Vivian Cremer Kalempa

Especificando Privacidade em Ambientes de Computação Ubíqua

Florianópolis – SC

Fevereiro / 2009

UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Vivian Cremer Kalempa

Especificando Privacidade em Ambientes de Computação Ubíqua

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação

Prof. Dr. João Bosco Mangueira Sobral

Florianópolis, fevereiro de 2009

ESPECIFICANDO PRIVACIDADE EM AMBIENTES DE COMPUTAÇÃO UBÍQUA

Vivian Cremer Kalempa

Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação na Área de Concentração Sistemas de Computação e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Prof. Dr. Frank Augusto Siqueira
Coordenador do PPGCC

Banca Examinadora

Prof. Dr. João Bosco Mangueira Sobral
Orientador

Prof. Dr. Guilherme Horta Travassos

Prof. Dr. Antônio Augusto Medeiros Fröhlich

Prof. Dr. Carlos Montez

Dedico a Deus, que sempre iluminou
meu caminho;
aos meus pais,
Carlos Alberto Cremer e
Romilda Teresinha Cremer,
e ao meu marido,
Saulo Eduardo Kalempa,
que sempre me deram amor,
carinho e palavras de incentivo.

Agradecimentos

Agradeço a Deus por ter me dado toda a sabedoria e tudo mais que foi necessário para a minha caminhada e para a realização deste trabalho.

Agradeço aos meus pais pelo incentivo, carinho e amor espontâneos que tanto me fizeram bem no tempo em que estive longe de casa.

Agradeço ao meu marido pela compreensão dos mais de 200 km de distância para a realização deste trabalho e por entender o que significa, para mim, ter um mestrado. Agradeço também pelo apoio, tanto no momento da mudança, quanto nos momentos difíceis que encontrei fora de casa, e que ele sempre com palavras de incentivo me deu forças para continuar.

Agradeço ao professor João Bosco pela orientação e amizade.

Agradeço ao PPGCC - Programa de Pós-Graduação em Ciência da Computação e à UFSC - Universidade Federal de Santa Catarina por disponibilizarem o curso e a estrutura para o desenvolvimento do trabalho.

Agradeço ao Jeferson e ao Lucas pela amizade e contribuições para o meu trabalho. Agradeço também a Pâmela e a Rebeca pelos momentos em que passamos juntas na casa amarela.

Agradeço a professora Daniela Barreiro Claro pelo apoio, incentivo, e por acreditar na minha capacidade.

Agradeço ao Rodrigo Campiolo pelas sugestões e críticas sobre a minha dissertação, o que contribuiu para o resultado final obtido.

Agradeço a atenção dos demais professores do curso, que através das disciplinas ministradas, tiveram influência no trabalho realizado.

"Os que esperam no Senhor renovam as suas forças,
sobem com asas como águias,
correm e não se cansam,
caminham e não se fatigam."
Isaías 40:31

Resumo

A computação ubíqua provê ambientes com serviços e dispositivos interconectados que promovem a integração de infra-estrutura digital na vida das pessoas. Porém, por haver muitos empecilhos técnicos que impedem que a computação ubíqua se torne realidade, o foco da pesquisa atual acaba voltando-se, na maioria dos casos, para os assuntos técnicos como, por exemplo, como conectar novos dispositivos e construir aplicações úteis para melhorar a funcionalidade desses ambientes. Contudo, assuntos como segurança e privacidade ainda são pouco tratados. Além disso, nesses ambientes torna-se difícil a separação entre a segurança física e a segurança digital. Assim, fica claro que o paradigma da computação ubíqua introduz novas vulnerabilidades e exposições aos usuários dos seus ambientes, mostrando que as tecnologias, políticas e leis existentes não estão adequadas para lidar com essas novas situações. Nesta dissertação, os desafios em garantir privacidade em ambientes de computação ubíqua são explorados. Além disso, um metamodelo que endereça alguns destes desafios é descrito, apresentado e simulado na ferramenta Opnet.

Palavras-chave: Computação Ubíqua. Privacidade. Simulação.

