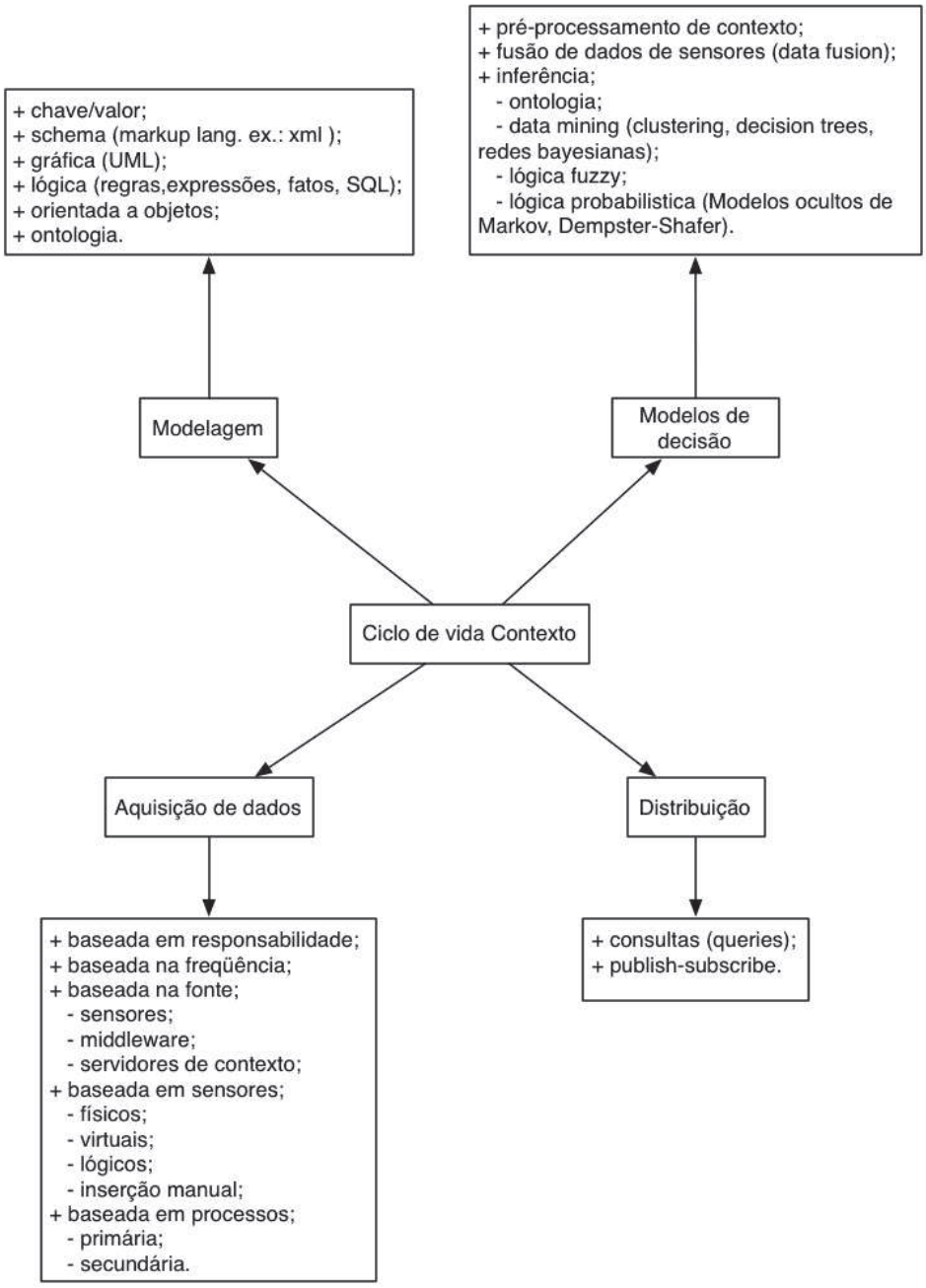
- Interpretação de contexto: o *framework* deve, em certa medida, oferecer interpretação para as informações de contexto, antes destas estarem disponíveis para a aplicação. De fato, o dado bruto sobre o contexto pode precisar passar por diversas camadas de abstração e análise antes de se tornarem úteis a uma aplicação. As informações básicas de contexto devem ser organizadas em conceitos mais complexos e tratados diretamente pelo *framework*, principalmente aqueles dados que podem ser reusados por diversas aplicações;

- Comunicação transparente e distribuída: a comunicação entre aplicação e sensores, que é de natureza distribuída, precisa ser transparente tanto para os sensores quanto para a aplicação; para o autor, um sistema de sincronização de *clock* distribuído (*global clock*) se faz necessário;

- Disponibilidade constante de dados de contexto: os sistemas baseados em Computação Ubíqua são naturalmente sistemas distribuídos. Ainda que a comunicação entre as partes de um sistema possa ser assíncronas, a disponibilidade dessas partes é essencial para o sucesso de uma tarefa de uma aplicação. Por exemplo, diversas aplicações podem estar acessando o mesmo sensor e algumas podem estar sincronizando informações com o mesmo servidor. As partes isoladas funcionam, mas perdem seu valor e sentido;

- Descoberta de recursos: esta característica é importante para permitir que o *framework* busque por algum sensor ou sistema para satisfazer a requisição da aplicação, de uma forma transparente e automática;

Figura 3: Ciclo de vida do Contexto



Fonte: elaborado pelo autor

- Armazenamento e histórico de contexto(trilhas): são registros históricos de uma entidade, em um contexto ou diversos contextos, em ordem cronológica. As trilhas permitem a ampliação da capacidade de um sistema de se adaptar a um usuário (SILVA et al., 2009), pois permitem estender o conceito de contexto para além dos dados em tempo real (reações do usuário, reações de outros agentes, sensores) para os dados históricos armazenados, de qualquer natureza. De fato, o *framework* deve estar preparado para armazenar e fornecer esses dados sempre que solicitado, no nível máximo possível de granularidade.

Um *framework* analisador de contexto, aderente aos requisitos apresentados por Costa, Yamin e Geyer (2008) e por Dey, Abowd e Salber (2001), pode ser útil para a implementação de sistemas ubíquos nas mais diversas áreas como ensino, saúde, gestão, negociação e comércio.

2.2 Comércio Ubíquo

A convergência entre a Computação Móvel eficiente (alta disponibilidade de redes sem fio e dispositivos móveis mais sofisticados) e o comércio eletrônico possibilitou o aparecimento de um novo paradigma na computação: o comércio ubíquo ou *u-commerce* (ou ainda *pervasive commerce* ou *p-commerce*) (ROUSSOS et al., 2003). Para Lin, Yu e Shih (2005) o *u-commerce* pode ser considerado uma extensão do *m-commerce* (comércio móvel) mas com sensoriamento do ambiente. Podemos estender essa ideia e dizer que *u-commerce* é *m-commerce*, mas com análise de contexto e, por que não, também análise de histórico de contexto (trilhas).

De fato, o *u-commerce* lida com novos tipos de problemas. Enquanto no *m-commerce* a preocupação no desenvolvimento de aplicativos era permitir uma experiência de acesso a produtos e serviços similar aos desktops, através do modelo cliente-servidor em dispositivos com conectividade e desempenho pobres, um dos principais desafios do comércio ubíquo é oferecer análise e integração das informações e disponibilizá-las de maneira inteligente e não perturbadora (LIN; YU; SHIH, 2005):

"(...) o problema para o comércio ubíquo não será a falta de poder computacional mas como a informação coletada será integrada e analisada. O ambiente inteligente poderá sobrecarregar os usuários com muita informação. Um bom sistema de comércio ubíquo deverá ajudar os usuários a processar a informação inteligentemente". (LIN; YU; SHIH, 2005)

Em resumo, o *e-commerce* refere-se ao uso da Internet para a comunicação e suporte a transações entre as organizações e stackholders (ZHANG et al., 2009), enquanto o *m-commerce* estende o mesmo através de novos canais comunicação sem fio e possui as características de portabilidade, alcançabilidade, acessibilidade, localização e identificação (Zhang et al. (2009) e Sheng, NAH e Siau (2008), ver Tabela 2). Desta forma, o *u-commerce* pode ser descrito como uma extensão lógica do comércio eletrônico (*e-commerce*) e do comércio móvel (*m-commerce*) (JUNGLAS; WATSON, 2003).

Para GALANXHI-JANAQI e NAH (2004), o comércio ubíquo é um novo paradigma que combina redes sem fio, televisão, voz e *silent commerce* com o atual comércio eletrônico. Os autores ainda analisam as características básicas desejáveis da Computação Ubíqua (estendidas ao comércio ubíquo), enunciadas por Junglas e Watson (2003), sendo:

- Ubiquidade: computadores estarão em toda a parte, com acesso a rede em qualquer lugar a qualquer momento; As pessoas vão interagir com os objetos do dia a dia que carregarão computadores, pequenos e "inteligentes". Isto se tornará tão automático e comum que as pessoas pouco notarão os agentes computacionais. Por isso, estes se tornarão "invisíveis";

- Universalidade: um mesmo equipamento poderá ser utilizado em qualquer local do planeta, funcionando sempre da mesma forma;
- Personalização: identificação dos usuários não somente pela sua identidade e preferências, mas também por sua localização geográfica;
- Sincronização: dados integrados através de aplicações múltiplas, de modo que o usuário visualiza suas informações em qualquer dispositivo utilizado.

Em outras palavras, de acordo com Sheng, NAH e Siau (2008):

- Ubiquidade = Alcançabilidade + Acessibilidade + Portabilidade;
- Universalidade = Redes sem fio+ Dispositivos móveis;
- Personalização= Localização + Identificação + Portabilidade;
- Sincronização = Aplicações móveis+ Sincronização de Dados.

A partir disso, Watson descreve o comércio ubíquo como um fluxo direto e contínuo de comunicação, conteúdo e serviços entre negociantes, fornecedores, empregados, clientes e produtos. O comércio ubíquo deve suportar interações e transações a qualquer tempo e lugar (JUNGLAS; WATSON, 2003).

Por sua vez, GERSHMAN (2004) identifica três características essenciais para o suporte ao comércio ubíquo:

- Estar sempre conectado com o cliente;
- Estar ciente em tempo real do contexto do cliente (Onde os clientes estão? O que eles estão fazendo? O que há em torno deles?);
- Ser sempre proativo identificando, em tempo real, possíveis oportunidades para satisfazer necessidades do cliente.

Por fim, mas não menos importante para a definição de comércio ubíquo, está a sensibilidade ao contexto (figura 4). Uma das principais características que o comércio ubíquo herda da Computação Ubíqua é o uso e a produção de informações de contexto, geradas por diversas fontes, que se enquadram nas quatro categorias identificadas por Dey, Abowd e Salber (2001): identidade, tempo, localização e atividade.

Quanto a artigos sobre comércio ubíquo, Liu (2013) preparou uma revisão dos trabalhos publicados em periódicos e eventos da área, entre 2000 e 2010. O autor propõe uma classificação dos artigos baseada em cinco temas: teoria e pesquisa, rede, tecnologia, aplicação e camada do usuário. O texto faz referência a trabalhos clássicos da área, assim como outros mais recentes, apresentando-os de forma sistemática e contextualizada. Além disso, traz informações

Tabela 2: Comparação M-Commerce e U-Commerce

		M-Commerce	U-Commerce
Nível Conceitual	Características	Móvel Sem Fio	Ubiquidade Universalidade Personalização Sincronização
Nível Tecnológico	Aplicação	número limitado de funcionalidade	maior diversificação e disponibilidade de funcionalidades
	Comunicação	Cobertura limitada de redes sem fio e heterogeneidade nos padrões de rede	Múltiplas redes; alta disponibilidade; acesso de dados entre redes
	Dispositivo	celulares, PDA's	Combinação de vários tipos de dispositivos incluindo dispositivos não tradicionais
	Dados	Integração e sincronização de dados limitadas	Completa sincronização e integração de dados

Fonte: Sheng, NAH e Siau (2008)

Figura 4: Elementos do u-commerce

Fonte: Franco et al. (2011)

sobre diferentes bases de dados de pesquisa e periódicos, assim como uma série de eventos e conferências de destaque na área de computação móvel e ubíqua.

Dado que um dos principais desafios do comércio ubíquo é oferecer análise e integração das

informações e disponibilizá-las de maneira inteligente e não intrusiva, segundo Lin, Yu e Shih (2005), existe um grande potencial para pesquisa e implementação de sistemas de recomendação associados a ambientes ubíquos. Este é o tema da próxima seção.

2.3 Recomendação em Sistemas Ubíquos

Ainda que pareça difícil antecipar as necessidades de um indivíduo em qualquer hora ou lugar, os sistemas de recomendação ubíquos deverão ajudar as pessoas a lidar com a crescente produção de informação. Não apenas o comércio eletrônico, mas os sistemas de recomendação associados com Computação Ubíqua serão úteis para implementar aplicações como guia de turismo inteligentes, auxílio a navegação, escolha de conteúdo online, entre outros.

Segundo McDonald (2003), diversas são as estratégias de análise dos dados. Contudo, qualquer uma delas recai em uma das 3 categorias:

- Filtragem baseada em conteúdo: esses sistemas utilizam técnicas de aprendizagem de máquina como naive Bayes para analisar texto automaticamente. Por exemplo, esse sistema pode comparar as palavras em opiniões sobre filmes online com termos que caracterizem as preferências de filmes que o usuário geralmente escolhe. Dessa forma, o sistema é capaz de estimar se o usuário tem chances ou não de gostar do filme;

- Filtragem colaborativa: são sistemas que ignoram descrições e focam somente na avaliação de um item por múltiplos usuários. Por exemplo, se um filme que o usuário ainda não viu foi altamente recomendado por outros usuários que compartilham gostos semelhantes ao primeiro, esse filme pode ser recomendado ao usuário;

- Sistemas baseados em relações: esses sistemas procuram descobrir relações entre itens a fim de determinar quais são mais representativos. Um exemplo é o algoritmo do Google que utiliza critérios de associatividade entre páginas de Internet e as buscas mais utilizados para determinar a relevância de uma página.

Contudo, um dos principais desafios dos sistemas de recomendação em aplicações ubíquas é a modelagem de perfis de entidade em ambientes sensíveis ao contexto. Nessa condições, é necessário que um modelo de entidade possa ser dinamicamente elaborado e não estático, pois pode representar diversos aspectos ou atividades da entidade e não apenas informar se o usuário gosta ou não gosta de algum item. A análise do contexto pode necessitar de informações implícitas. Por exemplo, a filtragem colaborativa pode assumir que usuários que gostaram de determinado item no passado vão gostar de coisas similares no futuro. Contudo, a análise vai levar em conta apenas o dado bruto "gostou"ou "não gostou", sem analisar o conteúdo em si. Mas como analisar o conteúdo de item de categorias diferentes? Podemos cruzar informações de crítica a um filme com o perfil de um usuário mas ainda não é possível analisar o filme em si, ou comparar um filme com uma trilha sonora. Portanto, os sistemas de recomendação em

ambientes sensíveis ao contexto exigem que se encontre uma maneira de relacionar, comparar, medir, grandezas diferentes, informações provenientes dos mais diversos sensores em diferentes formatos de dados.

Por fim, o contexto pode influenciar decisivamente na análise de um dado. Por exemplo, qual o peso de uma linha reta? Dado que uma linha reta é um objeto matemático, abstrato, exposto desta forma a pergunta não faz sentido. Quando perguntamos qual o peso de uma reta mas contextualizamos dizendo que esta reta é um vetor, que deve possuir um peso e uma orientação, então a pergunta passa a ter sentido. Dado que a entidade, ou usuário, está se movimentando entre diferentes contextos, os sistemas de recomendação podem dar uma saída diferente para um mesmo usuário e objeto de recomendação, pois o contexto passa a ser decisivo na computação deste dado.

3 TRABALHOS RELACIONADOS

Neste capítulo, serão apresentados trabalhos na área de *m-commerce* e *u-commerce*. Estes trabalhos apresentam a maior parte dos seguintes elementos:

- modelo ou *framework* relacionado a *u-commerce* ou *m-commerce* ;
- apresentação e implementação de dados de contexto e seu histórico (trilhas);
- apresentação e implementação de análise de contexto (sensibilidade ao contexto);
- representação estruturada de dados (XML, ontologias);
- arquitetura de execução descentralizada;
- apresentação de técnicas de recomendação e negociação automáticas.

Os trabalhos foram pesquisados em algumas das principais bibliotecas online de artigos técnicos e acadêmicos: a IEEExplore *Digital Library*, a ACM *Digital Library* e a Editora Springer. Dentro da IEEExplore, foram feitas pesquisas principalmente nas publicações *Pervasive Computing* e *Mobile Computing*. As principais chaves de busca utilizadas foram, *pervasive*, *ubiquitous*, *commerce*, *business*, *agents*.

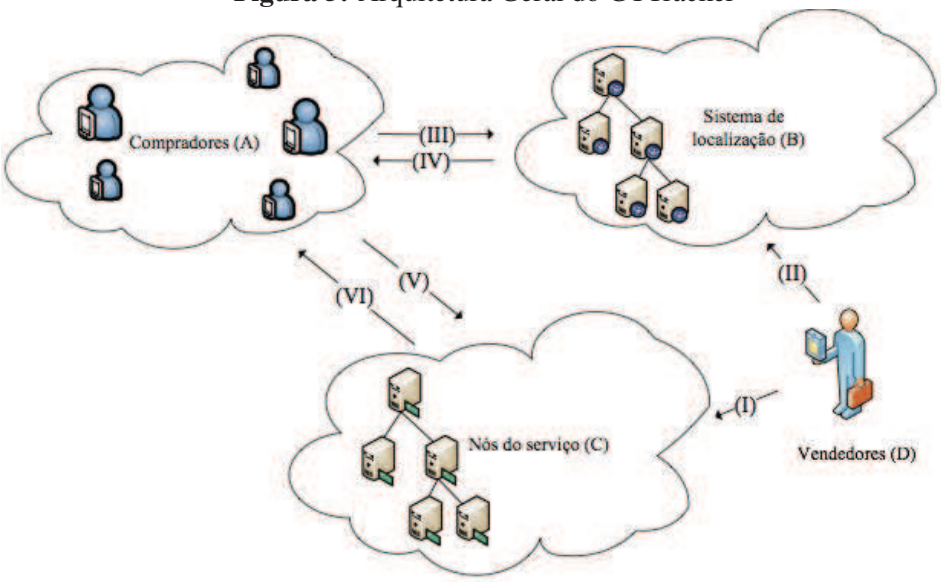
A seguir serão apresentados os principais aspectos de cada um dos trabalhos pesquisados e por fim uma tabela comparativa destacando os quesitos considerados mais relevantes para o estudo.

3.1 GTTracker

O GTTracker é um modelo de sistema para a implementação de um serviço compartilhado para publicação de ofertas e demandas comerciais. Este possibilita a identificação de oportunidades de negócio baseada na localização do usuário, que pode ou não ser móvel (CAZAROTTO, 2013). O modelo por sua vez pode ser utilizado como complemento a uma plataforma de comércio eletrônico mais abrangente. O GTTracker propõe uma execução distribuída das suas tarefas e utiliza infraestrutura de servidores em nuvem como *backend*. Na nuvem, esses servidores estão organizados hierarquicamente como Sistemas de Localização ou Nós de Serviço.