Abstract

The ubiquitous computing provides environments with services and devices interconnected that promote integration of digital infrastructure in the people's life. However, once there are many technical difficulties that do not allow ubiquitous computing come true, the focus of the existing researches, in most of the cases, goes into technical issues, for example, how to connect new devices and to build useful applications to improve the functionality of those environments. Nevertheless, issues like security and privacy, are not always approached. Besides, in those environments the distinction of physical and digital security becomes difficult. Thus, the paradigm of the ubiquitous computing introduces new vulnerabilities and exposures to the users of its environments, showing that the technologies, policies and existing laws are not adapted to work with those new situations. In this dissertation, the challenges on guaranteeing privacy in environments of ubiquitous computing are explored. In addiction, a metamodel that addresses some of these challenges it is described, presented and simulated in the Opnet tool.

Keywords: Ubiquitous computing. Privacy. Simulation.

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

Este trabalho versa sobre modelagem de sistemas na área de computação ubíqua. Mais precisamente, contém um estudo sobre a questão da privacidade, como forma de estender o trabalho em (CAMPIOLO, 2005), o qual trata da formalização de um metamodelo a ser utilizado como base para a construção de sistemas ubíquos. A extensão proposta aborda a questão da privacidade no nível de usuário de um sistema.

A computação ubíqua promete um mundo onde dispositivos computacionais embutidos no ambiente têm a capacidade de conhecer o contexto das atividades dos usuários e prover serviços baseados nesse contexto conhecido (WEISER, 1991). Contudo, as mesmas características que fazem os ambientes ubíquos convenientes e poderosos, os fazem também vulneráveis para novos tipos de ataque à segurança e privacidade.

Problemas de segurança não são tratados neste trabalho, enquanto questões de privacidade, dizem respeito à coleta de dados nesses ambientes, sobre os usuários, e podem ser compartilhados sem o consentimento do mesmo.

Exemplos de informações que podem ser consideradas confidenciais são: idade, pressão sanguínea, doença específica ou informação sobre os interesses, objetivos e planos do usuário, as quais podem ser exploradas por intrusos ou até mesmo por gerenciadores de sistemas interessados em localizar ou espiar eletronicamente determinados usuários. Dessa forma, o sistema inteiro se torna um sistema de vigilância distribuído que pode capturar muita informação sobre os usuários. Logo, os mecanismos de segurança tradicionais podem não prover garantias adequadas para lidar com essas novas situações.

Alguns requisitos devem ser levados em consideração para que um ambiente de computação ubíqua seja considerado confiável (FAHRMAIR; SITOU; SPANFELNER, 2005):

(1) Proteção contra abuso: deve existir uma possibilidade do usuário expressar condições e restrições para o uso de dados de contexto. Isto inclui a identificação de serviços;

(2) Identificação de conjunto de dados clandestinos: se um serviço puder evitar a proteção à privacidade, deve haver uma possibilidade em identificar esse serviço. Este item não será abordado com mais detalhes nessa dissertação;

(3) Leis: sob certas circunstâncias as regras podem prover uma segurança adicional;

(4) Facilidade de uso: um sistema de proteção com segurança máxima, mas com uma alta complexidade de controle é inútil.

Além disso, pode-se associar o sucesso da computação ubíqua à vontade dos usuários em aceitar e gerenciar o risco de privacidade em compartilhar suas informações pessoais neste novo mundo digital.

Assim, o gerenciamento de informação pessoal em ambientes inteligentes se caracteriza como um desafio de pesquisa social e técnico muito importante. Os usuários determinam qual informação eles desejam compartilhar, baseados no contexto atual e na identidade dos receptores da sua informação (LEDERER; DEY; MANKOFF, 2002). Ainda, as tecnologias que protegem a privacidade dos usuários devem permitir que eles controlem efetivamente as suas informações, sem carregá-los com avaliações excessivas ou muitas interações com o ambiente, visando esse objetivo. Os usuários também devem ter a possibilidade de compartilhar o mínimo possível de informação para alcançar os seus objetivos, usar pseudônimos e ter anonimato sempre que possível (BERENDT; GUNTHER; SPIEKERMANN, 2005).

Por outro lado, permitir que os usuários sejam anônimos em ambientes inteligentes, melhora a sua privacidade, mas pode tornar os sistemas adaptáveis ao usuário menos efetivos (KOBSA; SCHRECK, 2003). O uso de pseudônimos foi identificado como um método de permitir aos usuários usar os sistemas adaptáveis, mantendo um certo anonimato. Esta idéia busca permitir aos usuários a utilização de pseudônimos, que de certa maneira, irá proteger a sua privacidade de qualquer ataque de mineração de dados, e pode ser usada em conjunto com outras formas de segurança como, por exemplo, a criptografia e zonas de mixagem (BERESFORD, 2005).