A interação entre os componentes do sistema pode ser observada na Figura 5. Uma massa de usuários do serviço (A) interagem com o sistema de localização (B) e posteriormente com a nuvem do serviço de identificação de oportunidades (C). Aos vendedores (D) atribui-se as primeiras tarefas, criando nós de comparação (I) e os registrando no sistema de localização (II), desta forma, os ofertantes podem publicar seus objetos de oferta. Após o processo inicial de publicação, os compradores (A) solicitam ao sistema de localização(III) o endereço do nó de

Figura 5: Arquitetura Geral do GTTracker



Fonte: Cazarotto (2013)

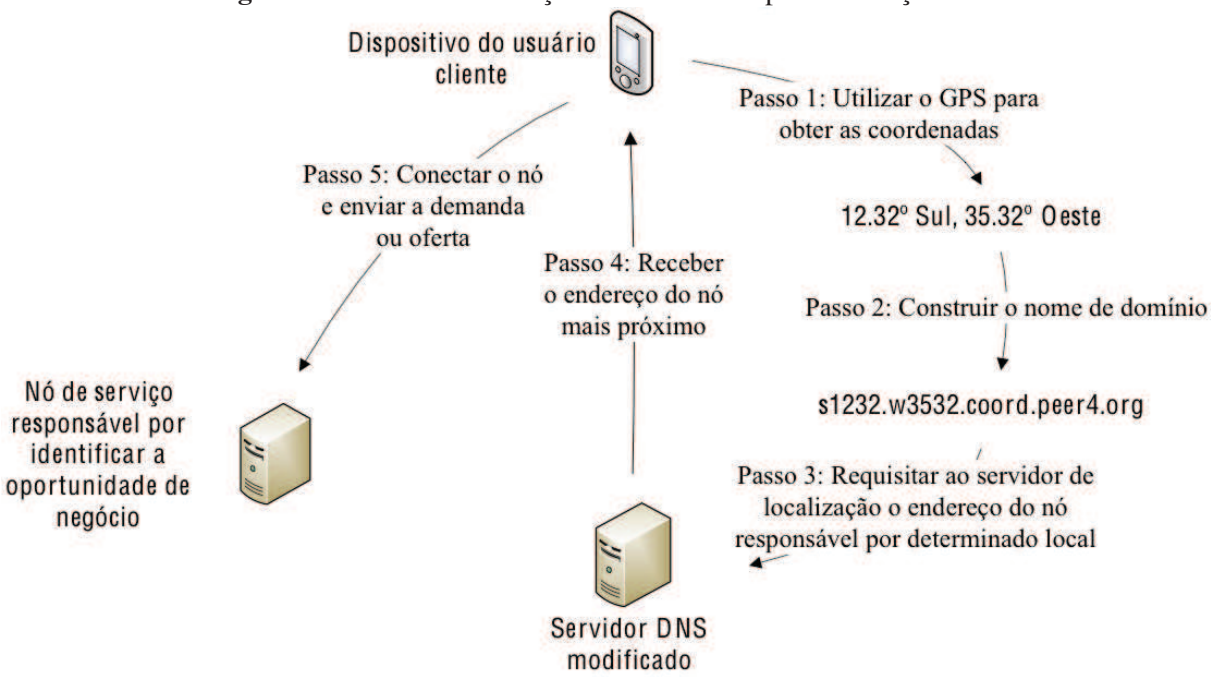
serviço correspondente a sua região, ao receber o endereço (IV) o comprador pode encaminhar objetos de demanda (V) e receber objetos de ofertas correspondentes a sua demanda (VI) (CAZAROTTO, 2013).

A Figura 6 mostra como um cliente, baseado na sua própria localização, utiliza uma estratégia de DNS modificada para construir o nome de domínio do Nodo de serviço responsável por uma determinada região.

Para o mapeamento da região, o sistema de localização determina polígonos nas áreas sobre autoridade de um servidor (ou mesmo grupo de servidores) e verifica em qual área está o cliente baseado nas suas coordenadas GPS. Através da utilização de algoritmo de cruzamento, verifica se o ponto apontado pelo GPS está em uma determinada área atendida. Se sim, devolve o endereço do nó de serviço.

A recomendação da oportunidade de negócio ou a publicação de uma oferta vão ocorrer uma vez que a aplicação do cliente passa a conhecer o nó de serviço. No caso da recomendação, o GTTracker utiliza um mecanismo de *matching* para relacionar as ofertas às demandas, onde os dados estão representados na forma de Meta Dados e não ontologias. Além disso, o modelo é flexível no que diz respeito ao mecanismo de *matching*. Na prática o modelo não apresenta um algoritmo de *matching* ou similaridade completo, mas uma arquitetura de *matching* baseada em *plugin*, um para cada tarefa do algoritmo de *matching*, deixando os detalhes de cada *plugin* fora do escopo. Dada a arquitetura de *matching* apresentada o modelo poderia ser implementado com *plugin* para execução de recomendação baseado em filtragem colaborativa, uma outra técnica de análise de similaridades ou ainda uma combinação de algoritmos para recomendação.

Figura 6: Processo de tradução de coordenada para endereço do nó



Fonte: Cazarotto (2013)

3.2 Modelo de Lin, Jiazao et al.

Lin et al. (2011) apresentaram um modelo de sistema de recomendação sensível ao contexto para aplicações de *m-commerce*. Os autores propõem um sistema de recomendação centralizado, baseado em agentes, conectado a bases de dados com histórico de movimentação comercial de clientes (por exemplo um grande atacado), que recomenda produtos ou serviços a esses clientes móveis. O modelo utiliza técnica de Filtragem Colaborativa para análise de similaridades e representa os dados de contexto através de um modelo de ontologia derivado de uma série de atributos de contexto. O modelo de ontologia foi batizado pelos autores como CUB-ONT e implementado no protótipo com a linguagem OWL. Os atributos do modelo de ontologia estão organizados em 4 categorias:

- perfil de usuário (estático) ;
- perfil do ambiente (dinâmico);
- histórico de contexto (trilhas - histórico de pesquisas, compras, recomendações);
- atividade corrente.

Quanto à arquitetura geral, a Figura 7 mostra o modelo constituído de 3 camadas principais:

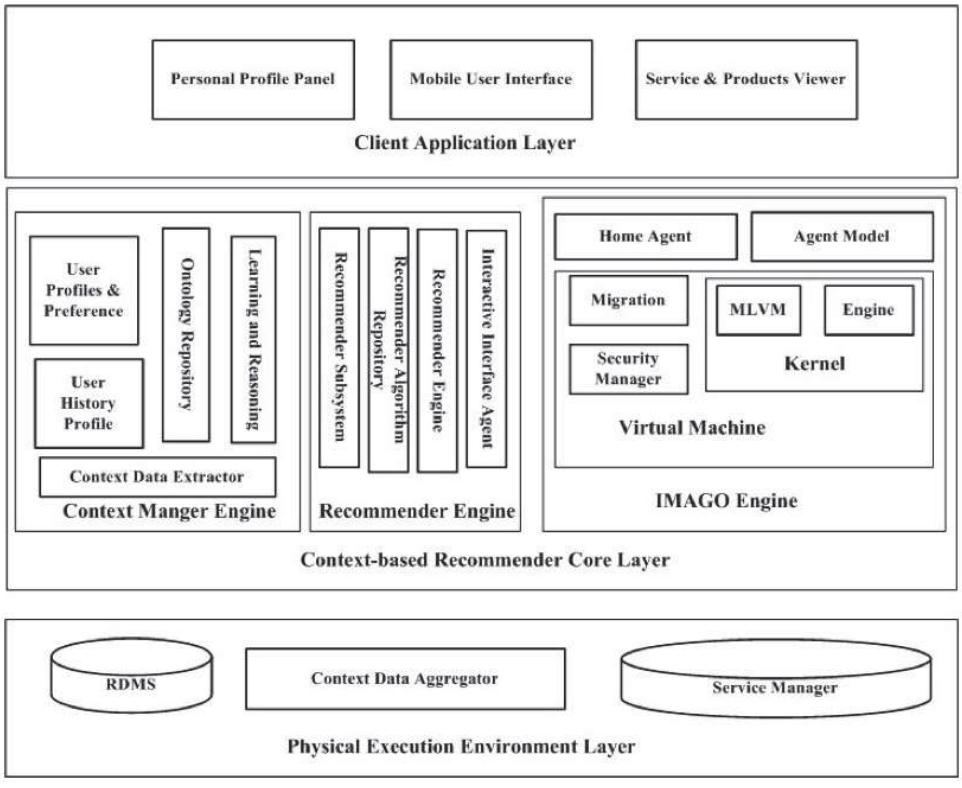
1. camada de aplicação do usuário: são as interfaces, os aplicativos dos usuários, instalados nos dispositivos móveis. O cliente é mínimo, contudo, pode fornecer informações de contexto como localização ou temperatura, por exemplo;

2. A camada intermediária, o Core ou Camada de Recomendação Sensível ao contexto é formada por 3 módulos principais:

   - motor de gerenciamento de contexto: esse módulo é responsável por organizar e analisar toda informação contextual, incluindo perfil de usuário e preferências, histórico de compras e atividades. Utiliza ontologia para representar os dados de contexto e perfil de usuário e Algoritmo de aprendizagem de máquina para avaliar mudanças no contexto;
   - motor de recomendação: utiliza dados estáticos e dinâmicos para gerar recomendações ao usuário. Esse motor pode utilizar uma implementação de técnicas de recomendação híbrida com Filtragem por colaboração ou Filtragem baseada em conteúdo;
   - motor dos Agentes: esse módulo tem a função de gerenciar a comunicação entre agentes, assim como coordenar o gerenciamento de memória, privacidade, segurança, mobilidade de código fonte.

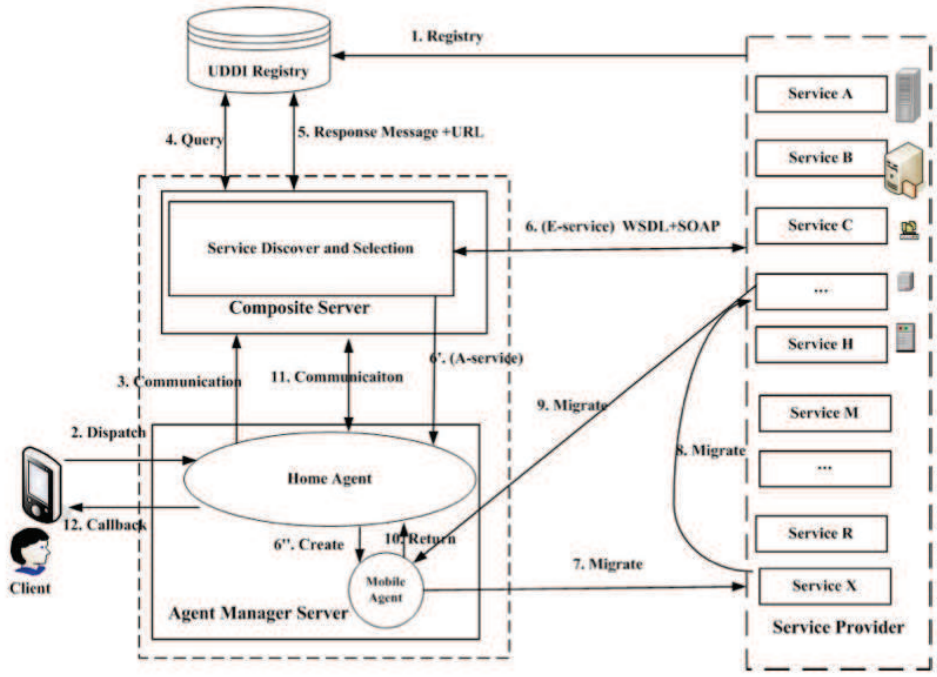
3. camada de Execução física: nessa camada se encontram os bancos de dados para as trilhas e os registradores UDDI para a descoberta de serviços.

Figura 7: Arquitetura geral do modelo de Lin et al.



Fonte: Lin et al. (2011)

Figura 8: Processo de descoberta e seleção de serviços



Fonte: Lin et al. (2011)

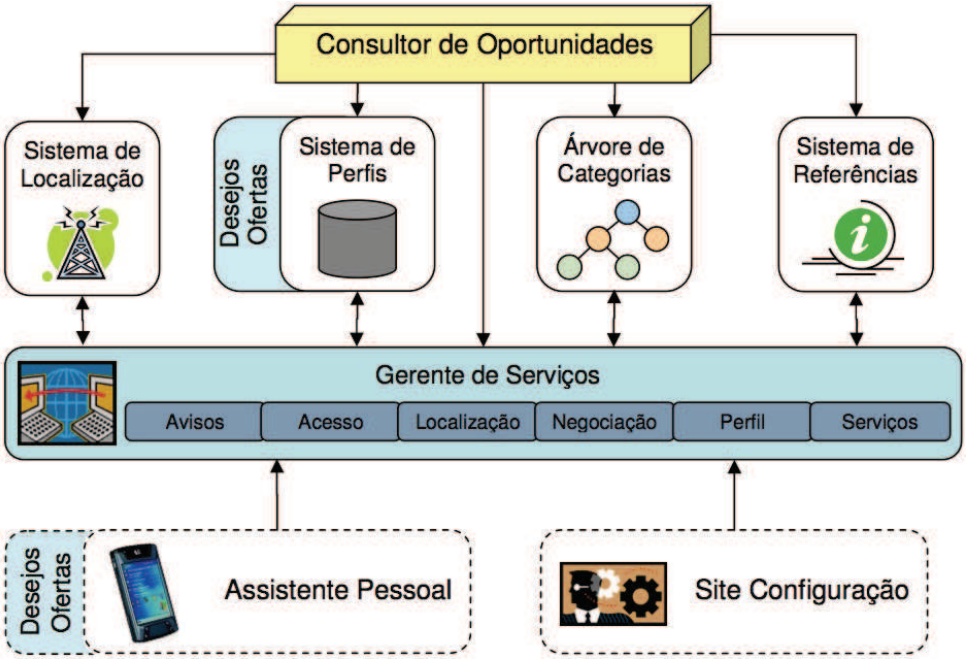
A Figura 8 mostra o fluxo típico de uma transação de *m-commerce*, utilizando descoberta e seleção de serviço. Após receber uma mensagem de um dispositivo móvel, o *Home Agent* traduz e envia a mensagem para o *composite server*. Este módulo extrai atividades atômicas da mensagem e utiliza regras de seleção definidas no servidor e informação de contexto para localizar serviços disponíveis no UDDI. Baseado nos resultados dessa busca o sistema invoca, para cada atividade, um *web service* se o resultado é do tipo E-service , ou cria um agente móvel se o resultado é do tipo A-service.

3.3 MUCS e TrailM

O MUCS é um modelo para suporte ao comércio ubíquo. O objetivo do modelo é suportar um cenário onde seja possível descobrir oportunidades de negócio (produtos ou serviços) baseado em informações de contexto (localização por exemplo), perfil de usuário, publicação de ofertas e demandas (FRANCO et al., 2011). Já o TrailM é um modelo genérico para gerenciamento de trilhas e foi concebido independentemente do MUCS.

A Figura 9 mostra a arquitetura geral do MUCS, composta por oito componentes. O primeiro é o Sistema de localização, que permite a identificação do contexto atual do usuário. O segundo é o Gerenciado de Perfil, que armazena os dados pessoais e as preferências dos usuários. O terceiro é a Árvore de Categorias, que mantém a taxonomia dos produtos e serviços. O quarto componente é o Sistema de Referência, que permite determinar a reputação de algum

Figura 9: Arquitetura Geral do MUCS



Fonte: Franco et al. (2011)

usuário. O quinto é o Gerenciador de serviços que opera como uma camada de troca de mensagens entre os componentes da interface e os outros componentes. O sexto componente é o Consultor de Oportunidades, o componente principal do modelo. O consultor de oportunidades é um motor de análise e recomendação, extraindo oportunidades de negócio dos perfis de usuários e do seu contexto (localização, preferências, histórico de transações). É no Consultor de Oportunidades que se encaixa o TrailM que gerencia especificamente as trilhas da entidade. Por fim, os dois últimos componentes são o Assistente pessoal (dispositivo móvel) e um *website*, para a administração do sistema.

O processo de identificação de uma oportunidade de negócio é orientado a "eventos"e está classificado em 3 categorias:

- Compra e Venda: quando ocorre o match entre ofertas e demandas;
- Troca de Conhecimento: quando ocorre um match entre os interesses de usuários;
- Troca de Experiência: quando ocorre um match entre os interesses de um usuário com negociações concretas executadas por outro usuário.

O TrailM foi agregado ao consultor de oportunidades do MUCS para permitir incluir não apenas dados do contexto do usuário mas o histórico de contexto (MARTINS et al., 2012). Esse módulo enriquece a recomendação de oportunidade na medida em que fornece dados de contexto em uma nova dimensão, a temporal. De fato o TrailM lida com trilhas de entidades e não propriamente usuários, o que permite a implementação de ubiquidade como observado por Sathyanarayanan(Sincronização). O TrailM possibilita não apenas o armazenamento e gestão

das trilhas pelas entidades mas também disponibiliza uma interface para consulta às trilhas, permitindo sua integração ao MUCS. Ele possui uma parte cliente e uma parte servidor e possui os seguintes serviços básicos (que podem ser estendidos):

- Gerenciador de trilhas que controla a interface com as aplicações (segurança, privacidade);
- Composição de Trilhas: o TrailM está baseado na ideia de composição de trilhas, onde as informações de contexto são recolhidas em pequenas porções pelo dispositivo móvel (ptrails) e depois combinadas e agrupadas em conjuntos maiores, em geral em um servidor na nuvem(trailpoint);
- Consulta de Trilha: disponibiliza serviços para consultar as trilhas.

3.4 Modelo de Sanchez-Pi e Molina

Sanchez-Pi e Molina (2009) apresentaram, em 2009, um sistema multiagente para suportar aplicações para *u-commerce* sensíveis ao contexto. O objetivo do trabalho consiste em adaptar um sistema multiagente (que suporta informações de sensores e preferências e perfil de usuário) aos requisitos de uma aplicação para *u-commerce*.

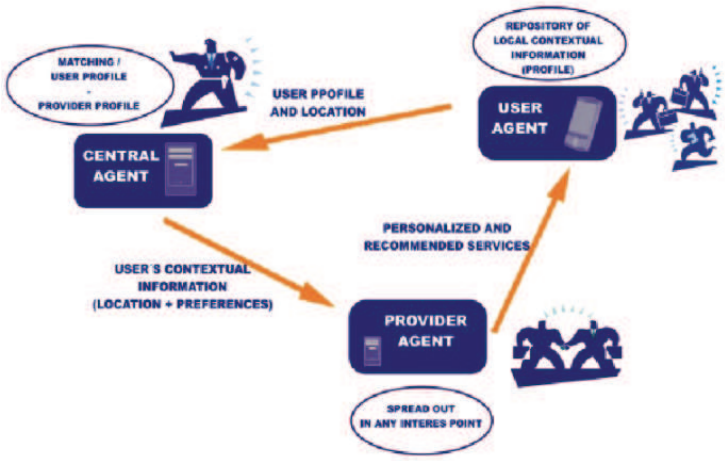
A Figura 10 mostra um modelo multiagente genérico para executar *matching* entre entidades. Neste sistema, um agente móvel se comunica com um agente central, que utiliza as informações de perfil e contexto do agente móvel para executar um *matching* entre o perfil agente do usuário e o agente do fornecedor de serviço ou produto.

A Figura 11 mostra um modelo semelhante ao da Figura 10, porém especializado para *u-commerce*. Neste caso, o principal objetivo é recomendar produtos e serviços a usuários móveis. As informações de perfil de usuário e contexto são analisadas por um agente centralizado (*Broker*) que é responsável pelo *matching* entre os compradores e vendedores. Os dados de perfil e contexto são representados através de ontologias específicas (uma estrutura para o vendedor e outra para o comprador.

O modelo é composto dos seguintes elementos:

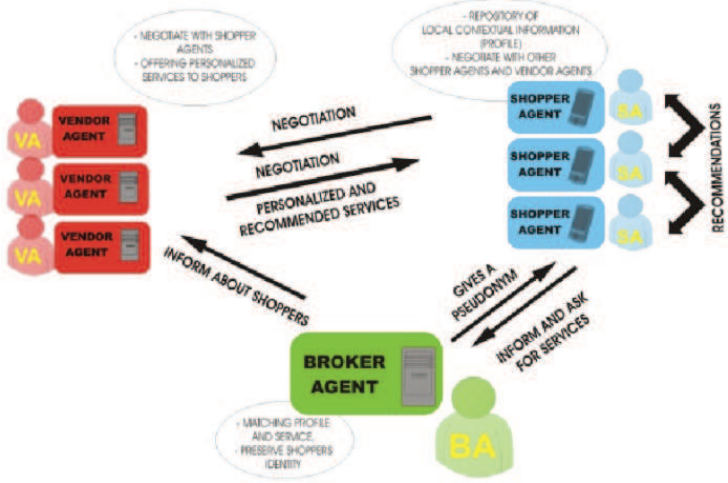
- Agente *Broker*: o *Broker* tem a função de agir como um *proxy* ativo. O cliente, através do dispositivo móvel se conecta ao *Broker* e este, a partir do seu perfil e intenção de compra do cliente, executa uma recomendação preliminar para descobrir qual o Agente Vendedor mais adequado, baseado em dados de contexto de ambos compradores e vendedores.
- Agente Consumidor(*Shopper Agent)*: o papel do agente do comprador ou consumidor é interagir com o agente vendedor, a fim de receber recomendação de serviços. Além disso, podem avaliar a qualidade do serviço e a qualidade da recomendação;

Figura 10: Arquitetura geral do Sistema Multiagente



Fonte: Sanchez-Pi e Molina (2009)

Figura 11: Arquitetura geral do Sistema Multiagente adaptado para u-commerce



Fonte: Sanchez-Pi e Molina (2009)

- Agente vendedor (*Vendor Agent*): este agente tem a tarefa de receber as notificações sobre a presença do cliente através do *Broker*e dar prosseguimento às tratativas de venda. Os agentes vendedores estão preparados para realizar um *matching* entre os interesses do comprador e as opções oferecidas pela loja;

Em um cenário de utilização do sistema, proposto pelos autores, uma cliente chega a um Shopping Center e o agente consumidor instalado em seu dispositivo móvel automaticamente conecta-se com o *Broker* do Shopping. A cliente informa alguns dados básicos, para a criação de um perfil inicial, já que é a primeira que visita o estabelecimento. Além do perfil, o *Broker* faz algumas perguntas adicionais sobre que tipo de produto o cliente procura. A cliente informa que está procurando vestidos e o *Broker* identifica um Agente vendedor. Ao ser notificado, o agente vendedor inicializa a negociação com o cliente, procurando mais detalhes sobre o que o cliente está interessado e qual o seu perfil. Com esses dados, o agente vendedor recomenda

ternos. A cliente, ao perceber o erro, avalia a qualidade do serviço. Neste caso, o cliente reporta o erro e avalia o sistema como pouco confiável. Baseado nisso, tanto o *Broker* quanto o Agente Vendedor podem reavaliar e refinar seus mecanismos de recomendação.

3.5 PAM

PAM (*Pervasive Active Manager*) foi apresentado como um modelo ou arquitetura genérica para implementação de sistemas de *u-commerce*. O modelo é centrado em um assistente pessoal, que auxilia o cliente na tomada de decisões no momento das compras. Lin, Yu e Shih (2005) propõem um modelo híbrido de execução, onde o assistente pessoal pode descobrir e consumir serviços alocados em servidores, sensores e mesmo outros dispositivos móveis, utilizando P2P.

A Figura 12 mostra o modelo dividido em quatro componentes principais:

- *Context Manager* (CM): Gerenciador de contexto dos usuários;
- *Object Manager* (OM): Gerenciador de produtos;
- *Profile Manager* (PM): Gerenciador de Perfis de usuários;
- *Service Manager* (SM): Gerenciador de recomendações do sistema.

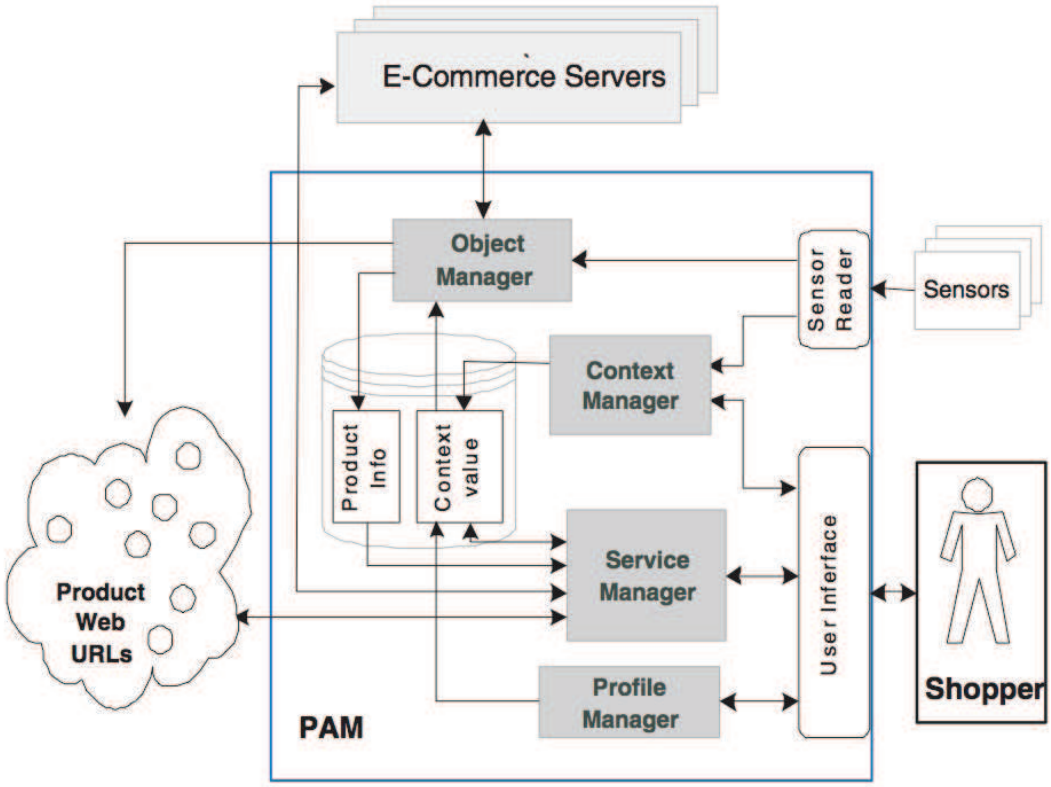
As tarefas básicas de um sistema PAM são:

1. coletar automaticamente todas as informações relacionadas aos produtos ou serviços identificados pelo usuário;
2. analisar eficientemente a informação coletada;
3. apresentar sistematicamente opções que auxiliem os usuários na tomada de decisão.

Abaixo, cenários de utilização do PAM, propostos pelos autores, utilizando uma arquitetura semelhante à mostrada na Figura 13:

- Locação de DVD: no primeiro cenário, um cliente vai a uma loja do shopping para alugar um DVD que pretende assistir com a família. O seu assistente pessoal identifica, através do leitor de RFID, que o cliente está segurando um filme a mais de 30 segundos. Neste instante, o assistente (PAM) procura outras informações sobre o filme nos serviços fornecido pela locadora, e conecta a Internet para buscar referências, exibindo as informações compiladas ao cliente (por exemplo, o resumo do filme, avaliações ou mesmo algumas cenas). Caso o cliente queira locar o filme, a operação pode ser realizada diretamente entre o assistente e o servidor da loja;

Figura 12: Arquitetura do PAM

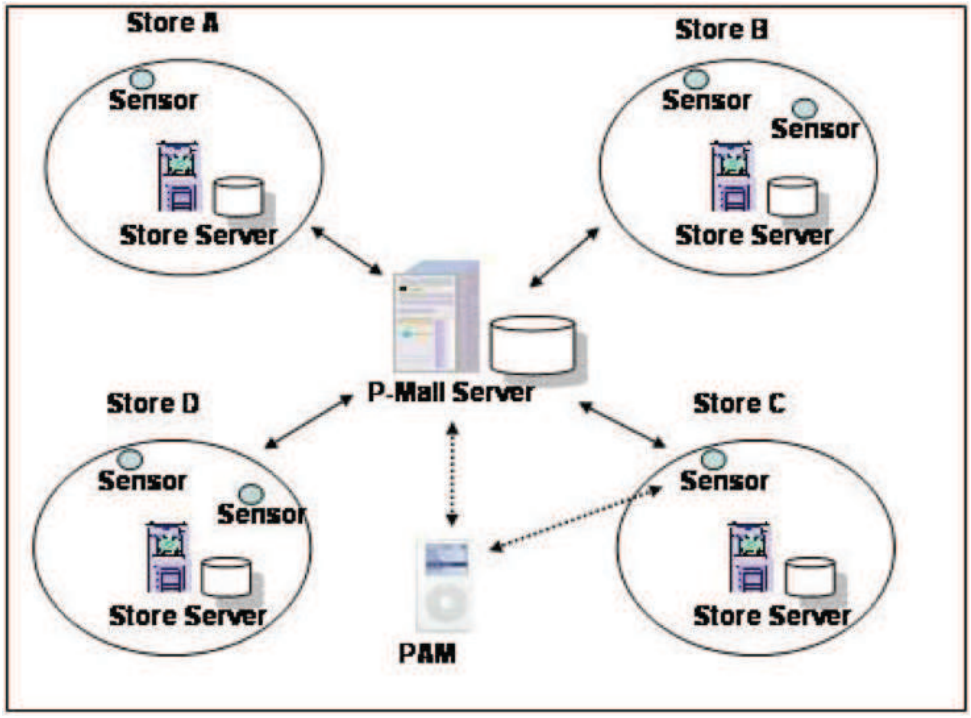


Fonte: Lin, Yu e Shih (2005)

- Reserva de restaurante: no segundo cenário, o PAM é utilizado para comunicação entre um casal que está em lojas diferentes dentro do mesmo shopping. Eles pretendem almoçar juntos, então o PAM verifica quais os restaurantes com reserva disponível, conforme perfil de preferência dos usuários. Realiza uma pré-reserva e sugere o restaurante e horário ao casal. Caso eles confirmem, o assistente irá efetivar a reserva e alertar os usuários quando estiver próximo do horário, considerando inclusive a localização atual e o tempo de deslocamento.

Lin, Yu e Shih (2005) não apresentam detalhes sobre representação dos dados de contexto e de perfil de usuário, apesar de apresentarem um componente específico para gestão de perfil. É interessante notar, nesse componente, a preocupação em modelar não apenas um gerenciador, mas um gerador de perfil de usuário, na medida em que esse módulo combina preferências pessoais inseridas no sistema como análise de histórico de contexto (trilhas). Além disso, o artigo não cita as técnicas ou algoritmos para recomendação ou mesmo para extração do perfil de usuário.

Figura 13: Cenário de uso do PAM (conectado a um servidor central em um Shopping Center)



Fonte: Lin, Yu e Shih (2005)

3.6 Comparação de características

A Tabela 3 exibe um comparativo entre os trabalhos apresentados neste capítulo, onde os seguintes atributos são levados em consideração:

- modelo de dados: identifica se o modelo de dados apresenta uma forma estruturada (metadados). Por modelo de dados entende-se a representação dos dados de contexto, dados do perfil de usuário e comunicação;
- utiliza Trilhas: identifica se o trabalho estudado utiliza histórico de contexto. As trilhas estão intimamente ligadas ao modelo de dados;
- perfil de entidade: Verifica se o modelo utiliza perfil de usuário. Em caso positivo, identifica se esse perfil é estático, com dados registrados pelos usuários ou administrador do sistema ou dinâmico se o perfil é atualizado de acordo com eventos externos, sem a interferência do usuário;
- modelo de sistema: procura descrever qual a arquitetura do sistema, dado que os trabalhos abordados baseiam-se em computação distribuída;
- privacidade e segurança: verifica se o trabalho abordado possui algum mecanismo não apenas para controle de acesso mas também ajuste para nível de compartilhamento de dados;

Tabela 3: Comparação dos Trabalhos relacionados

Atributos / Trabalhos	GTTracker	Jiazao Lin et al.	MUCS	Sanchez-Pi;Molina	PAM
Modelo de Dados	JSON	CUB-ONT (Ontologia)	Árvore de Categorias/Modelo Relacional/XML	Ontologia	Modelo Relacional
Utiliza Trilhas	Não	Sim	Sim	Não	Não
Perfil de Entidade	Estático	Estático	Estático	Estático	Dinâmico
Modelo de Sistema	P2P	Cliente-Servidor	Cliente-Servidor	Cliente-Servidor e P2P	Cliente-Servidor e P2P
Privacidade e Segurança	Controle Acesso	Controle Acesso	Controle Acesso	Controle Acesso	Controle Acesso

Fonte: Elaborado pelo autor

Quanto à sensibilidade ao contexto, todos os trabalhos examinados levam em conta dados de contexto no seu processamento. Esse contexto pode ser composto de localização, perfil do usuário ou mesmo sensores, como no casos do PAM. Contudo, a maioria trabalha com dados de contexto em tempo real, sem levar em conta seu histórico, exceto em Martins et al. (2012), que utiliza o MUCS (FRANCO et al., 2011) integrado a um modelo de apoio específico para o gerenciamento de contexto chamado TrailM, e em Lin et al. (2011) que armazena os dados de contexto descrito na sua ontologia chamada CUB-ONT e utiliza esses dados para a execução de análise de dados para recomendação.

Nos trabalhos analisados, o uso de perfil de entidade ficou restrito a um cadastro básico dos dados de usuário, em alguns casos enriquecido com preferências dos usuários. O Ubitrade (MARTINS et al., 2012) e o GTTracker procuram acrescentar ofertas e demandas ao perfil, a fim de executar o *matching* de preferências entre usuários.

A maioria dos trabalhos apresentam uma arquitetura distribuída predominante cliente/servidor, com clientes mínimos sendo executados em dispositivos móveis e servidores responsáveis pelo armazenamento de dados, processamento de recomendação, *matching* entre perfis de usuários ou gerenciamento da segurança. O GTTracker, que a princípio poderia ser identificado como cliente/servidor, na prática apresenta nodos computacionais funcionando como *peers* e *super-peers*. O nodo que corresponde ao cliente, de fato, executa funções mais complexas que um cliente mínimo, sendo responsável, por exemplo, pela coleta de dados de contexto (localização). No GTTracker outros nodos funcionam como *super-peers*, como o servidor responsável pela montagem do DNS dinâmico e o servidor que atenderá a requisição. Os trabalhos de Sanchez-Pi e Molina e PAM chamam a atenção por, apesar de uma parte do sistema implementar um modelo cliente servidor, utilizarem canais secundários de comunicação e interação com outros usuários ou sensores através de comunicação P2P. De fato, o atributo arquitetura

tem a função de caracterizar os sistemas do ponto de vista de interoperabilidade dos sistemas: são considerados centralizados aqueles sistemas onde as entidades precisam de servidores, *brokers* ou outros dispositivos nas rede para identificar, executar ou resolver o problema principal, nesse caso uma recomendação de oportunidade de negócios. São considerados descentralizados aqueles sistemas onde os *peers* ou entidades são autossuficientes e auto-organizadas a fim de publicar suas demandas e identificar suas oportunidades. Os modelos híbridos são aqueles que dependem de servidores ou *brokers*, mas possuem algum nível de interoperabilidade direta entre os dispositivos móveis.

Por recomendação entende-se alguma técnica ou algoritmo que procure identificar padrões ou preferências dos usuários, baseado em seu perfil ou histórico. Dessa forma, o GTTracker e o UbiTrade, que implementam *matching* de perfil de usuários (que inclui ofertas e demandas de produtos ou serviços), apresentam técnicas híbridas de recomendação, pois combinam filtragem colaborativa e filtragem por conteúdo.

Quanto à negociação automática, nenhum trabalho implementa esse recurso. De fato, todos utilizam algum serviço de *rating* ou referência de usuários, sendo que essas informações podem ser incluídas em análise de *matching* ou recomendação sem contudo executar negociação inter-agentes, seja de cooperação ou concorrência.

Os modelos estudados suportam as principais características da Computação Ubíqua, contudo não favorecem a escalabilidade local (SATYANARAYANAN, 2001) . Sistemas baseados no modelo cliente-servidor requerem a implantação de infraestruturas para operar os serviços no nível da aplicação. Existe portanto espaço para avaliação de estratégias descentralizadas para o comércio ubíquo, onde a aplicação está distribuída em nodos de rede de mesmo tipo.

A estratégia de utilizar recomendação para avaliação de oportunidades de comércio ou negócio é mais indicada para estruturas centralizadas, baseadas no modelo cliente-servidor, onde uma massa de dados de perfil, trilhas e movimentação de usuário pode ser combinada e analisada por um motor de recomendação. Em modelos descentralizados, não existe um repositório de dados global, onde estão todos os dados dos usuários, para a aplicação de técnicas de mineração e aprendizagem de máquina, utilizados no motores de recomendação. As entidades envolvidas na avaliação de uma oportunidade muitas vezes podem conhecer apenas os seus dados e ter limitado acesso a informações de outras entidades. Dessa forma, uma estratégia de negociação entre agentes surge como alternativa nesse cenário.

O próximo capítulo apresenta um modelo descentralizado para identificação de oportunidades de negócio, baseado em trilhas e negociação entre agentes.

4 MODELO U-DEAL

Neste capítulo, é apresentado o modelo U-Deal. A seção 4.1 apresenta uma visão geral do modelo descrevendo seus agentes e componentes auxiliares. A seção 4.2 descreve o diagrama geral do modelo e as demais seções mostram o detalhamento dos agentes, utilizando a metodologia Prometheus (PADGHAM; WINIKOFF, 2003).

4.1 Visão Geral

O U-Deal é baseado em dois outros modelos: a Global (OLIVEIRA, 2010) e o MUCS (FRANCO et al., 2011) . A Global é uma infraestrutura multiagente que oferece uma arquitetura descentralizada e extensível para aplicações sensíveis ao contexto e que utilizem perfis de usuários. Este trabalho foi criado para suportar o desenvolvimento de ambientes de aprendizagem ubíqua. Contudo, a Global apresenta um conjunto de características e serviços que podem ser aproveitados em outros modelos para Computação Ubíqua, como a utilização de um protocolo de comunicação entre agentes baseado em mensagens no padrão FIPA-ACL (FIPA-ACL, 2002) . Além disso, o modelo representa os dados de uma maneira estruturada, ainda que não faça uso de ontologias para representar contexto e perfil de usuários (aprendizes). O MUCS foi desenvolvido como um consultor de oportunidade de negócios, utilizando uma arquitetura cliente/servidor. O MUCS não utiliza ontologia nem trilhas, mas utiliza dados de localização e perfis de usuário para executar o *matching* entre ofertas e demandas. O consultor de oportunidade é executado em um servidor, que recomenda oportunidade a usuário com dispositivos móveis. Martins et al. (2012) fizeram o primeiro estudo sobre suporte a trilhas no MUCS, através da integração com o TrailM.

A integração das ideias trazidas por esses trabalhos permite a elaboração de um modelo multiagente, descentralizado, para identificação de oportunidades de negócio. Este novo modelo, o U-Deal, suportará trilhas e perfis de entidade, assim como deve permitir a identificação de oportunidades através da negociação entre agentes [1].

A identificação de uma oportunidade de negócio não é apenas um confronto e combinação entre as ofertas e demandas explícitas entre duas entidades, mas uma identificação mais geral das potenciais oportunidades baseadas nos desejos e experiências das mesmas (perfil, contexto, trilhas). Isso pode ser atingido se houver uma processo de negociação e troca de informações entre as entidades. Dessa forma, cada agente do modelo deve suportar uma ou mais das características descritas a seguir, a fim de dar subsídios ao agente de negociação executar sua tarefa.

De acordo com Lempereur, Sebenius e Duzert (2009), são 10 os elementos básicos en-

[1] Wooldridge e Jennings (1995) afirmam que um agente é uma entidade de software autônoma e extensível, sensível ao ambiente onde atua. Ampliando essa definição, Huhns e Stephens (1999) destacam que Sistemas Multiagentes são compostos por agentes que interagem e atuam em conjunto, sendo a interação baseada em protocolos de comunicação. Essas características estimulam o uso de agentes na modelagem de sistemas ubíquos

volvidos em uma negociação baseada na abordagem de ganhos mútuos (matriz de negociação complexa - Figura 14):

Figura 14: Matriz de Negociação Complexa

Fonte: Lempereur, Sebenius e Duzert (2009)

- Interesses: são preocupações, desejos, necessidades, receios, esperanças e temores motivadores das posições das partes. São valores subjacentes das posições e constituem-se nas razões pelas quais são estabelecidas as posições e as exigências. O problema básico de uma negociação não está nas posições conflitantes, mas sim no conflito entre as necessidades, desejos, interesses e temores de cada um dos lados;

- Alternativas: Alternativas são ações que podem ser realizadas por uma das partes independente dos interesses das outras partes;

- Opções: são possíveis acordos ou partes de acordos que podem, criativamente, satisfazer ambos os lados. São também maneiras e formas de se utilizar os diferentes interesses para criar valor;

- Legitimidade: refere-se à percepção de quão justo é o acordo ou a proposta realizada;

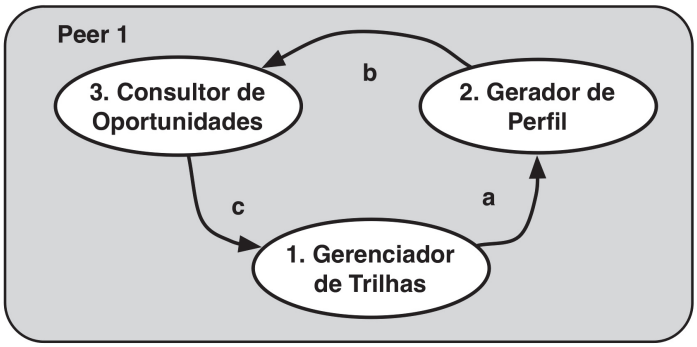
- Compromissos: são declarações sobre as intenções do que as partes pretendem fazer no futuro;

- Relacionamentos: corresponde ao padrão geral de como as partes se relacionam, dentro e fora da negociação;
- Comunicação: é constituída por mensagens e por meios pelos quais as partes trocam informações entre si. Deve ser clara e eficiente;
- Tempo: utilizado como uma variável estratégica, é o tempo necessário para o desenvolvimento das negociações;
- Conformidade: O elemento conformidade refere-se ao embasamento legal necessário à viabilização de um acordo;
- Contexto: qualquer informação que auxilie na análise situacional do negociador.