1.1 Histórico da Pesquisa sobre Privacidade em Ambientes Ubíquos

Essa seção apresenta alguns dos principais trabalhos desenvolvidos na área da computação ubíqua com foco na privacidade desses ambientes. Além disso, as principais características e problemáticas nesses trabalhos são apresentados.

Em (JIANG; LANDAY, 2002a) é apresentado o *Modelo de Espaço de Informação*, o qual organiza informações, recursos e serviços em torno de fatores contextuais, relevantes à privacidade do usuário. Os espaços de informação são delimitados e possuem responsáveis que determinam suas permissões. Dessa forma, os espaços de informação fornecem informações, recursos, serviços e gerenciamento de autorizações em sistemas sensíveis ao contexto. Para o controle de privacidade, um espaço de informação age como um gatilho para forçar as permissões definidas pelos responsáveis daquele espaço a serem cumpridas. Este modelo protege a informação sensível por meio do controle da informação pelos responsáveis pelo espaço. Porém, essa confiança nos responsáveis pelo espaço se torna um ponto de possível falha neste modelo, como também o componente de software que processa esse mecanismo pode ser problemático para sistemas de grande escala descentralizado.

Já em (CAMPBELL et al., 2002), o *Mist* é apresentado como uma infra-estrutura de comunicação que preserva a privacidade em ambientes de computação ubíqua. O Mist faz a separação entre localização e identidade, permitindo, assim, que as entidades autorizadas do sistema tenham acesso aos serviços, protegendo a sua privacidade de localização. O Mist consiste em uma hierarquia de roteadores que formam uma única rede que, dessa forma, fornece uma comunicação privada. Além disso, o Mist utiliza criptografia de chave pública e dispõe de um nível customizável de privacidade. Essa customização é útil, porém, é definida pelo usuário e se o usuário escolher uma conexão com um roteador em um nível mais baixo na hierarquia, a sua privacidade também estará menos protegida.

Em (LANGHEINRICH, 2002), o sistema *pawS* (*Privacy Awareness System*) proporciona aos usuários ferramentas que os auxiliam a proteger sua própria privacidade e fazer com que os outros também a respeitem. Esse sistema é baseado em normas sociais e legais ao invés de somente ter a proteção técnica. Em um sistema com pawS, quando um usuário entrar em um ambiente no qual há serviços coletando dados, um

dispositivo de privacidade anuncia as políticas de privacidade de cada serviço no ambiente. O *proxy* de privacidade do usuário confere essas políticas com relação às preferências de privacidade do usuário e, se as políticas estiverem de acordo, os serviços podem coletar informações e os usuários podem utilizar os serviços. Se as políticas não estiverem de acordo, o sistema notifica o usuário que pode optar por não utilizar o serviço em questão ou deixar a área na qual a coleta de informação é feita.

Por fim, em (CHENG; ZHANG; TAN, 2005) é apresentado o PSIUM (*Privacy Sensitive Information Diluting Mechanism*) como um modelo que tenta eliminar o excesso de dados do usuário utilizado pelos provedores de serviço. Para isso, o modelo reduz a utilidade dos dados coletados pelos provedores de serviço sem afetar o serviço provido aos usuários. Um dispositivo com o PSIUM habilitado envia múltiplas mensagens de requisição de serviço baseadas em localização ao provedor de serviço. Todas, menos uma dessas mensagens contém a verdadeira localização. No retorno da informação de serviço o dispositivo sabe utilizar a informação correta e fazê-la disponível ao usuário. O PSIUM mantém falsos dados que parecem verdadeiros, de forma que se torna difícil de distingui-los dos verdadeiros. Esses falsos dados são criados a partir de localizações previamente utilizadas pelo usuário. Dessa forma, o PSIUM tem como ponto forte a prevenção ao abuso da utilização dos dados dos usuários por um provedor de serviço, sem reduzir a qualidade do serviço. Seu ponto fraco é que aumenta o custo da obtenção dos resultados pelo provedor de serviço, já que o número de consultas aumenta.