Desta forma, o objetivo deste trabalho é modelar e avaliar um sistema multiagente sensível ao contexto, com suporte a trilhas e que seja descentralizado. Neste caso, a principal contribuição do trabalho é apresentar um sistema onde cada usuário possua suas trilhas localmente e as informações de histórico de contexto e relacionamento entre os usuários estão distribuídas por diversas instâncias. A Figura 15 mostra uma visão geral de uma instância do modelo, focando o principal ciclo interno de informações. O primeiro agente a ser destacado é o Gerenciador de Trilhas(1), responsável pelas operações de leitura, escrita e organização das trilhas. O segundo agente é o Gerador de Perfil(2), cuja função é produzir periodicamente um perfil dinâmico de usuário, baseado nas informações básicas do utilizador e nas trilhas da entidade. O terceiro agente apresentado é o Consultor de Oportunidades(3), que utiliza o perfil dinâmico da entidade local e o perfil de outras instâncias do modelo para a avaliação de oportunidades de negócio. As setas da Figura 15 indicam a direção do fluxo de dados. Inicialmente, cada entidade deverá gerar o seu perfil dinamicamente, através da análise das trilhas e do perfil estático do utilizador (seta a). Esse perfil pode ser gerado periodicamente ou a partir de algum evento do sistema. Essas informações de perfil estarão disponíveis para o Consultor de Oportunidades(3) realizar, quando solicitado, uma comparação com o perfil de outras entidades a fim de determinar se alguma das instâncias da aplicação oferece oportunidades de negócio (seta b). Por fim, as ações e resultados do Consultor de Oportunidades(3) são armazenados nas trilhas de cada entidade, formando um banco de dados com o histórico das interações entre as diferentes instâncias(seta c). Esse histórico de interações e resultados, por sua vez, será utilizado nas futuras atualizações de perfil de cada entidade (seta a), recomeçando ciclo.

No que diz respeito à troca de informações entre instâncias do modelo, o agente de Comunicação possui um papel fundamental: não apenas tem a capacidade de identificar outras instâncias do U-Deal (que estejam disponíveis na rede via *wifi* ou *bluetooth*) e notificar outros agentes interessados, mas também suportar todas as trocas de mensagens entre os agentes de instâncias diferentes . Na Figura 16, assim que o agente de Comunicação identifica uma nova instância disponível na rede, uma requisição de troca de perfis é solicitada (a). Após a troca de perfis, o agente Consultor de Oportunidades é notificado e recebe o perfil da entidade

Figura 15: Visão geral de uma instância do U-Deal



Fonte: elaborado pelo autor

externa (b). De posse do perfil externo, Consultor de oportunidades requisita o seu próprio perfil atualizado ao agente Gerenciador de perfis (c e d) e procede com a comparação dos perfis a fim de responder às seguintes perguntas: entidades estão no mesmo contexto? perfis demonstram que existe potencial de negócios? A seta 'e' indica envio de notificações por parte do Consultor de Oportunidades aos agentes correspondentes em outras instâncias. Por exemplo, se uma oportunidade é identificada, esta notificação é passada ao agente de Comunicação que deverá fazer a entrega ao agente e instância corretos.

De um modo geral, é possível identificar 6 momentos em um processo de identificação de oportunidades. A Figura 17 mostra as etapas de 1 a 6:

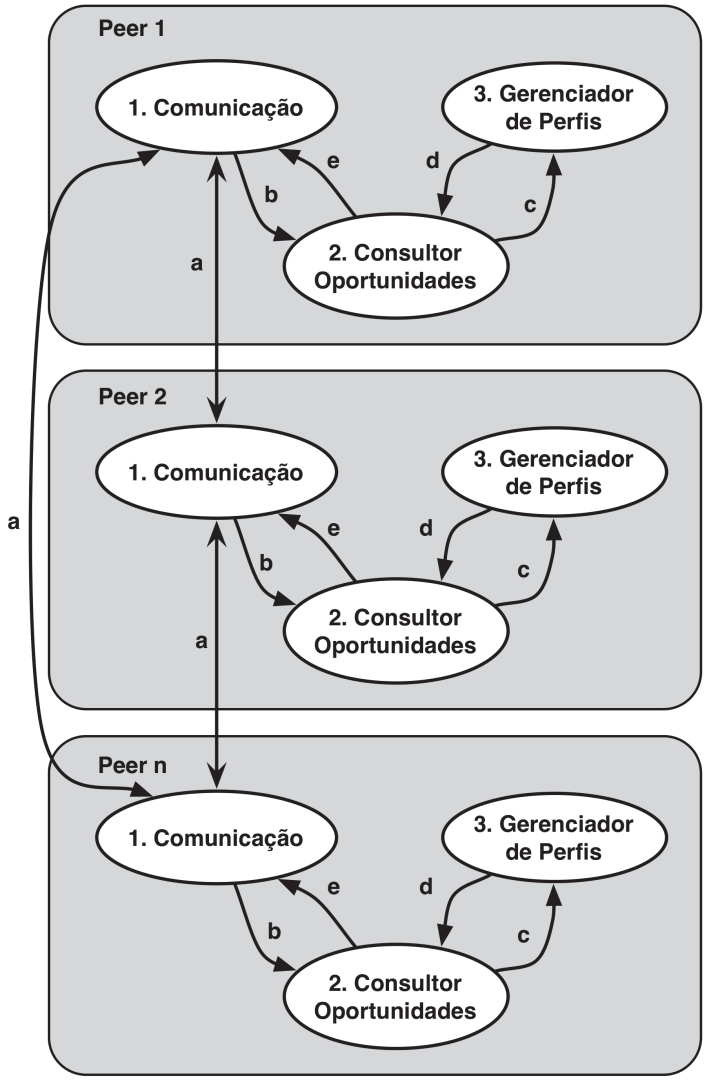
1. Descoberta de *peers* na rede: nesta etapa, os *peers* identificam-se disponíveis na rede (via *wifi* ou *bluetooth*). o U-Deal deve ter sido iniciado com um contexto básico, mesmo que o usuário não tenho criado nenhum contexto;

2. Troca de perfis: duas instâncias do U-Deal, assim que se descobrem na rede, compartilham seu perfil. Se um *peer* já conectado tiver sua conexão interrompida, assim que ele estiver disponível novamente, o procedimento de troca de perfil será refeito, a fim de que as instâncias envolvidas possam determinar se alguma atividade deve ser retomada ou se houve alteração nos contextos (forçando todo o processo deve ser iniciado do zero);

3. Análise de contexto: nessa etapa, cada instância do U-Deal vai comparar o perfil externo recebido com o seu próprio. As informação de perfil são geradas dinamicamente a partir do perfil estático do usuário e as trilhas da entidade. Dessa forma, ao comparar os perfis, as entidades terão capacidade de avaliarem se estão em um mesmo contexto, se possuem interesses convergentes e se já negociaram no passado (atribuição do nível de confiança);

4. Inicialização contexto de negociação: se entidades estão em contextos em comum e tem interesses convergentes, um novo sub-contexto será criado a fim de coordenar a etapa de negociação. Uma heurística vai colocar uma entidade em modo *ativo* e outra em modo *passivo*. Por exemplo, na parte 4 da Figura 17, a instância identificada como comprador

Figura 16: Visão geral do U-Deal



Fonte: elaborado pelo autor

(alguém que procura algum bem ou serviço) é setada em modo *ativo* e a instância vendedor (alguém que oferece/tem algum bem ou serviço) é setada como *passivo*. A instância ativa vai criar e gerenciar o novo contexto de negociação, incluindo e notificando a instância externa. Essa instância, por sua vez, vai registrar a notificação de criação e a sua própria inclusão no novo sub-contexto de negociação;

5. Inicio ciclo de negociação: a partir da etapa 4, onde os *peers* estão organizados em um contexto de negociação (gerenciado por uma das instâncias), inicia-se o processo de negociação propriamente dito, tendo como base os interesses em comum já identificados. Uma vez conectados através do contexto de negociação, os usuários podem trocar mensagens a respeito de uma oportunidade para ambos ou executar uma negociação automática entre os agentes (através da implementação de um *plugin* que dê o suporte a negociação automática, fora do escopo deste trabalho);

6. Finalização: após a conclusão de uma negociação, as instâncias do U-Deal precisam ser desconectadas do contexto de negociação e o mesmo deve ser desalocado. O resultado do processo de negociação deve ser registrado nas trilhas, juntamente com informações de contexto e de perfil das entidades envolvidas. Neste momento, cada instância deve avaliar todos os *peers* contactados. Esta informação vai estar presentes nas trilhas das entidades e será importante principalmente para futuras avaliações dos níveis de confiança em um determinado *peer* antes mesmo do processo de negociação (ver Figura 17, etapa 3).

Como dito anteriormente, as tarefas do modelo estão distribuídas entre agente computacionais, atuando de forma autônoma, de acordo com as características indicadas por Wooldridge e Jennings (1995). A Figura 18 mostra a organização geral desses agentes no U-Deal: as caixas de texto correspondem aos agentes e as setas a direção do fluxo de dados, sendo que as caixas em destaques representam agentes novos apresentados neste trabalho.

Na Figura 18, o agente Comunicação é um agente de serviço, atuando como suporte a outros agentes no modelo. Neste grupo, encontram-se os agentes responsáveis pela sensibilidade ao contexto, gerenciamento de trilhas e perfis e identificação de oportunidades. Estes agentes são Gerenciador de Contexto, Gerador de Perfil, Gerenciador de Perfis, Gerenciador de Trilhas e Consultor de Oportunidades. Cada instância do U-Deal é executada em um dispositivo, sendo que a ausência de um agente compromete apenas as funcionalidades que dele dependem. A partir da organização geral do U-Deal (Figura 18), é possível realizar um mapeamento entre os elementos principais em um cenário de negociação e os agentes do modelo (ver Tabela 4).

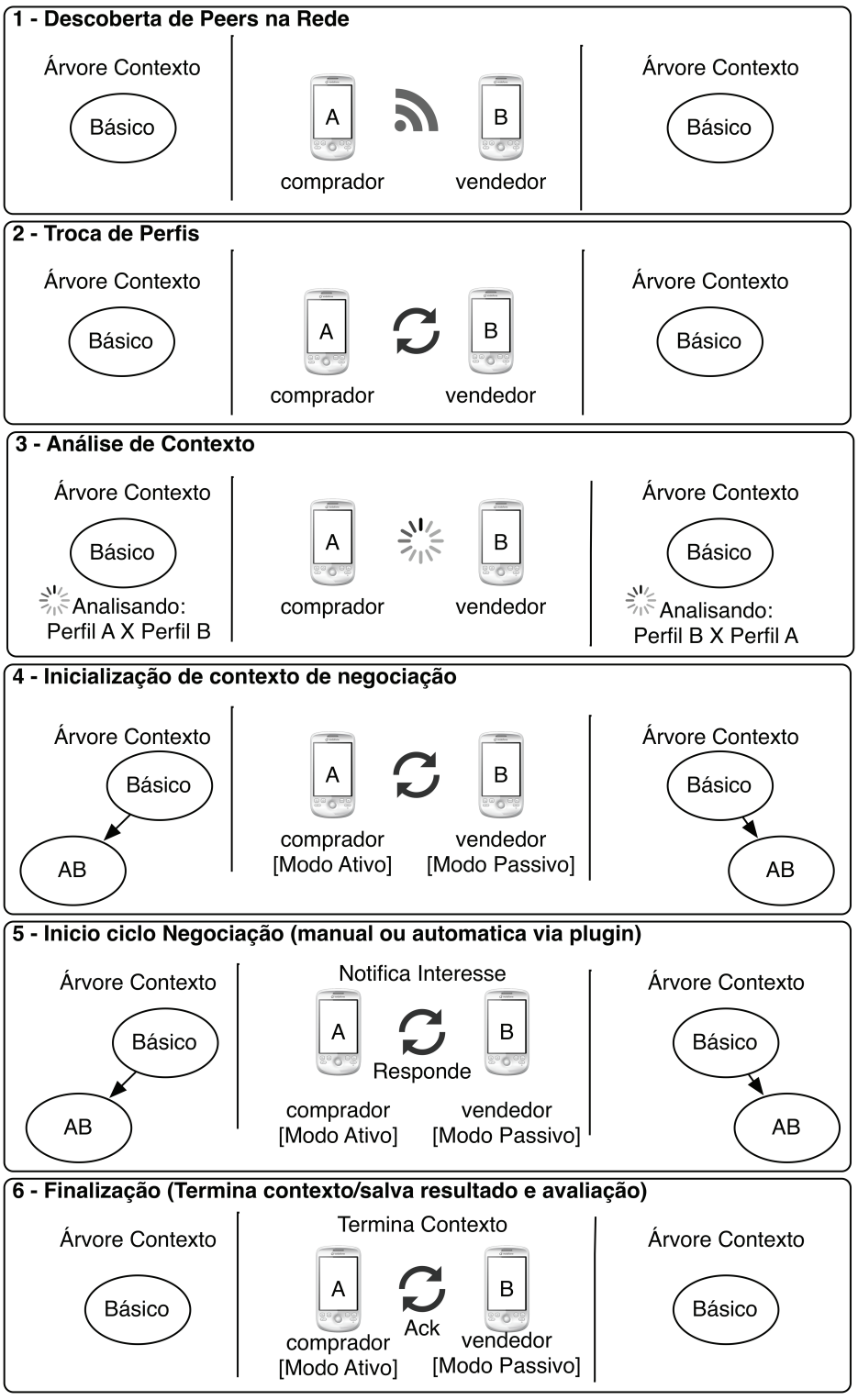
Tabela 4: Mapeamento entre elementos da negociação e os agentes do modelo

Elemento negociação	Agente(s) no modelo
Interesses	Perfil, Contexto, Trilhas
Alternativas	Perfil, Contexto, Trilhas
Opções	Perfil, Contexto, Trilhas
Legitimidade	Oportunidades, Contexto, Perfil
Compromissos	Perfil, Contexto, Trilhas
Relacionamentos	Perfil, Contexto, Trilhas
Comunicação	Comunicação
Tempo	Contexto
Conformidade	Contexto, Perfil
Contexto	Perfil, Contexto, Trilhas

Fonte: Elaborado pelo autor

A seguir, os agentes do sistema serão apresentados em detalhe, usando a metodologia Prometheus.

Figura 17: Etapas do processo de negociação

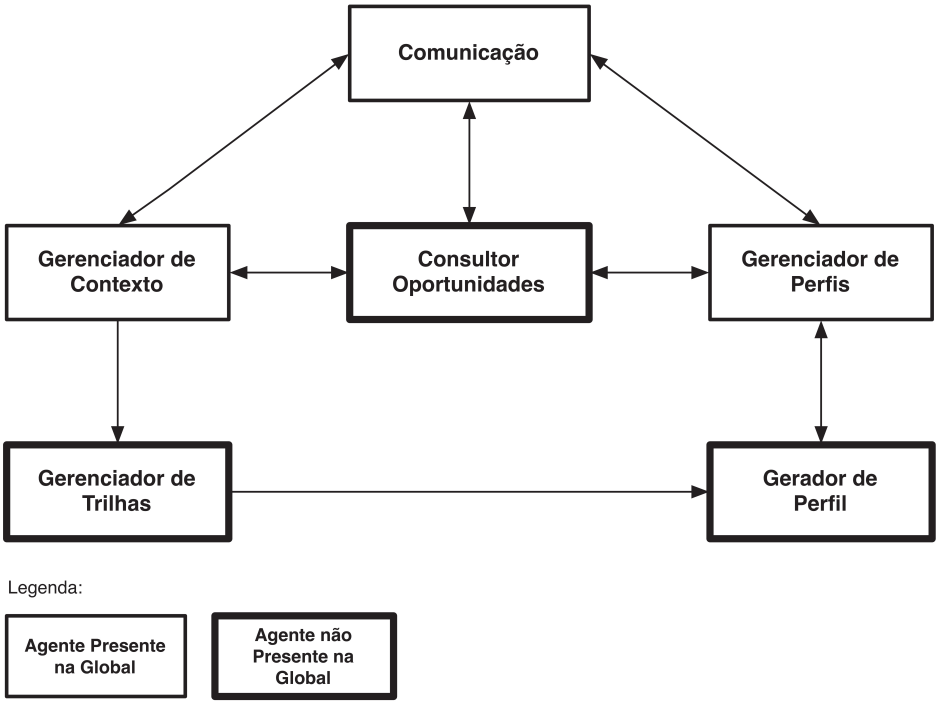


Fonte: elaborado pelo autor

4.2 Diagrama Geral do Sistema

Nesta sessão, será apresentado o diagrama geral do sistema utilizando metodologia Prometheus.

Figura 18: Organização dos Agentes no U-Deal



Fonte: Elaborado pelo autor

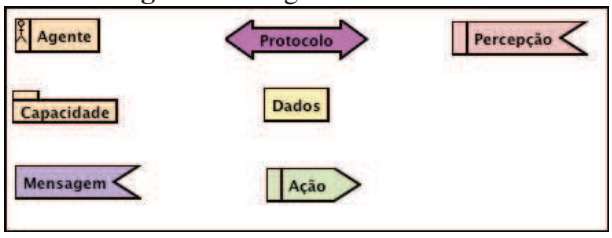
O Prometheus é uma metodologia detalhada de especificação para Sistemas Multiagentes e está dividida em três fases.

Na fase de especificação do sistema (*System Specification*) deve-se especificar o que o sistema deve ser capaz de fazer (*System Goals*), desenvolver cenários de caso de uso que ilustrem a operação do sistema, identificar as funcionalidades e especificar uma interface com o ambiente em termos de ações e percepções.

A partir das saídas da fase de especificação, a fase de desenvolvimento da arquitetura (*Architectural Design*) determina quais serão e como se dará a interação entre os agentes.

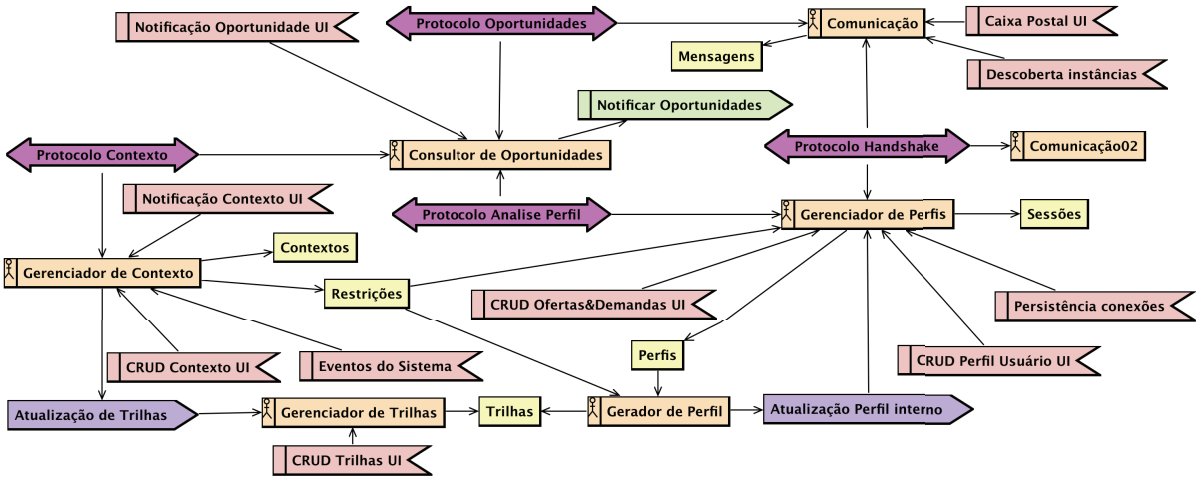
Na fase de projeto detalhado (*Detailed Design*), desenvolve-se a estrutura interna de cada agente e defini-se como cada um realizar a sua tarefa dentro do sistema. Refina-se cada agente definindo capacidades, eventos internos, planos e dados acessados. A Figura 19 mostra o significado dos símbolos utilizados na especificação.