Analisando os trabalhos existentes, voltados para a privacidade em ambientes de computação ubíqua, pode-se perceber que várias abordagens são utilizadas a fim de preencher essa lacuna. Porém, nenhuma dessas abordagens define formalmente os aspectos dos ambientes ubíquos, principalmente o da privacidade. Além disso, algumas técnicas aparentam ser muito complexas ao tentar manter a privacidade dos usuários dos seus ambientes, ou mesmo ineficientes.

1.2 Objetivos

O objetivo geral dessa dissertação é na especificação da privacidade para ambientes ubíquos. Para isso, o aspecto de privacidade em ambientes de computação

ubíqua é especificado formalmente, caracterizando a extensão ao trabalho em (CAMPIOLO, 2005), usando-se a linguagem do OBJECT Z (DUKE et al., 1991). Além disso, um cenário foi modelado com base no metamodelo, o qual foi simulado, suportando uma métrica (DIAZ et al., 2002) para medir o grau de privacidade.

Os objetivos específicos são:

- permitir que o metamodelo desenvolvido em (CAMPIOLO, 2005) permaneça genérico, porém estendido com a característica de privacidade;
- permitir a utilização desse metamodelo em simulações de ambientes de computação ubíqua;
- desenvolver a especificação de privacidade com base nos conceitos de anonimato (PFITZMANN; KöHNTOPP, 2001), uso de pseudônimos (BERESFORD; STAJANO, 2004), perfil de preferências do usuário (LEDERER; DEY; MANKOFF, 2002) e zonas de mixagem (BERESFORD, 2005);
- simular um cenário de ambiente ubíquo com base na especificação desenvolvida, utilizando-se a ferramenta Opnet[1];
- utilizar a entropia (DIAZ et al., 2002) como métrica para a simulação realizada.

Neste trabalho não será abordado o problema de segurança em computação ubíqua, como técnicas de criptografia ou como evitar ataques. É apresentada uma abordagem baseada em confiança, com a necessidade de que sejam feitas algumas extensões abordando segurança. Ou seja, o objetivo desse trabalho é na privacidade das pessoas, e não na segurança dos sistemas que essas pessoas utilizam. Assim, é importante destacar uma distinção entre segurança e privacidade feita por Saltzer e Schroeder (SALTZER; SCHROEDER, 1975), onde a segurança pode ser entendida como os mecanismos e técnicas que controlam quem pode usar determinado computador, ou modificar as informações armazenadas por ele. E a privacidade é tida como a habilidade de um indivíduo (ou organização) decidir se, quando, e para quem a informação pessoal (ou organizacional) pode ser cedida.

[1] O Opnet é um simulador que permite projetar e estudar redes de comunicação, dispositivos, protocolos e aplicação.

1.3 Publicações

Um trabalho inicial sobre a área de computação ubíqua e uma prévia desse metamodelo foi apresentada em (CAMPIOLO; SOBRAL, 2005) e como contribuição do trabalho em questão, o artigo "*On Modeling for Pervasive Computing Environments*", foi publicado no *The 10th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIN)*, em outubro de 2007. Nesse artigo é apresentado o metamodelo para ambientes de computação ubíqua estendido com a característica de privacidade.

1.4 Estrutura da Dissertação

No que segue, o capítulo 2 apresenta as principais características e desafios da área de computação ubíqua. No capítulo 3, são discutidos os principais aspectos em torno da privacidade em ambientes ubíquos, assim como os principais motivos para a invasão da privacidade e algumas soluções técnicas para proteção à privacidade. No capítulo 4, é apresentado o metamodelo desenvolvido por (CAMPIOLO, 2005), para descrever ambientes ubíquos, o qual foi estendido para atender as características necessárias para garantir privacidade nesses ambientes. No capítulo 5 é apresentado um estudo de caso, que foi desenvolvido utilizando-se o metamodelo proposto e simulado na ferramenta Opnet. No capítulo 6, as conclusões e direções para trabalhos futuros.

2 Computação Ubíqua

Tecnologias, tais como, redes sem fio e dispositivos móveis, vem tomando cada vez espaço no dia-a-dia das pessoas, tanto em espaços públicos, empresas, quanto em suas casas. Seja através dos *handhelds*, telefones celulares, ou até mesmo das câmeras digitais, o fato é que as pessoas estão cada vez mais acostumadas com estes dispositivos para a realização das suas tarefas cotidianas. Um ambiente que contém um, ou vários destes dispositivos embutidos ou móveis, pode ser considerado um ambiente de computação ubíqua.

O objetivo deste capítulo é apresentar um breve histórico sobre a área de computação ubíqua; mostrar os principais impactos que esta tecnologia tem na forma como as pessoas trabalham, aprendem, interagem e se entretem. Além disso, este capítulo aborda as principais características de um ambiente de computação ubíqua e alguns desafios.

2.1 Contextualização

A história da computação ubíqua originou-se em 1987 no Laboratório de Eletrônica e Imagem da Xerox, no Centro de Pesquisa em Palo Alto. A idéia da pesquisa era distribuir dispositivos ubíquos com capacidades de computação pelo ambiente, mas de forma invisível, ou seja, sem que a presença deles fosse notada. Essa nova abordagem de computação, denominada *Ubiquitous Computing* ou, computação ubíqua, era diferente da forma de computação existente na época, que tinha a relação "uma pessoa - um computador"(WEISER; GOLD; BROWN, 1999).

A partir disso, em 1988 foi fundado o Programa de Computação Ubíqua no Laboratório de Ciência da Computação (CSL). O objetivo principal deste programa era estudar o que estava errado com o computador pessoal: muito complexo e difícil

de ser usado; e isolado de outras pessoas e outras atividades.

O projeto de computação ubíqua do Centro de Pesquisa em Palo Alto teve como principal contribuição a criação de uma nova área da computação: a computação ubíqua. Além disso, resultou em muitos artigos, dentre eles, destaca-se o artigo *The Computer for the 21th Century* (WEISER, 1991) de Mark Weiser, onde ele imaginou um mundo saturado de computação para auxiliar as pessoas, sem ser intrusiva.

Além de todas as contribuições de vários pesquisadores, a criação de eventos científicos específicos, tais como o *Ubicomp*[1] da ACM, e o surgimento de revistas específicas, com destaque a *IEEE Pervasive Computing*[2], impulsionaram a evolução desta nova área da Ciência da Computação (CAMPIOLO, 2005).

2.1.1 Definições

Computação ubíqua (também chamada de *pervasive computing* (SATYANARAYANAN, 2001)), de acordo com a visão de Mark Weiser (WEISER, 1991), é a computação que desaparece, que faz parte da vida cotidiana sem ser percebida.

Ainda, segundo Debashis Saha e Amitava Mukherjee (SAHA; MUKHERJEE, 2003), computação ubíqua é a computação que torna a vida cotidiana mais simples por meio de ambientes digitais que são sensíveis, adaptáveis, e pró-ativos às necessidades dos humanos. Muito mais que computação móvel, esta tecnologia mudará a natureza da computação, permitindo que a maioria dos objetos encontrados na vida diária sejam "atentos", interagindo com usuários no mundo físico e virtual.

A computação ubíqua pode ser considerada como um desafio que afeta toda a ciência da computação, pois ela faz com que conceitos como sistemas de computadores e de *hardware* sejam repensados; faz pensar como criar sistemas ubíquos complexos e como entender ambientes que são suportados pela computação ubíqua. Há muito ainda que ser pesquisado e experimentado nesta área da ciência da computação para que todos os conceitos relacionados a computação ubíqua se tornem realidade.

Na seção 2.1.2 são apresentadas as principais características de um ambiente de computação ubíqua para melhor entendimento da seção 2.1.3, que apresenta alguns

[1] http://ubicomp.org
[2] http://www.computer.org/pervasive

dos desafios desta área da computação.

2.1.2 Características

Como visto na seção anterior, o propósito da computação ubíqua é servir as pessoas, ou seja, auxiliá-las em suas tarefas, sem que muitas vezes, elas percebam. Porém, antes da realização de um profundo estudo de computação ubíqua, é necessário conhecer quais as qualidades que caracterizam um sistema de computação ubíqua. Conforme (CHALMERS et al., 2006), um sistema de computação ubíqua deve demonstrar as seguintes qualidades:

- deve ser fluido, ou seja, sua estrutura varia a curto prazo e evolui a longo prazo;
- cada entidade não-humana tem suas ações expressas vagamente ou formalmente;
- é parcialmente autônomo: algumas de suas ações são determinadas por seu propósito e sua experiência interativa, ao invés de por invocação de uma entidade em um nível mais alto;
- é reflexivo: um subsistema pode relatar sua experiência a um sistema de nível mais alto (talvez para uma pessoa), para permitir intervenção ou orientação;
- é confiável: se comportará de uma maneira segura e não afetará adversamente a informação, outros componentes do sistema ou as pessoas;
- é sustentável: seus componentes (hardware e software) são projetados e construídos para ter longa duração, manutenção eficiente e efetiva e eventual decomposição, enquanto seu impacto no ambiente é apropriado mas, mínimo;
- é eficiente: qualquer atraso em seu desempenho é tolerável;
- é escalável: seus subsistemas podem ter diferença em tamanho, contudo, uma possível complexidade intratável será evitada aplicando os mesmos princípios de projeto e métodos de análise a cada nível.