Figura 19: Legenda Prometheus



Fonte: Eclipse PDT IDE

Figura 20: Arquitetura Geral U-Deal



Fonte: Elaborado pelo autor

O diagrama geral do modelo U-Deal, bem como o detalhamento dos agentes a seguir apresentados, são o resultado das fase de *Architectural Design* e *Detailed Design* respectivamente, utilizando a metodologia descrita. O diagrama do modelo U-Deal é baseado no diagrama da Global, contudo, o U-Deal apresenta agentes voltados para a área de comércio ubíquo, já que a Global é focada em educação ubíqua. Os agentes incluídos no U-Deal são:

- Gerenciador de Trilhas: responsável pela manipulação das trilhas do modelo;
- Gerador de Perfil: tem a função de extrair um perfil dinâmico da entidade a partir de suas trilhas e perfil estático do utilizador;
- Consultor de Oportunidades: agente principal do modelo responsável pelo processo de negociação e identificação de oportunidades.

No modelo, a comunicação entre os agentes de diferentes instâncias é feita através do agente Comunicação, utilizando o protocolo *HandShake*. A função desse protocolo é controlar as comunicações de entrada e saída do modelo, a fim de verificar a autenticidade e o *status* das mensagens, de acordo com o contexto das interações entre as entidades. Por exemplo, uma mensagem de notificação de oportunidade de negócio, enviada para uma instância não conectada será considerada inválida e descartada. Além disso, o protocolo é usado para coordenar a persistência das conexões, garantindo que uma mensagem seja entregue mesmo que a conexão tenha sido interrompida e mais tarde retomada.

Assim como na Global, o U-Deal possui uma estrutura de *listeners*, onde qualquer agente pode enviar uma mensagem para outro agente, notificando alguma mudança de estado ou evento, como por exemplo, mudança de contexto, atribuição de nível de confiança ou notificação de recomendação.

A Figura 20 mostra o diagrama geral do sistema, onde estão representadas as principais relações entre os agentes.

As próximas subseções descrevem os agentes, que compõem o U-Deal, em maior detalhe.

4.3 Comunicação

Este agente é baseado no agente Conectividade apresentado por Oliveira (2010). O agente Comunicação é responsável, principalmente, pela descoberta de outras instâncias do U-Deal (disponíveis na rede) e pela troca de mensagens entre os agentes, utilizando o padrão FIPA-ACL. Na interação entre agentes de diferentes instâncias, o agente Comunicação deve descobrir a melhor forma de entregar a mensagem, abstraindo para os demais agentes a camada de comunicação. O modelo de entrega é o de *best-effort* (COMER; YAVATKAR, 1989).

A Tabela 5 mostra a estrutura das mensagens, compatível com o padrão FIPA-ACL.

Tabela 5: Estrutura Parâmetros FIPA-ACL

Parâmetro	Tipo de comunicado	Dado comunicado
Performative	tipo de comunicado	Denota o tipo de ação comunicativa
Sender	participante na comunicação	Identidade do agente que enviou a mensagem
Receiver	participante na comunicação	Identidade dos destinatários da mensagem
reply-to	participante na comunicação	Identidade dos destinatários do retorno da mensagem
Content	conteúdo da mensagem	Denota o conteúdo da mensagem
Language	descrição do conteúdo	Denota a linguagem em que o conteúdo é expresso
Encoding	descrição do conteúdo	Denota a codificação do conteúdo
Ontology	descrição do conteúdo	Denota a ontologia do conteúdo
Protocol	controle da comunicação	Denota o protocolo de interação que o agente está usando na mensagem ACL
conversation-id	controle da comunicação	Identifica a sequência de mensagens que juntas formam uma conversação
reply-with	controle da comunicação	Identifica a resposta esperada
in-reply-to	controle da comunicação	Identifica qual mensagem está sendo respondida
reply-by	controle da comunicação	Hora/data limite até a qual o agente emissor esperá por uma resposta

Fonte: Oliveira (2010)

No modelo, cada entidade será representada por um identificador único, formado pelo sobrenome do usuário e seu código fiscal, no seguinte formato: "SOBRENOME:CodigoFiscal". O uso do código fiscal permite a utilização de uma chave unívoca e internacionalmente difundida (ainda que com diferentes formatos). Um exemplo para o Brasil seria "MOTTA:87654612345".

Da mesma forma que na Global, o agente Comunicação do U-Deal utiliza um protocolo

denominado *Handshake* (OLIVEIRA, 2010) para estabelecer contato com agentes em outras instâncias. De acordo com o diagrama apresentado na Figura 21, qualquer mensagem entre agentes de diferentes instâncias deve passar pelo protocolo *Handshake*, onde o agente Comunicação, primeiramente, vai verificar se o destinatário está conectado e se existe algum processo de identificação de oportunidades em andamento. Em caso positivo, o agente Comunicação faz a entrega da mensagem ao destinatário. Caso o usuário não estiver conectado, o agente Comunicação vai solicitar o perfil dessa entidade, a fim de verificar se trata-se de um novo acesso ou se a entidade já estava conectada e se alguma tarefa deva ser resumida. Além disso, o perfil da entidade externa vai ser usado para determinar o nível de confiança entre as instâncias. Analogamente, o agente Comunicação deve validar, através de perfil, todas as mensagens e notificações que tenham origem em outras instâncias e sejam destinadas a agentes de sua própria instância.

A Figura 22 mostra a definição das capacidades e mensagens do agente Comunicação. Algumas capacidades do agente foram adaptadas da Global, sendo que o capacidade *Handshake* foi totalmente alterada. As capacidades do agente Comunicação são:

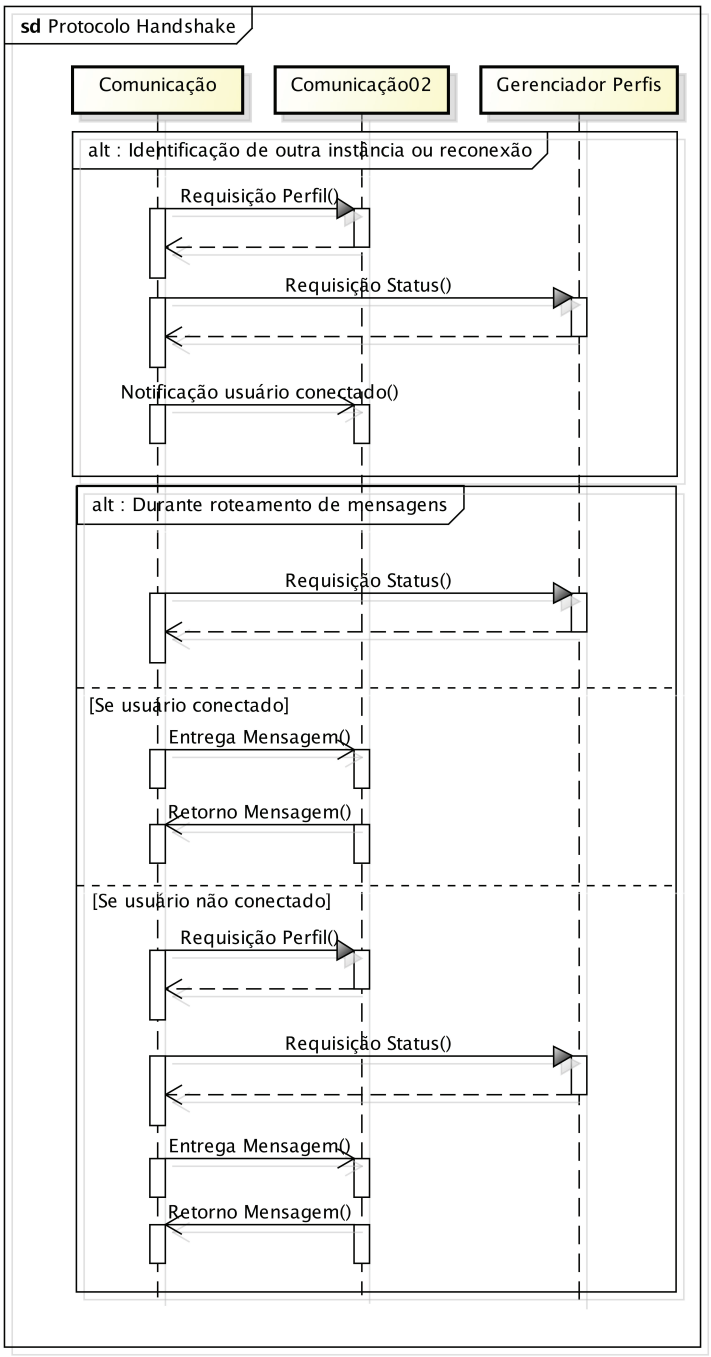
- Caixa Postal: tem a função de transportar mensagens e notificações entre diferentes instâncias do modelo, sendo que essas mensagens podem ser endereçada a um usuário específico ou a usuários em um contexto inteiro. A Caixa Postal pode transportar mensagens originadas em agentes ou mesmo pelo usuário, através da interface gráfica;

- Conectividade: capacidade que representa o protocolo *Handshake*, que estabelece a primeira comunicação entre os agentes de instâncias diferentes (ver diagrama Figura 21) e coordena o transporte de mensagens entre diferentes instâncias do modelo. A descoberta de serviços na rede consiste em identificar outras instâncias do U-Deal, estabelecer o primeiro contato e notificar os *listeners* adequados. Além disso, a agente deve ser capaz de identificar quando uma conexão interrompida está sendo retomada e notificar os agentes interessados. O agente deve abstrair para a aplicação a conexão com redes disponíveis, sendo que o mesmo deve encontrar a melhor maneira de entregar a mensagem, já que o modelo deve suportar comunicação P2P em redes sem fio comuns, redes sem fio *ad hoc* ou mesmo conexões *bluetooth*;

- CRUD *listeners*: capacidade de registrar *listeners* e notificar usuários e agentes conectados;

- Determinação confiança: O protocolo *Handshake*, quando necessário, pode determinar o nível de confiança em uma entidade externa e também atualizar o perfil dessa entidade localmente. A determinação da confiança é feita através do histórico de contatos com determinada entidade.

Figura 21: Protocolo HandShake



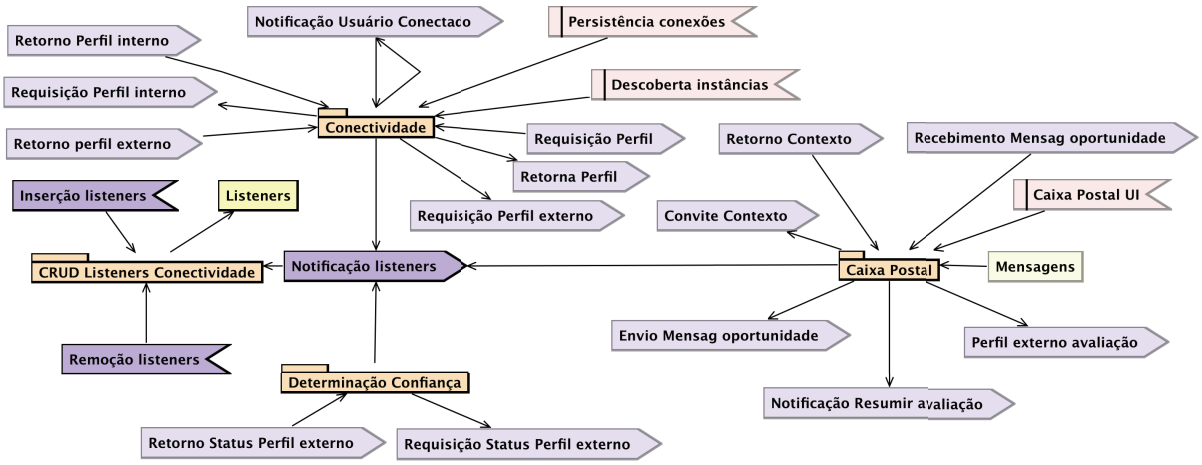
Fonte: Elaborado pelo autor

4.4 Consultor de Oportunidades

O Consultor de Oportunidades é um agente para comércio ubíquo modelado no U-Deal, não presente na Global, pois esta é uma infraestrutura específica para educação ubíqua. Contudo, um componente similar existe no MUCS.

O agente tem a função de identificar oportunidades de negócios (compras, vendas, serviços, projetos) e dar suporte ao processo de negociação. O agente Consultor de Oportunidades é

Figura 22: Detalhamento agente Comunicação



Fonte: Elaborado pelo autor

acionado pelo agente Comunicação, após a descoberta de uma outra instância na rede. Ao ser notificado, o agente consultor recebe o perfil dessa instância, juntamente com as informações a respeito da confiança nesse usuário. A partir disso, o consultor deve iniciar uma comparação entre o perfil recebido e o perfil da entidade local, para determinar se existem oportunidades em potencial de negócio (os perfis são gerados com base nas trilhas e perfil estático das entidades).

Basicamente, Consultor de Oportunidades, ao identificar automaticamente um potencial de negócio, contata o agente Gerenciador de contexto e cria um contexto específico para suportar o processo de negociação. Além disso, o Consultor de Oportunidades também avalia se existem no momento outros contextos de negociação já iniciados onde este perfil poderia se encaixar. Após solicitar a criação desse contexto, o Consultor de Oportunidades convida a entidade externa a se conectar a esse contexto. Dentro do contexto de negociação, as instâncias do modelo podem, através de intervenção manual dos usuários (utilizando suas caixas postais), trocar mensagens e evoluir na negociação. Uma oferta ou demanda pode estar configurada para suportar 1 ou n usuários, trocando mensagens dentro de um mesmo contexto de negociação ou ainda um entidade pode estar negociando com n entidades em n contexto de negociação diferentes. Um mesma entidade pode estar ofertando um produto, em um contexto e oferecendo um serviço em outro, ao mesmo tempo.

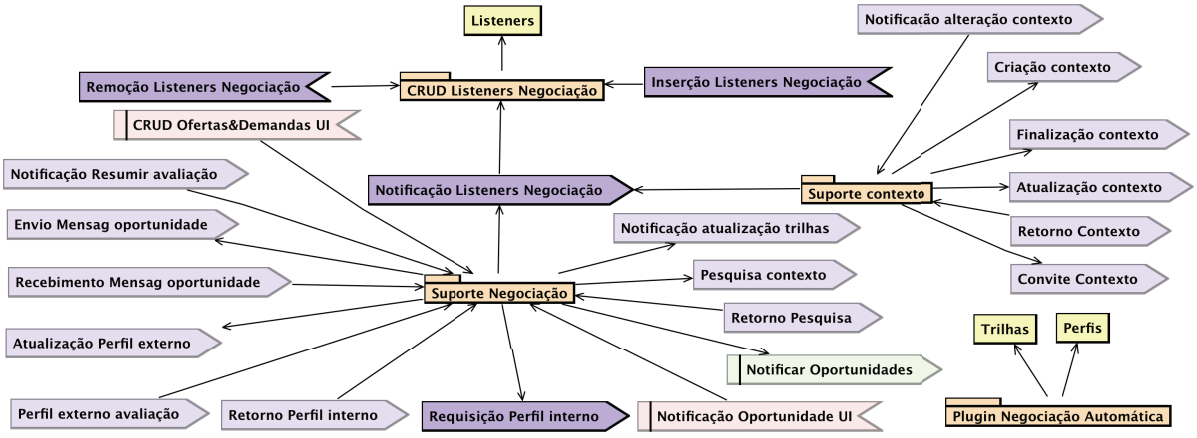
O Consultor de Oportunidades deve ser flexível o suficiente para permitir a negociação automática entre os agentes, através de implementação de um *plugin* (a implementação deste *plugin* está fora do escopo deste trabalho). Isto permite que, dentro de certos limites configuráveis, os agentes possam trocar mais informações dentro de um mesmo contexto de negociação, a fim de que ambos possam aproximar-se de seus objetivos (método de negociação ganha-ganha de acordo com Nash (1950)). Essa aproximação dos objetivos pode ser suficiente ou não para a recomendação do fechamento do negócio. No escopo desse trabalho, apenas a negociação manual está presente, sendo que a análise sobre as trilhas é indireta (através do perfil dinâmico) e utilizada apenas no processo inicial de identificação de oportunidade. Contudo, este *plugin*

permite automatizar também o processo de negociação, dando ao consultor acesso aos dados brutos das trilhas e perfil das entidades, o que poderia fornecer informações valiosas às mesmas no sentido de resolver conflitos e ambiguidades ou ainda explorar um diferente aspecto de uma área de interesse. A seguir, são listadas as capacidades do agente Consultor de Oportunidades:

- CRUD *listeners* de Negociação: permite inserir, remover e notificar agentes ou usuários interessados em eventos de negociação;
- Suporte contexto: essa capacidade é uma interface para o agente Gerenciador de Contexto, a fim de suportar as tarefas de pesquisa e manipulação de contextos, dentro do processo de negociação;
- Suporte negociação: essa capacidade é o *core* do agente, onde acontecem as eventos de identificação de oportunidades e o posterior processo de notificação, montagem de contexto e troca de mensagens com fins de negociação. É nessa parte do agente que uma negociação inicia e termina, onde os perfis são comparados e onde a notificação para atualização das trilhas é disparada ao fim do processo;
- *Plugin* negociação automática: ao ser implementado, esse *Plugin* assume o papel de Suporte a negociação, tendo acesso a toda a sua interface com os demais agentes. Além disso, o *Plugin* tem acesso direto às trilhas e aos perfis, permitindo inferências mais completas, dando subsídio para a implementação de um algoritmo para negociação automática entre os agentes consultores de diferentes instâncias.

A Figura 23 mostra as capacidades do agente, bem como as mensagens suportadas.

Figura 23: Detalhamento agente Consultor de Oportunidades



Fonte: Elaborado pelo autor

4.5 Gerenciador de Perfis

O agente Gerenciador de Perfis é responsável pelo controle e manipulação das informações do usuário, disponibilizando-as aos demais agentes do U-Deal. Além disso, tem a função de gerenciar os desejos e interesses do utilizador de forma explícita, como interesses, ofertas e demandas de produtos e serviços, através de cadastro estático de perfil, ofertas e demandas. Ainda, o agente Gerenciador de Perfis também tem a função de administração dos dados de perfil de outras entidades, com a qual o usuário já travou contado (como o agente de Perfis Externos da Global).

O modelo de dados utilizado é baseado no modelo proposto no MUCS por Franco et al. (2011). A estrutura do meta modelo contém informações de identificação, preferências, ofertas e demandas.

Tanto o perfil da entidade local quanto os perfis das entidades externas serão registrados utilizando a seguinte estrutura, onde *estático* refere-se a atributos que serão manipulados manualmente pelo usuário e *dinâmicos* as informações extraídas das trilhas pelo agente Gerados de Perfil:

- Id: identificador único que representa o perfil, composto pela chave da entidade | estático;
- Identificação: subestrutura com mais dados de identificação do usuário | estático;
- *timestamp*: indica a data e hora da geração do perfil | dinâmico;
- Descrição: descrição textual do perfil | estático;
- Ofertas: lista com as eventuais ofertas da entidade | estático;
- Demandas: lista com as eventuais demandas da entidade | estático;
- Áreas de interesse: lista com as 5 principais áreas de interesse | dinâmico;
- Ofertas não finalizadas: lista com as 5 principais ofertas não finalizadas | dinâmico;
- Demandas não finalizadas: lista com as 5 principais demandas não finalizadas | dinâmico;
- Ofertas finalizadas: lista com as 5 principais ofertas finalizadas | dinâmico;
- Demandas finalizadas: lista com as 5 principais demandas finalizadas | dinâmico;
- Ontologia: indica a estrutura que representa as oportunidades no perfil (Produtos, serviços, interesses);
- Relacionamento: listas com as entidades conhecidas (id, rate, % sucesso, *flag black-list*) | dinâmico.

A Figura 24 mostra as capacidades do agente Gerenciador de Perfis:

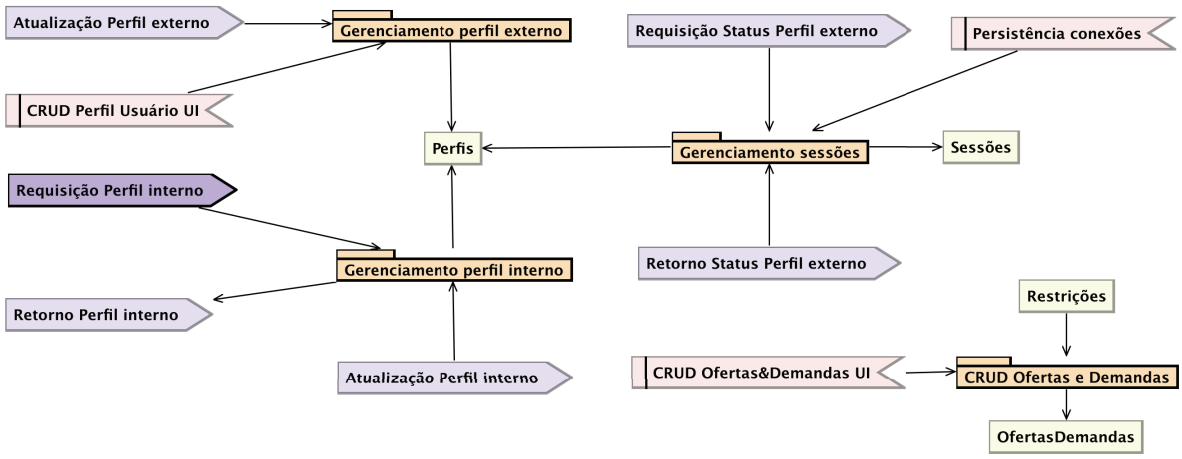
- CRUD Ofertas e Demandas: essa capacidade permite inserir, manualmente através da interface gráfica, ofertas e demandas de interesse e associá-las a restritores, a fim de que elas sejam válidas em um determinado contexto;

- Gerenciamento perfil interno: responsável pelo armazenamento e manipulação do perfil local da entidade, gerado dinamicamente pelo agente Gerador de Perfil, a partir das trilhas da entidade e do perfil estático do usuário. Esse perfil estático é um cadastro dos dados básicos do utilizador e das suas ofertas e demandas. Também tem a função de disponibilizar o perfil da entidade local para os agentes interessados;

- Gerenciamento perfil externo: permite o armazenamento e manipulação de perfis de entidades externas. Essas informações são atualizadas pelo agente Comunicação, através do protocolo *Handshake* e podem ter seu *status* atualizado pelo agente Consultor de Oportunidades.

- Gerenciamento de Sessões: essa capacidade permite registrar perfis de entidade em sessões, a fim de identificar as entidades conectadas e suas ações no sistema durante todo o processo. Além disso, o registro em sessões permite retomar as tarefas de usuários que tenham reconectado e rejeitar aquelas reconexões consideradas inválidas após a reavaliação do perfil da entidade externa.

Figura 24: Detalhamento agente Gerenciador de Perfis



Fonte: Elaborado pelo autor

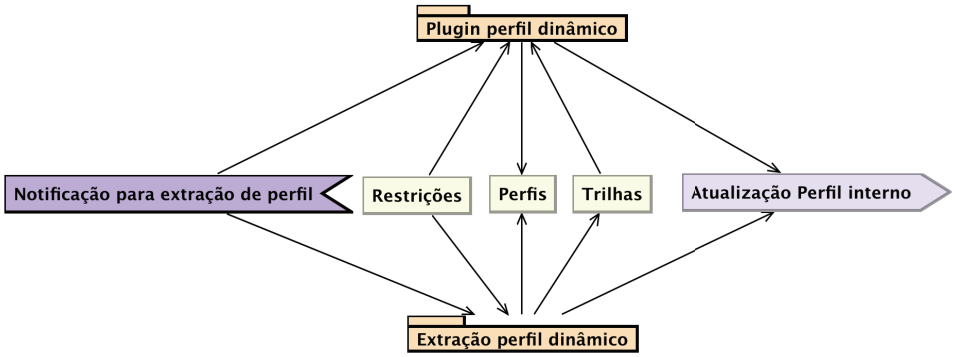
4.6 Gerador de Perfil

O agente Gerador de Perfil é responsável pela atualização do perfil local dinamicamente, baseado nas trilhas e nos dados estáticos do usuário (cadastro de dados pessoais, interesses). O perfil dinâmico pode ser extraído em intervalos regulares ou sob demanda, sendo que o agente

deve ser disparado por eventos do sistema ou mudanças de contexto (através da implementação de restritores ou mesmo uma notificação direta para proceder com a extração). As informação serão registradas utilizando a mesma estrutura descrita no agente Gerenciador de Perfis.

A Figura 25 mostra as capacidades do agente:

Figura 25: Detalhamento agente Gerador de Perfil



Fonte: Elaborado pelo autor

- Extração perfil dinâmico: essa capacidade permite que o agente Gerador de Perfil periodicamente examine as trilhas da entidade local e extraia determinadas informações. O resultado desta extração será utilizado para atualizar os seguintes atributos do Perfil interno da entidade:
- Áreas de interesse: lista com as 5 principais áreas de interesse;
- Ofertas não finalizadas: lista com as 5 principais ofertas não finalizadas;
- Demandas não finalizadas: lista com as 5 principais demandas não finalizadas;
- Ofertas finalizadas: lista com as 5 principais ofertas finalizadas;
- Demandas finalizadas: lista com as 5 principais demandas finalizadas;
- Relacionamento: listas entidades conhecidas (id, rate, % sucesso, *flag black-list*).
- *Plugin* perfil dinâmico: assim como no agente Consultor de Oportunidades, o Gerador de Perfil permite a implementação de um algoritmo alternativo para a extração do perfil dinâmico. Através dessa capacidade do agente, é possível implementar, por exemplo, um sistema de inferência sobre as trilhas, que utilize uma base de regras e modelos em conjunto com um raciocinador semântico (*Semantic Reasoner*). Ao ser implementado, o *Plugin* assume as funcionalidades de extração de perfil. A utilização de um raciocinador ou do algoritmo padrão para extração dos dados das trilhas deve ser transparente para o resto do modelo.

4.7 Gerenciador de Contexto

O agente Gerenciador de contexto aqui apresentado é baseado no agente Contexto da Global. No U-Deal, esse agente é responsável pela coordenação dos contextos no modelo, mapeando a movimentação dos usuários pelos contextos e alertando os demais agentes sobre essas alterações.

A principal contribuição do U-Deal na parte de gestão de contexto é a introdução do suporte a Trilhas. Dessa forma, o suporte a histórico de movimentação entre contextos presente na Global foi transferido como uma das capacidades do agente Gerenciador de Trilhas. Neste caso, as trilhas serão formadas tendo como base as informações de contexto, perfil e negociação e a sua atualização deve ser disparada por eventos do sistema como movimentação por contextos, descoberta de *peers* na rede, criação e manipulação de contextos, bem como identificação de oportunidades e notificações de negociação.

Assim como na Global, os contextos podem ser derivados de outros, uma vez que dentro de um contexto podem ser formados sub-contextos apenas pela adição de novas características a ele. Cada instância do U-Deal, ao ser inicializada, vai criar automaticamente um contexto básico, com restrições mínimas. O usuário também pode configurar quais as restrições este contexto básico terá. Todos os demais contextos de negociação serão criados como sub-contextos deste contexto básico.

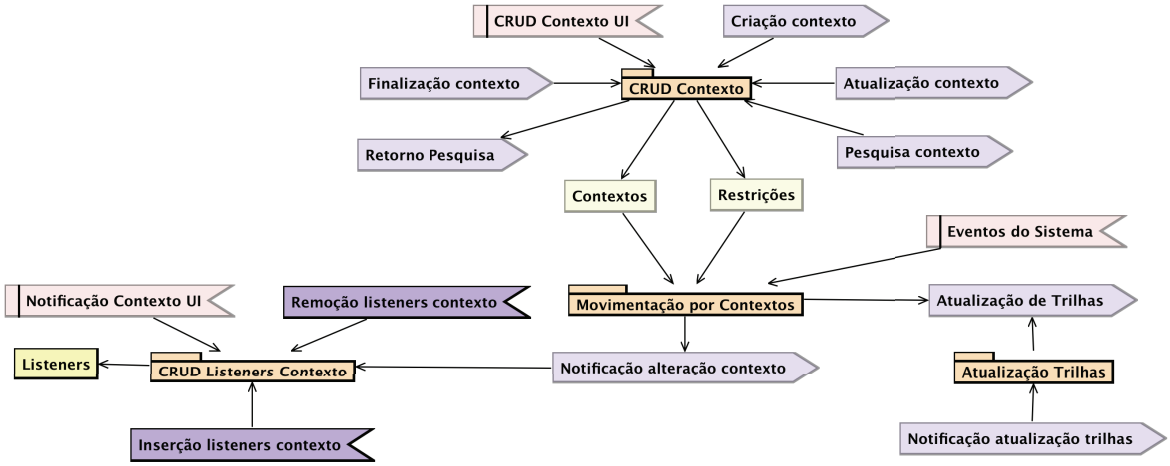
O modelo permite que os usuários criem contextos para configuração de diferentes tipos de situações. Por exemplo, pode-se criar um contexto para um leilão ou para uma negociação em grupo.

A descrição dos contextos é baseada em metadados, dentre os quais destacam-se:

- Id: identificador único que representa o contexto;
- Descrição: descrição textual do contexto;
- Ontologia: indica a estrutura que representa as informações no contexto;
- Responsável: identificador do criador do contexto;
- Centralizado: identifica se o contexto é centralizado no seu criador;
- Restrições: contém a lista de restrições do contexto;
- id parent: permite a formação de uma árvore de informa do contexto (que devem incluir a lista das entidades que participaram de uma negociação neste contexto, a lista das oportunidades avaliadas e o resultado final do processo).

O modelo utiliza restrições e Restritores para o gerenciamento dos contextos (OLIVEIRA, 2010), sendo que, na Global, os Restritores foram modelados em um agente dedicado. Contudo, no U-Deal os Restritores foram integrados no agente Gerenciador de Contexto, que

Figura 26: Detalhamento agente Contexto



Fonte: Elaborado pelo autor

monitora o ambiente e identifica as mudanças de estado nas restrições, notificando os *listeners* associados. Os restritores são associados na inicialização do agente Gerenciador de Contexto, podendo ser adaptados de forma dinâmica, dependendo dos recursos do dispositivo. Por exemplo, restrições do tipo *GlobalPosition* podem ser associadas a um restritor de localização por GPS ou por triangulação de antena, caso o dispositivo não tenha o equipamento.

A Figura 27 exemplifica o uso de restrições. O contexto possui uma restrição temporal para os dias da semana, uma restrição lógica de operação "ou" formada por duas restrições temporais para os horários de atuação, uma restrição física que representa a área da atuação do contexto e uma restrição que representa os usuários que podem participar do contexto.

A Figura 26 mostra as capacidades do agente Contexto e suas mensagens suportadas. As capacidades são assim descritas:

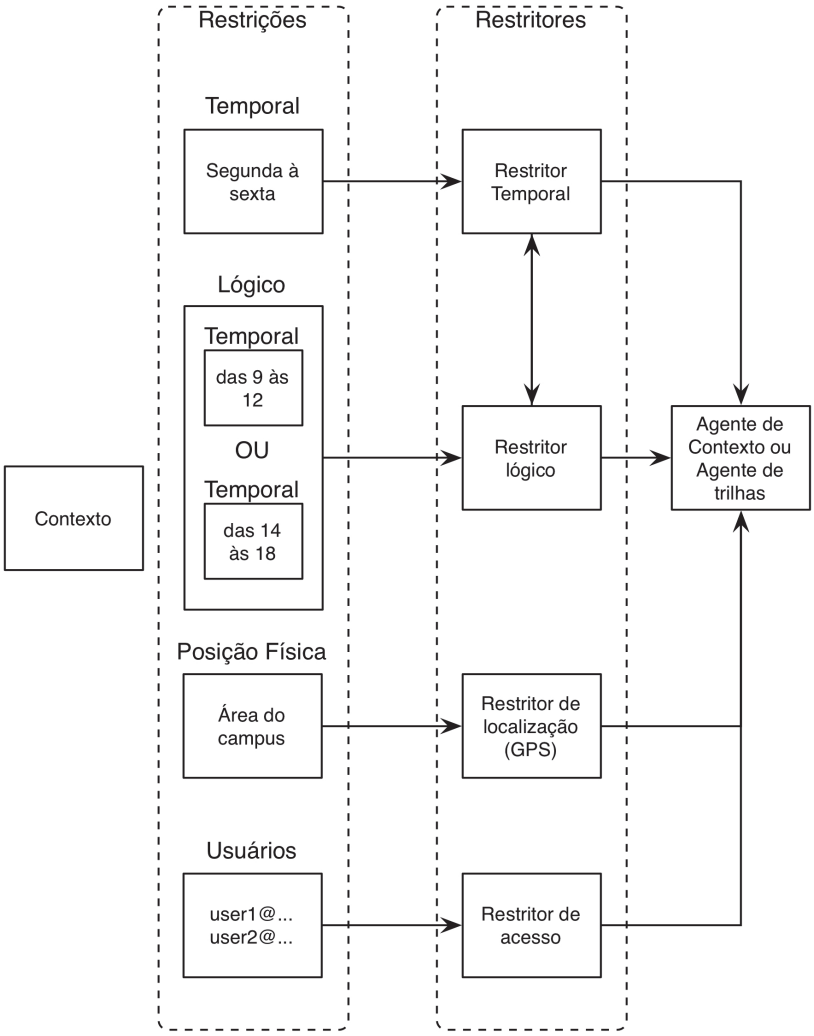
- CRUD Contexto(*Create, Retrieve, Update e Delete*): são as operações para criar, buscar, atualizar e deletar informações de contexto. Cada uma dessas operações dispara uma notificação aos *listeners* de contexto que informa os agentes registrados sobre alterações no mesmo. Os contextos são implementados através da configuração de Restritores;

- CRUD *listeners* de contexto: permite inserir, remover e notificar agentes ou usuários interessados em eventos de contexto;

- Movimentação por contextos: essa capacidade é responsável por monitorar a movimentação entre os contextos, através da avaliação dos Restritores de um determinado contexto. A partir da identificação de mudança de contexto, a capacidade tem a função de notificar os *listners* de contexto registrados;

- Atualização de Trilhas: é a capacidade de coordenar a atualização de trilhas, notificando e fornecendo informações para o agente Gerenciador de Trilhas.

Figura 27: Exemplo do uso de restritores



Fonte: Oliveira (2010)

4.8 Gerenciador de Trilhas

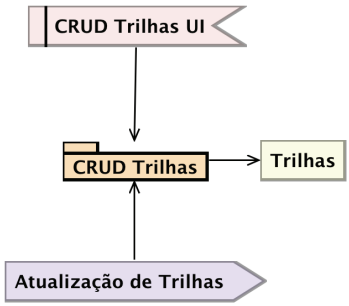
O agente Gerenciador de Trilhas é um dos agentes introduzidos pelo modelo U-Deal, não fazendo parte da Global nem do MUCS. Este agente é responsável pelo armazenamento e recuperação de dados de histórico de contexto. *Listeners* ou restritores podem ser usados para disparar o armazenamento da trilha. Ainda, o armazenamento pode ocorrer por solicitação direta de um agente da instância local. Além disso, o agente Gerenciador de de Trilhas tem a capacidade de oferecer busca nas trilhas para outros agentes. Dessa forma, o agente funciona como uma interface de entrada e saída para as trilhas.

As trilhas estendem o tipo de estrutura de um contexto, servindo, não meramente como uma tabela de contextos, mas como uma tabela de *snapshots* de contextos, em um determinado momento. Um mesmo contexto pode aparecer, em ordem cronológica, diversas vezes nas trilhas, mas cada registro irá armazenar o estado do contexto até aquele momento. De fato, as trilhas

tem, como uma das suas principais funções, o registro do produto final do modelo: o resultado da avaliação de uma oportunidade de negócio e de um eventual processo de negociação posterior.

A Figura 28 mostra a capacidade e as mensagens suportadas pelo agente. A capacidade CRUD Trilhas tem a função de servir como interface para operações de inserção, consulta, atualização e deleção de trilhas.

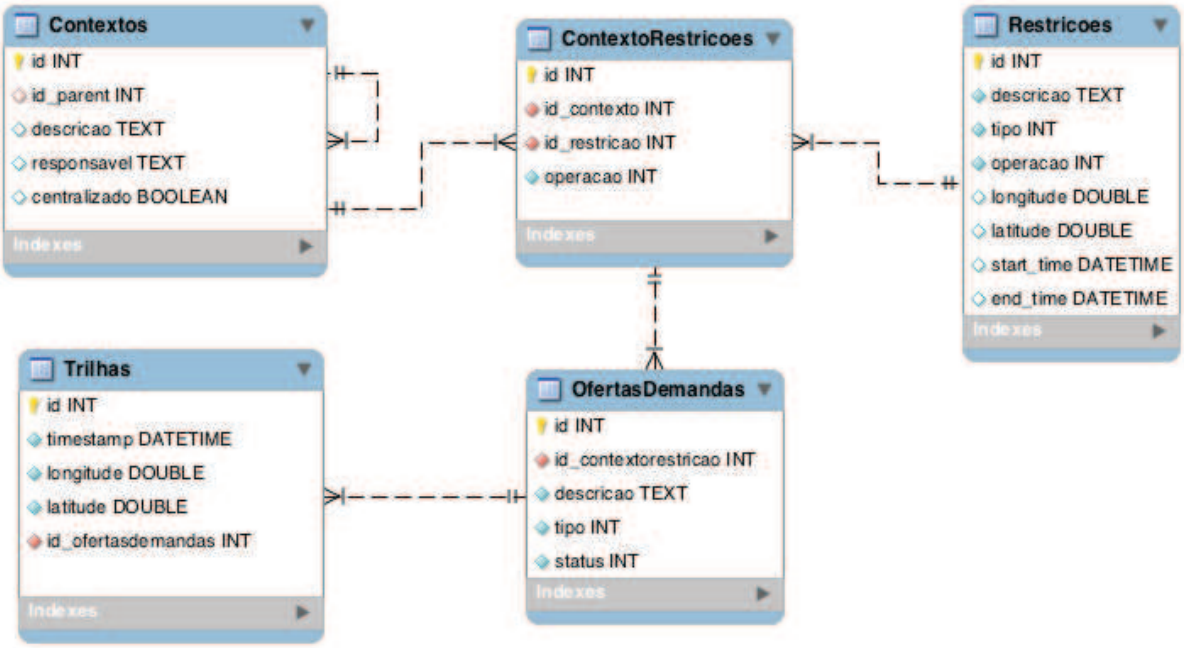
Figura 28: Detalhamento agente Trilhas



Fonte: Elaborado pelo autor

A Figura 29 mostra o diagrama ER da relação Contexto/Trilhas. A tabela Contexto serve como tabela identidade de um contexto e subcontexto, com atributos como "descrição"e "centralizado". Além disso, através da autorreferência via chave id parent, é possível montar uma árvore de contexto e seus subcontextos. A tabela de restrições contém as informações de restrição que permitem configurar o contexto, sendo que a tabela ContextoRestritores implementa a relação entre um Contexto que pode ter 0, 1 ou n restrições. Os campos "operação"nas tabelas Restritores e ContextoRestritores permitem a combinação de operações lógicas entre as restrições. Dado que uma Oferta ou uma Demanda contém restrições (por exemplo uma oferta pode apenas ser válida em algum local ou momento), uma das maneiras de se implementar as trilhas é como um *container* com *timestamp* e localização da entidade, no momento em que se gerou um registro do estado do sistema.

Figura 29: Moldelo ER da Relação Contexto/Trilhas



Fonte: Elaborado pelo autor

5 ASPECTOS DE IMPLEMENTAÇÃO E AVALIAÇÃO

5.1 Implementação

Um protótipo do U-Deal foi desenvolvido na plataforma *Android* 4.3, utilizando a linguagem de programação Java. O Java é uma linguagem de alto nível, orientada a objetos, de código aberto e independente de arquitetura. Além disso, é amplamente utilizada em aplicações comerciais e científicas. Já o *Android* é um sistema operacional baseado no *kernel* do Linux para dispositivos móveis e também possui código aberto.

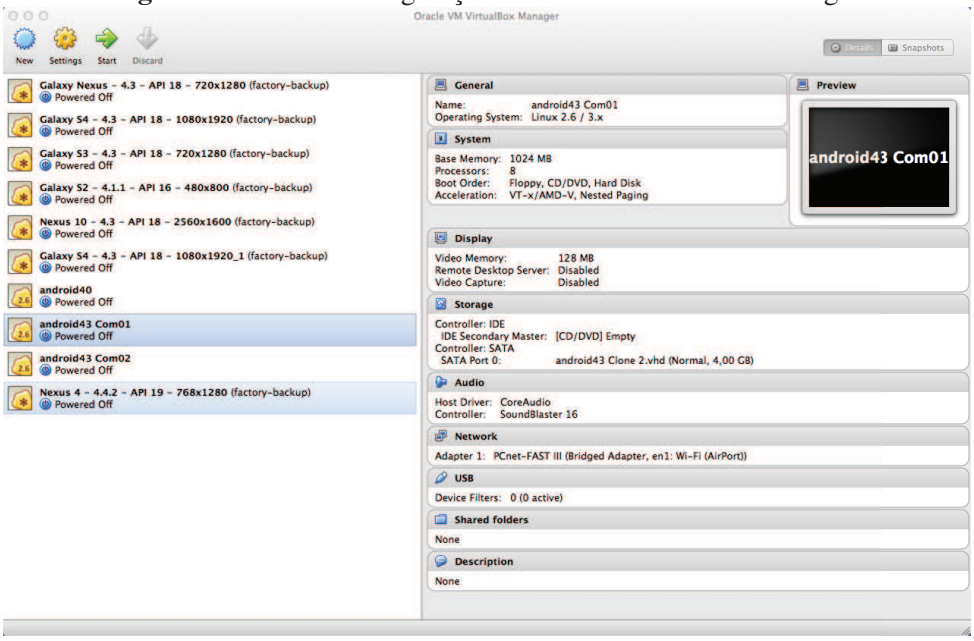
O protótipo foi totalmente desenvolvido utilizando, como emuladores, as imagens do *Android* criadas pelo projeto Android-x86[1]. O objetivo do projeto é disponibilizar versões do *Android* que rodem diretamente na plataforma x86. Essa é uma iniciativa dentro do âmbito do *Android Open Source Project*, onde o Google disponibiliza o código fonte do *Android* e incentiva a adaptação do mesmo a outras plataformas além da ARM. As dezenas de imagens do sistema operacional estão disponíveis na página do projeto e podem ser instaladas diretamente nos PC's, juntamente com um *bootloader* para seleção no momento da inicialização. Para este protótipo, foi selecionada a imagem android-x86-4.3-20130725.iso, para instalar uma dezena de sistemas operacionais virtuais, utilizando o *Oracle VirtualBox Manager* para a instalação e a inicialização das instâncias. A Figura 30 mostra a tela de configuração do aplicativo da Oracle. Através dessa interface, pode-se configurar os atributos de execução de cada instância, como por exemplo, a imagem do sistema operacional a ser utilizada, memória RAM, tipo de rede, entre outros.

Quanto a versão do Android, a 4.3 já está estável e consolidada, ao mesmo tempo que possui um nível de API com recursos interessantes ao protótipo. A utilização de máquinas virtuais rodando o Android x-86 tem algumas vantagens sobre os emuladores disponibilizados com o Eclipse. Umas dessas vantagens é a não necessidade de emular completamente a arquitetura ARM, o que torna todo o sistema mais rápido. Além disso, sendo o Android x86 uma imagem completa de um sistema *Android*, é possível ter acesso a recursos não disponíveis num emulador, já que o *debugger* do Eclipse enxerga a máquina virtual não como um emulador mas como um dispositivo móvel. Com o auxílio do *Oracle VirtualBox Manager*, é possível inicializar, a partir do Mac OSX, diversos emuladores do *Android* ao mesmo tempo e permitir que as instâncias tenham acesso direto à rede, sem a necessidade de utilização de *port forwarding*. Assim, as instâncias rodando *Android* têm acesso direto umas às outras via seus IP's.

O diagrama de classes do UDeal está apresentado na Figura 31, mostrando uma visão geral dos agentes do protótipo. A classe principal do projeto é a *MainActivity*, que inicializa a aplicação e implementa a tela inicial, menus, *Thread* principal , entre outras tarefas. Além disso, a *MainActivity* é responsável por instanciar todos os agentes do protótipo, apresentados no diagrama. Cada agente é baseado na classe abstrata Agente do UDeal, que estende a classe padrão

[1] http://www.android-x86.org

Figura 30: Tela de configuração do Oracle VirtualBox Manager



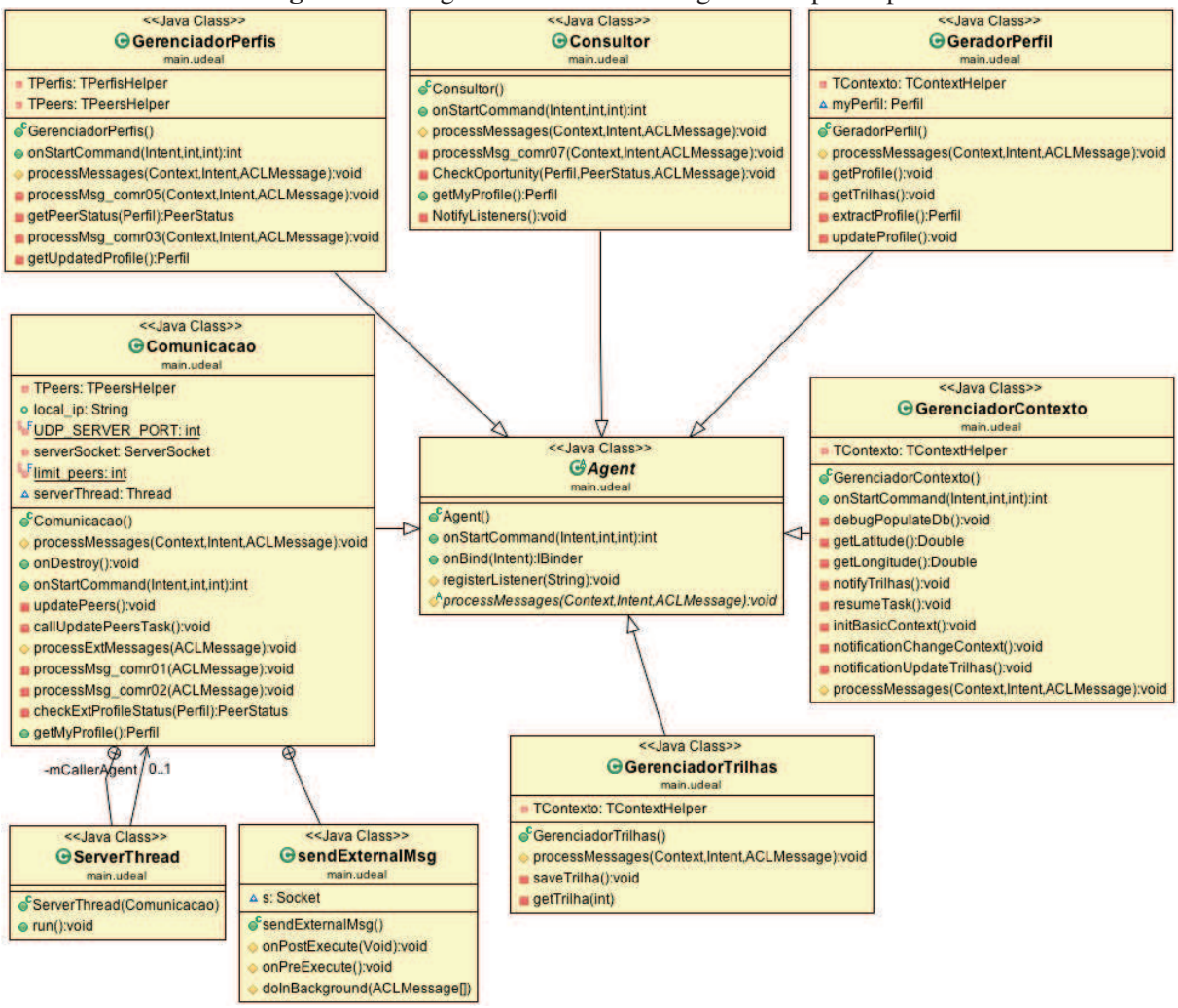
Fonte: Elaborado pelo autor

Services. Portanto, ao inicializar, a aplicação instancia um serviço local para cada agente, ou seja, todos os serviços rodam dentro da *Thread* principal da aplicação. A classe Agente, por sua vez, possui algumas rotinas em comum para os agentes, como a implementação de um método para o registro de *Listeners* e o processamento de mensagens recebidas.

As mensagens, por sua vez, são enviadas através do registro de *intents* via classe estática LocalBroadcastManager, utilizando o método sendBroadcastSync. A classe recebe a mensagem e aciona o *listener* de forma síncrona, ou seja, o agente que enviou a mensagem aguarda até que o listener responda (caso o agente não precise aguardar uma resposta imediata, como no caso de uma simples notificação, é possível utilizar o método assíncrono sendBroadcast). Cada agente, ao ser inicializado, faz o registro de um l*istener* para cada mensagem especificada no modelo. Esses *listeners* são registrados pelo Broadcast Manager utilizando a classe BroadcastReceiver e, através da sobrecarga do método onReceive, é possível capturar mensagens, processá-las e, se necessário, enviar diretamente ao remetente uma resposta ou ainda criar uma nova mensagem para ser enviada a um outro *listener* . Esse método é muito eficiente e permite enviar aos *listeners* não apenas notificações, mas qualquer objeto serializado. Quando um agente necessita enviar uma mensagem para outro agente, o protótipo primeiro instancia um objeto da classe ACLMessage, que está de acordo com a especificação FIPA-ACL. Os campos principais utilizados são performative, sender, receiver, content e ontologia. No campo performative, é possível utilizar, entre outras, as operações request e subscribe; sender e receiver representam os agentes envolvidos e alternativamente é possível utilizar o campo reply-to; o campo content, no protótipo, server para o envio de objetos serializados, como um perfil de usuário; o campo ontologia, no protótipo, contém o nome da classe que representa o campo content, assim as

informações podem ser desserializadas no agente destinatário. Após a criação de um objeto do tipo ACLMessage, esse é serializado e enviado como parâmetro ao Broadcast Manager, depois desserializado e processado pelo *receiver*.

Figura 31: Diagrama de classes dos agentes do protótipo



Fonte: Elaborado pelo autor

Contudo, desde o *Android* 3, por motivos de desempenho, não é possível executar tarefas que utilizem a rede dentro da *Thread* principal (*UI Thread*). Por exemplo, se existe a necessidade de chamar um *webservice* para popular uma lista de itens na tela, a consulta ao serviço REST deverá ser executada em uma nova *Thread*. Isto evita que a interface fique bloqueada pela execução desta operação, que pode se tornar longa, forçando a mesma a rodar em *background*. O *Android* possui uma classe nativa para implementação de *Threads* chamada *AsyncTask*, que permite uma fácil manipulação de tarefas em *background*, como consultas a bancos de dados, utilização de *webservices*, *sockets* e outras operações. No protótipo, foi utilizado *AsyncTask* dentro do Agente de Comunicação, para a troca de mensagens entre diferentes instâncias do protótipo, sobre a rede.

De forma análoga às mensagens entre agentes de mesma instância do protótipo, as mensa-

gens externas são compatíveis com FIPA-ACL, utilizando a mesma classe ACLMessage. Após a criação do objeto mensagem, esse, ao invés de ser enviado ao Broadcast Manager, é serializado e anexado a um datagrama UDP, que por sua vez é enviado via socket, para todas as instâncias na rede identificada pelo protótipo. O protocolo UDP é simples, não tem garantia de entrega e, dessa forma, não exige uma conexão longa entre os dispositivos. O controle do processo é feito pela aplicação. Além disso, o UDP permite, com um único socket, executar Multicast. Para a recepção do datagrama UDP com a mensagem, foi implementado um listener no agente Comunicação, utilizando uma Thread Java dedicada. Esse *listener*, ao receber o datagrama UDP, desserializa a mensagem e a envia para ser processada pelo agente Comunicação.

5.2 Avaliação

A avaliação do modelo foi feita a partir da simulação de um cenário semelhante ao proposto pelo MUCS e um cenário de negociação paralela, introduzido nesta dissertação. O design baseado em cenários e casos de uso é essencial durante a fase de especificação do sistema na metodologia Prometheus. A partir dos cenários, é possível deduzir metas (*goals*) e papeis (*roles*) responsáveis por cada meta. Com base nos cenários, metas e papéis, é possível modelar agentes para exercer determinado papel. Portanto, a avaliação de sistemas baseados em agentes através da simulação ou execução real de cenários é natural para sistemas desenvolvidos usando esta metodologia, pois os agentes foram inicialmente modelados a partir de cenários e casos de uso. De fato, a avaliação através de cenários é um importante método para a verificação do comportamento de sistemas sensíveis ao contexto. Avaliações baseadas em cenários são comuns na área de computação móvel e ubíqua, sendo que todos os trabalhos relacionados utilizam a execução de cenários como parte importante de sua avaliação.

Para a avaliação do U-Deal, os critérios utilizados foram a aderência (ser capaz de executar alguns cenários de comércio ubíquo com respostas compatíveis com outros modelos) e desempenho (resolução de oportunidades em tempo razoável). As simulações foram realizadas utilizando instâncias executando Android 4.3, através do Oracle VirtualBox. As máquinas utilizadas foram um Mac Mini com 16GB de RAM e processador Intel Core i7 2.6GHz quad-core e um Mac Book com 2GB RAM e processador Intel Core2 Duo 1.8GHz. Os dois cenários de simulação elaborados são apresentados nas seções 5.2.1 e 5.2.2.

5.2.1 Cenário 1: Relacionamento entre cliente e fornecedor em feira de negócios

Este cenário apresenta o relacionamento entre cliente e fornecedor em uma feira de negócios, a Expomusic. A feira é a maior da América Latina e concentra negócios na área de equipamentos e instrumentos musicais. Os atores são apenas visitantes da feira. A Tabela 6 mostra, passo-a-passo, a execução do caso de uso no sistema. As informações relativas ao cenário são as seguintes:

- Feira: Expomusic 2014 – SP

- Cliente(C): dono de loja de instrumentos raros, visitante da feira, com interesse em adquirir guitarras *vintage*;

- Fornecedor(F): músico profissional, visitante da feira; com interesse em vender equipamentos próprios usados. Entre eles, algumas guitarras.

5.2.2 Cenário 2: Relacionamento entre um cliente e dois fornecedores em Mercado Público

Este cenário apresenta o relacionamento entre um cliente e dois fornecedores em Mercado Público. Na simulação, os usuários chegam em momentos diferentes ao local dos eventos descritos. No caso específico desse, o cliente restringe sua demanda a apenas avaliar oportunidades no Mercado Público, enquanto os fornecedores registraram as ofertas sem restrições. Ou seja, os fornecedores estão periodicamente e em qualquer lugar executando varreduras na rede e avaliando perfis. A Tabela 7 mostra, passo-a-passo, a execução do caso de uso no sistema.

5.2.3 Comentários sobre os Cenários

Para a execução das simulações, foram criadas, no total, 17 instâncias do Android, doze no Mac Mini e cinco no MacBook. Dessa forma, foi possível executar o envio de datagramas UDP através da rede *wi-fi* e monitorar a comunicação entre duas máquinas diferentes. Um único datagrama UDP é suficiente para o transporte do perfil de usuário, sendo que esse datagrama pode carregar uma mensagem de até 65508 bytes. As bases de dados, no Sqlite, foram populadas via terminal, de acordo com o papel que cada instância teria no modelo. Quase todos os passos descritos nas Tabelas 2 e 3 aconteceram automaticamente, excetuando aqueles indicados como manuais, como login na rede e troca de mensagens via Caixas Postal. O protótipo do U-Deal executou ambos os cenários, satisfatoriamente. O envio interno de cada mensagem ACL entre os clientes é feito em poucos milissegundos (mesmo com a transferência de objetos serializados), assim como as comunicações via UDP (com instâncias executando em computadores diferentes na mesma rede *wi-fi*). O principal problema de desempenho na execução do protótipo ocorreu na geração dos perfis dinâmicos, pois o agente Gerador de Perfil deve fazer uma leitura sequencial nas Trilhas e agrupar as informações de acordo com o modelo do Perfil dinâmico proposto. O ponto positivo dessa estratégia é que esse processo pode ser realizado assincronamente, executado por eventos do sistema ou do protótipo, fazendo com que o Consultor de oportunidades tenha muito mais agilidade para executar o *match*. Dessa forma, o agente Consultor pode trabalhar apenas com os perfis a serem analisados, sem a necessidade de, no momento do *match*, executar uma busca nas trilhas.

Tabela 6: Cenário 1: Relacionamento cliente e fornecedor em feira de negócios

C	Insere demanda no sistema com restritor de tempo: 17 a 21 de setembro, e das 13h às 19h.; sem restritor de localização; escalabilidade localizada vai ser atingida pela cobertura wifi	Agente Gerenciador de Perfil, CRUD de uma Demanda
F	Insere oferta no sistema. Oferta não possui restrições de contexto.	Agente Gerenciador de Perfil, CRUD de uma Oferta.
C	Login na rede wifi publica da feira e inicializa Udeal	Agente Comunicação: scan rede; identifica 3 peers e solicita perfil via UDP
C		Agente Comunicação recebe 2 mensagens via UDP com os perfis de 2 peers e verifica se esses perfis ja estavam registrados; Agente Consultor não identifica oportunidade e não executa notificação nem salva trilhas.
F	Login na rede wifi publica da feira. Udeal já estava inicializado	Agente Comunicação identifica 4 peers na rede e envia solicitação de perfil
C		Agente de Comunicação recebe solicitação de perfil via UDP, feita pelo fornecedor; solicita internamente ao Agente Gerenciador de Perfil seu próprio perfil atualizado e repassa ao fornecedor via UDP.
F		Agente Comunicação recebe perfil do cliente via UDP;
F		Agente Consultor usa o perfil local e o perfil externo para executar um match, com auxilio do agente de Contexto, para verificar a restrição da demanda; Agente Gerenciador de Perfis registra o perfil externo; Agente de Contexto cria um subcontexto e salva registro nas trilhas; Agente consultor não dispara notificação.
F		Agente Gerador de Perfil extrai das trilhas e do perfil estático do utilizador um novo perfil dinâmico, incluindo o match do passo anterior; Agente Gerenciador de Perfil registra novo perfil atualizado.
C		Agente Comunicação: scan rede; identifica 1 peer e solicita perfil via UDP
F		Agente Comunicação recebe solicitação de perfil via UDP, feita pelo cliente. Agente de Comunicação solicita internamente ao agente Gerenciador de Perfil seu próprio perfil atualizado e repassa ao cliente via UDP.
C		Agente Comunicação recebe 1 mensagens via UDP, com o perfil do fornecedor. Agente de Comunicação verifica se esse perfil estava registrado no sistema e repassa para o Agente Consultor para análise. Agente Consultor realiza o match de uma demanda local com uma oferta externa, com auxílio dos dados do perfil externo, onde já constava uma análise de oportunidade relativa a demanda local; Agente Gerenciado de Contexto analisa se as restrições de contexto da demanda estão válidas. Agente Consultor manda perfil para o Agente Gerenciador de Perfis, a fim de ser registrado no sistema, e solicita ao Gerenciador de Contexto a criação de um contexto de negociação, como um container para processo de negociação. O contexto herda as restrições da demanda. Agente de Contexto dispara a criação de um registro nas trilhas; Agente Consultor dispara uma notificação de oportunidade para o usuário local, através da UI.
C	Cliente verifica notificações na UI e utiliza a caixa postal do U-Deal para enviar mensagem ao fornecedor. Na mensagem, cliente agenda um encontro com fornecedor na cafeteria da feira, em 30 min para falar sobre o seu interesse.	Agente Comunicação coordena o envio e recebimento de mensagens UDP e a UI da caixa postal.
F	Recebe mensagem enviada pelo cliente, através da UI da caixa postal do agente Comunicação, a respeito de uma reunião na cafeteria da feira. Fornecedor responde a mensagem	Agente Comunicação coordena o envio e recebimento de mensagens UDP e a UI da Caixa Postal.
C/F	Após reunião e conclusão do negócio, cliente e fornecedor marcam oportunidade como concluída com sucesso.	CRUD de Ofertas e Demandas do Agente de Perfis é acionado através da UI. Ao finalizar a oportunidade, o Agente de Contexto é acionado para disparar a geração de registro em Trilhas do evento e desalocar contexto de negociação.

Fonte: Elaborado pelo autor

Tabela 7: Cenário 2: Relacionamento entre um cliente e dois fornecedores em Mercado Público

C	Insere demanda no sistema com restritor de localização, com a posição GPS aproximada do Mercado	Agente Gerenciador de Perfil, CRUD de uma Demanda
F1	Insere oferta no sistema. Oferta sem restrições de contexto.	Agente Gerenciador de Perfil, CRUD de uma Oferta.
F2	Insere oferta no sistema. Oferta sem restrições de contexto.	Agente Gerenciador de Perfil, CRUD de uma Oferta.
C	Login na rede wifi publica do Mercado; U-Deal já está rodando	Agente Comunicação: scan rede; identifica 15 peers e solicita perfil via UDP
C		Agente Comunicação recebe 5 mensagens via UDP com os perfis de 5 peers e verifica se esses perfis ja estavam registrados; Agente Consultor não identifica oportunidade e não executa notificação nem salva Trilhas.
F1/F2	Login na rede wifi publica do Mercado. U-Deal já estava inicializado	Agente Comunicação identifica 17 peers na rede e envia solicitação de perfil
C		Agente de Comunicação recebe solicitações de perfil via UDP, feita pelos F1 e F2; solicita internamente ao agente Gerenciador de Perfil seu próprio perfil atualizado e responde via UDP.
F1/F2		Agente Comunicação recebe perfil do cliente via UDP;
		Agente Consultor usa o perfil local e o perfil externo para executar um match, com auxilio do agente de Contexto, para verificar a restrição da Demanda; Agente Gerenciador de Perfis registra o perfil externo; Agente de Contexto cria um subcontexto e salva registro nas trilhas; Agente Consultor não dispara notificação.
F1/F2		Agente Gerador de Perfil extrai das trilhas e do perfil estático do utilizador um novo perfil dinâmico, incluindo o match do passo anterior; Agente Gerenciador de Perfil registra novo perfil atualizado.
C		Agente Comunicação: scan rede; identifica 17 peer e solicita perfil via UDP
F1/F2		Agente Comunicação recebe solicitação de perfil via UDP, feita pelo cliente. Agente de Comunicação solicita internamente ao Agente Gerenciador de Perfil seu próprio perfil atualizado e repassa ao cliente via UDP.
C		Agente Comunicação recebe 2 mensagens via UDP, com o perfil do fornecedor. Agente de Comunicação verifica se esse perfil estava registrado no sistema e repassa para o Agente Consultor para análise. Agente Consultor realiza o match de uma Demanda local com uma Oferta externa, com auxílio dos dados do perfil externo, onde já constava uma análise de oportunidade relativa a demanda local; Agente Gerenciado de contexto analisa se as restrições de contexto da Demanda estão válidas. Agente Consultor manda perfil para o Agente Gerenciador de Perfis, a fim de ser registrado no sistema, e solicita ao Gerenciador de Contexto a criação de um contexto de negociação, como um container para processo de negociação. O contexto herda as restrições da demanda. Agente de Contexto dispara a criação de um registro nas Trilhas; Agente Consultor dispara uma notificação de oportunidade para o usuário local, através da UI.
C	Cliente verifica notificações na UI e utiliza a caixa postal do U-Deal para enviar mensagem ao F2 e descarta F1; Cliente utiliza telefone do perfil do F2 para realizar chamada e pedir mais informações sobre os produtos.	UI do Gerenciador de Perfis é acionada para Verificar os dados do F2.
F1		Trilha e perfis do usuário terão registrado esse evento como oportunidade incompleta
C/F2	Após conclusão do negócio, cliente e fornecedor marcam oportunidade como concluída com sucesso.	CRUD de ofertas e demandas do Agente de Perfis é acionado através da UI. Ao finalizar a oportunidade, o agente de Contexto é acionado para disparar a geração de registro em trilhas do evento e desalocar contexto de negociação.

Fonte: Elaborado pelo autor

6 CONSIDERAÇÕES FINAIS

Este capítulo apresenta uma breve conclusão sobre o trabalho e uma discussão sobre as contribuições do U-Deal, comparando-o com outros modelos. Por fim, destaca questões de pesquisa em aberto, para futuros trabalhos.

6.1 Conclusões

Existe ainda um potencial não explorado para aplicações *peer-to-peer* nas mais variadas áreas, em um contexto tecnológico em que os dispositivos móveis estão mais poderosos e é cada vez maior a disponibilidade de redes sem fio. A utilização de uma arquitetura *peer-to-peer* para uma aplicação não implica na não utilização de serviços na nuvem. Pelo contrário, estratégias de computação móvel *peer-to-peer* são adequadas para determinados tipos de tarefas, principalmente aquelas ligadas à sensibilidade ao contexto e que exijam apenas uma escalabilidade local. Além disso, é possível que uma aplicação distribuída tenha nodos computacionais tanto na nuvem quanto em uma rede local *ad hoc*. No caso do U-Deal, ele poderia ser integrado a algum serviço na nuvem para backup ou *dump* das trilhas mais antigas ou ainda a algum serviço de sincronização entre dispositivos de um mesmo usuário.

6.2 Contribuições

De acordo com a Tabela 8, as contribuições do modelo U-Deal, em relação a outros trabalhos de comércio ubíquo são:

- criação de um modelo extensível para negociação entre agentes. Extensível porque o modelo permite a implementação de diferentes estratégias de negociação, sem a necessidade de modificação nos agentes de infraestrutura como trilhas e contexto;
- modelo é sensível a trilhas, não apenas a contexto e perfil de usuário;
- as trilhas são descentralizadas, ou seja, cada usuário tem suas trilhas armazenadas em seu dispositivo, mas informações sobre suas atividades podem estar distribuídas nas trilhas de outros usuários, com os quais travaram contato;
- possui mecanismo para definição de níveis de confiança em uma outra entidade baseado no perfil do usuário (através da configuração de níveis de privacidade) e no histórico de relacionamento entre as entidades, tendo como base suas trilhas;
- utiliza perfil dinâmico.

Dentre as contribuições deste trabalho, é possível destacar como principal contribuição a apresentação de um modelo sensível ao contexto com trilhas descentralizadas, onde cada usuário mantém suas próprias trilhas localmente. Isso implica em que a sensibilidade ao contexto e

Tabela 8: Comparação dos Trabalhos relacionados

Atributos / Trabalhos	GTTracker	Jiazao Lin et al.	MUCS	Sanchez-Pi;Molina	PAM	U-Deal
Modelo de Dados	JSON	CUB-ONT (Ontologia)	Árvore de Categorias/Modelo Relacional/XML	Ontologia	Modelo Relacional	Modelo Relacional
Utiliza Trilhas	Não	Sim	Sim	Não	Não	Sim
Perfil de Entidade	Estático	Estático	Estático	Estático	Dinâmico	Dinâmico
Modelo de Sistema	P2P	Cliente-Servidor	Cliente-Servidor	Cliente-Servidor e P2P	Cliente-Servidor e P2P	P2P
Privacidade e Segurança	Controle Acesso	Controle Acesso	Controle Acesso	Controle Acesso	Controle Acesso	*trust* baseado no histórico de relacionamento

Fonte: Elaborado pelo autor

às trilhas seja gerenciada no dispositivo do usuário e não a partir do cálculo executado em um servidor, o que favorece a Escalabilidade Localizada. Além disso, a arquitetura *peer-to-peer* exige menos infraestrutura, sendo que a perda de *peers* ou de comunicação na rede não afeta o serviço de outros *peers*. A arquitetura P2P mostra-se mais adequada aos conceitos de Computação Ubíqua do que o modelo cliente-servidor, de acordo com as características da Computação Ubíqua apresentadas por Satyanarayanan (2001).

6.3 Trabalhos Futuros

Por fim, a discussão teórica e a execução do protótipo levantam as seguintes questões de pesquisa para trabalhos futuros:

- o conteúdo das mensagens poderia ser representado por ontologias de Perfil, Trilhas ou Negócios. Contudo, as API's para interpretar ontologias, como a OWL-API utilizada no JFact [22], apenas são suportadas pelo Java *Standard* e não pelo Android;
- o Agente Gerador de Perfil poderia ser construído utilizando um Raciocinador semântico, para a inferência nas Trilhas, que, juntamente com uma base de regras, poderia gerar perfis dinâmicos mais completos para a análise do Consultor. Ainda, o próprio Agente Consultor de oportunidades poderia ser implementado com um raciocinador semântico, assim manipulando dados mais complexos e com uma maior capacidade de extrair informação dos perfis.

REFERÊNCIAS

BARBOSA, J. L. V.; HAHN, R. M.; BARBOSA, D. N. F.; SACCOL, A. I. d. C. Z. A Ubiquitous Learning Model Focused on Learner Interaction. **Int. J. Learn. Technol.**, Geneva, Switzerland, v. 6, n. 1, p. 62–83, May 2011.

BARKHUUS, L.; BARKHUUS, L.; DEY, A.; DEY, A. Is Context-Aware Computing Taking Control Away from the User? Three Levels of Interactivity Examined. In: IN PROCEEDINGS OF UBICOMP 2003, 2003. **Anais...** Springer, 2003. p. 149–156.

CAZAROTTO, P. H. **GTTRACKER**: serviço para identificação de oportunidades comerciais através de dispositivos móveis. 2013. Dissertação (Mestrado em Ciência da Computação) — Universidade do Vale do Rio dos Sinos - UNISINOS, 2013.

COEN, M.; PHILLIPS, B.; WARSHAWSKY, N.; WEISMAN, L.; PETERS, S.; FININ, P. Meeting the Computational Needs of Intelligent Environments: the metaglue system. In: IN PROCEEDINGS OF MANSE'99, 1999. **Anais...** Springer-Verlag, 1999. p. 201–212.

COMER, D.; YAVATKAR, R. FLOWS: performance guarantees in best effort delivery systems. In: INFOCOM '89. PROCEEDINGS OF THE EIGHTH ANNUAL JOINT CONFERENCE OF THE IEEE COMPUTER AND COMMUNICATIONS SOCIETIES. TECHNOLOGY: EMERGING OR CONVERGING, IEEE, 1989. **Anais...** IEEE Computer and Communications Societies, 1989. p. 100–109 vol.1.

COSTA, C. A. da; 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, Jan. 2008.

DEY, A. K. Understanding and Using Context. **Personal Ubiquitous Comput.**, London, UK, v. 5, n. 1, p. 4–7, Jan. 2001.

DEY, A. K.; ABOWD, G. D.; SALBER, D. A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-aware Applications. **Hum.-Comput. Interact.**, Hillsdale, NJ, USA, v. 16, n. 2, p. 97–166, Dec. 2001.

FIPA-ACL. **Foundation of Intelligent Physical Agents-Agent Communication Language**. `http://www.fipa.org/specs/fipa00061/`. acesso em Junho 2013.

FRANCO, L. K.; ROSA, J. H.; BARBOSA, J. L. V.; COSTA, C. A.; YAMIN, A. C. MUCS: a model for ubiquitous commerce support. **Electron. Commer. Rec. Appl.**, Amsterdam, The Netherlands, v. 10, n. 2, p. 237–246, Mar. 2011.

GALANXHI-JANAQI, H.; NAH, F. F.-H. U-commerce: emerging trends and research issues. In: INDUSTRIAL MANAGEMENT AND DATA SYSTEMS, 2004. **Anais...** Emerald Group Publishing Limited, 2004. v. 104, n. 9, p. 744–755.

GANTI, R.; YE, F.; LEI, H. Mobile crowdsensing: current state and future challenges. **Communications Magazine, IEEE**, New York, NY, USA, v. 49, n. 11, p. 32–39, November 2011.

GERSHMAN, A. Ubiquitous Commerce - Always On, Always Aware, Always Pro-active. In: INDUSTRIAL MANAGEMENT AND DATA SYSTEMS, 2004. **Anais...** Emerald Group Publishing Limited, 2004. v. 104, n. 9, p. 744–755.

HENRICKSEN, K. **A framework for context-aware pervasive computing applications**. Queensland, Australia: University of Queensland, 2003.

HUHNS, M. N.; STEPHENS, L. M. Multiagent Systems and Societies of Agents. In: MULTIAGENT SYSTEMS: A MODERN APPROACH TO DISTRIBUTED ARTIFICIAL INTELLIGENCE, 1999. **Anais...** MIT Press, 1999. p. 79–120.

JUNGLAS, I. A.; WATSON, R. T. U-Commerce: a conceptual extension of e-commerce and m-commerce. In: ICIS, 2003. **Anais...** Association for Information Systems, 2003. p. 667–677.

LEMPEREUR, A.; SEBENIUS, J.; DUZERT, Y. **Manual de negociações complexas**. Rio de Janeiro, Brazil: FGV, 2009.

LIN, J.; LI, X.; YANG, Y.; LIU, L.; GUO, W.; LI, X.; LI, L. A Context-aware Recommender System for M-commerce Applications. In: INTERNATIONAL CONFERENCE ON ACTIVE MEDIA TECHNOLOGY, 7., 2011, Berlin, Heidelberg. **Proceedings...** Springer-Verlag, 2011. p. 217–228. (AMT'11).

LIN, K.-J.; YU, T.; SHIH, C.-Y. The Design of A Personal and Intelligent Pervasive-Commerce System Architecture. In: MOBILE COMMERCE AND SERVICES, 2005. WMCS '05. THE SECOND IEEE INTERNATIONAL WORKSHOP ON, 2005, New York, NY, USA. **Anais...** IEEE Computer and Communications Societies, 2005. p. 163–173.

LIU, Q. U-commerce Research: a literature review and classification. **Int. J. Ad Hoc Ubiquitous Comput.**, Inderscience Publishers, Geneva, SWITZERLAND, v. 12, n. 3, p. 177–187, Mar. 2013.

MARTINS, C.; ROSA, J. a.; FRANCO, L.; BARBOSA, J.; BEZERRA, E. Towards a model to explore business opportunities in trail-aware environments. In: BRAZILIAN SYMPOSIUM ON MULTIMEDIA AND THE WEB, 18., 2012, New York, NY, USA. **Proceedings...** ACM, 2012. p. 143–150. (WebMedia '12).

MCDONALD, D. Ubiquitous Recommendation Systems. **IEEE Computer**, New York, NY, USA, p. 111–112, 2003.

NASH, J. The Bargain Problem. **Econometrica**, New York, NY, USA, n. 18, p. 155–162, 1950.

OLIVEIRA, J. M. d. **Um Modelo Multi-agente Descentralizado para Ambientes de Educação Ubíqua**. 2010. Dissertação (Mestrado em Ciência da Computação) — Universidade do Vale do Rio dos Sinos - UNISINOS, 2010.

PADGHAM, L.; WINIKOFF, M. Prometheus: a methodology for developing intelligent agents. In: GIUNCHIGLIA, F.; ODELL, J.; WEISS, G. (Ed.). **Agent-Oriented Software Engineering III**. Berlin, Heidelberg: Springer-Verlag, 2003. p. 174–185. (Lecture Notes in Computer Science, v. 2585).

PERERA, C.; ZASLAVSKY, A. B.; CHRISTEN, P.; GEORGAKOPOULOS, D. Context Aware Computing for The Internet of Things: a survey. **IEEE Communications Surveys and Tutorials**, New York, NY, USA, v. 16, n. 1, p. 414–454, 2014.

ROUSSOS, G.; KOUROUTHANASIS, P.; SPINELLIS, D.; GRYAZIN, E.; PRYZBLISKI, M.; KALPOGIANNIS, G.; GIAGLIS, G. Systems architecture for pervasive retail. In: ACM SYMPOSIUM ON APPLIED COMPUTING, 2003. **Anais...** ACM Press, 2003. p. 631–636.

SALBER, D.; DEY, A. K.; ABOWD, G. D. **Ubiquitous Computing**: defining an hci research - agenda for an emerging interaction paradigm. 1998.

SANCHEZ-PI, N.; MOLINA, J. A multi-agent platform for the provisioning of U-commerce services. In: FUZZY INFORMATION PROCESSING SOCIETY, 2009. NAFIPS 2009. ANNUAL MEETING OF THE NORTH AMERICAN, 2009, New York, NY, USA. **Anais...** IEEE Computer and Communications Societies, 2009. p. 1–6.

SATYANARAYANAN, M. Pervasive Computing: visions and challenges. In: IEEE PERSONAL COMMUNICATION, 2001, New York, NY, USA. **Anais...** IEEE Computer and Communications Societies, 2001.

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

SCHILIT, B.; ADAMS, N.; WANT, R. Context-Aware Computing Applications. In: PROCEEDINGS OF THE WORKSHOP ON MOBILE COMPUTING SYSTEMS AND APPLICATIONS, 1994. **Anais...** IEEE Computer Society, 1994. p. 85–90.

SEGATTO, W.; HERZER, E.; MAZZOTTI, C. L.; BITTENCOURT, J. a. R.; BARBOSA, J. Mobio threat: a mobile game based on the integration of wireless technologies. **Comput. Entertain.**, New York, NY, USA, v. 6, n. 3, p. 39:1–39:14, Nov. 2008.

SHENG, H.; NAH, F. F.-H.; SIAU, K. An Experimental Study on Ubiquitous Commerce Adoption: inpact of personalization and privacy concerns. **Journal of the Association for Information Sistem - JAIS**, Illinois, USA, v. 9, n. special issue, p. 344–376, 2008.

SILVA, J.; ROSA, J. a.; BARBOSA, J.; BARBOSA, D. N. F.; PALAZZO, L. A. M. Content Distribution in Trial-aware Environments. In: XV BRAZILIAN SYMPOSIUM ON MULTIMEDIA AND THE WEB, 2009, New York, NY, USA. **Proceedings...** ACM, 2009. p. 15:1–15:8. (WebMedia '09).

TAVARES, J. a.; BARBOSA, J.; COSTA, C.; YAMIN, A.; REAL, R. Hefestos: a model for ubiquitous accessibility support. In: INTERNATIONAL CONFERENCE ON PERVASIVE TECHNOLOGIES RELATED TO ASSISTIVE ENVIRONMENTS, 5., 2012, New York, NY, USA. **Proceedings...** ACM, 2012. p. 27:1–27:8. (PETRA '12).

WEISER, M. The Computer for the Twenty-First Century. **Scientific American**, New York, NY, USA, v. 265(3), p. 94–104, 1991.

WEISER, M.; BROWN, J. S.; DENNING, P. J.; METCALFE, R. M. The coming age of calm technology. In: DENNING, P. J.; METCALFE, R. M. (Ed.). **Beyond Calculation**: the next fifty years of computing. New York, NY, USA: Copernicus, 1998. p. 75–85.

WOOLDRIDGE, M.; JENNINGS, N. R. Intelligent agents: theory and practice. **The Knowledge Engineering Review**, Cambridge, England, v. 10, p. 115–152, 5 1995.

XIAO, Y.; SIMOENS, P.; PILLAI, P.; HA, K.; SATYANARAYANAN, M. Lowering the barriers to large-scale mobile crowdsensing. In: WORKSHOP ON MOBILE COMPUTING SYSTEMS AND APPLICATIONS, 14., 2013, New York, NY, USA. **Proceedings...** ACM, 2013. p. 9:1–9:6. (HotMobile '13).

YOUSSEF, A. E. Towards Pervasive Computing Environments with Cloud Services. **International Journal of Ad hoc, Sensor & Ubiquitous Computing (IJASUC)**, Chennai, Tamil Nadu, India, v. 4, p. 1–9, 2013.

ZHANG, L.; LIU, Q.; ; LI, a. Ubiquitous Commerce: theories, technologies, and applications. In: JOURNAL OF NETWORKS, 2009, Road Town, Tortola, British Virgin Islands. **Anais...** Academy Publisher, 2009. v. 4, n. 4, p. 271–278.