Obviamente, estas devem ser um pequeno conjunto de qualidades que um sistema de computação ubíqua deve apresentar. Além dessas qualidades, características

tais como: invisibilidade, consciência de contexto, privacidade e confiança, também se destacam quando o assunto é computação ubíqua.

2.1.2.1 Invisibilidade

Um sistema apresenta a característica de invisibilidade quando requer intervenção humana mínima. De acordo com (WEISER, 1991), é o desaparecimento completo da tecnologia da consciência de um usuário. Ou seja, a distração mínima do usuário.

A intervenção humana pode ser necessária quando, por exemplo, os ambientes inteligentes não satisfizerem automaticamente a sua expectativa. Tal intervenção também poderia ser parte de um ciclo de aprendizagem contínuo do ambiente (SAHA; MUKHERJEE, 2003). Porém, satisfazer continuamente as expectativas do usuário exige do ambiente e dos objetos uma atualização sem provocar a distração dos usuários em um nível consciente.

2.1.2.2 Consciência de Contexto

Consciência de contexto é uma característica intrínseca de ambientes inteligentes, que torna um sistema de computação ubíqua invasivo o mínimo possível. Ou seja, o sistema é capaz de conhecer o estado do usuário e do seu ambiente, e alterar seu comportamento baseado nesta informação (SATYANARAYANAN, 2001).

O principal desafio é como o sistema obtém as informações necessárias para tornar-se consciente de contexto. As informações que definem a consciência de contexto devem ser precisas; caso contrário, podem confundir ou podem intrometer na experiência do usuário (SAHA; MUKHERJEE, 2003).

2.1.2.3 Privacidade e Confiança

A privacidade é um grande problema em sistemas distribuídos e na computação móvel, e é um assunto ainda mais complicado na computação ubíqua. Através de um sistema de computação ubíqua é possível obter informações sobre os movimentos de um determinado usuário, padrões de comportamento e hábitos. Estas informações são importantes para que o sistema apresente consciência de contexto, porém, a menos que o uso desta informação seja estritamente controlado, estas informações

podem ser utilizadas para várias finalidades, por exemplo, para chantagear o usuário (SATYANARAYANAN, 2001).

Também é um grande desafio estabelecer confiança em um sistema de computação ubíqua, de modo que este não seja intrusivo, preservando a característica de invisibilidade. Os problemas são como garantir e provar que um ambiente não está monitorando os usuários em todos os lugares e obtendo informações que não lhe foram autorizadas (CAMPIOLO, 2005).

2.1.3 Desafios

Para suprir as necessidades que um sistema de computação ubíqua tem para atender as suas características e qualidades ideais, como exposto nas seções anteriores, é necessário ainda vencer muitos dos seus desafios, que impedem que esta área da computação seja atendida por completo. Nesta seção são apresentados os principais desafios enumerados por (CHALMERS et al., 2006).

2.1.3.1 O Entendimento do Contexto

O entendimento da natureza das atividades humanas é um ramo de pesquisa muito importante para muitas disciplinas, tais como, psicologia, sociologia, e ergonomia. Essa variedade de perspectivas gera vários problemas, visto que, cada disciplina pode explorar táticas diferentes para entender as ações humanas. O entendimento destas diferentes disciplinas é utilizado na computação ubíqua com a finalidade de representar atividades humanas, e pode resultar em um desafio de multidisciplinaridade significante.

A diversidade de abordagens dentro das ciências humanas e sociais aumenta a complexidade envolvida. A abordagem de computação ubíqua freqüentemente utiliza os conceitos da sociologia. Dessa forma, a riqueza de interação humana é enfatizada mas, também tende a resistir a representações mais abstratas de atividades, que é importante aos desenvolvedores de sistemas de computação ubíqua.

2.1.3.2 Interação com Ambientes

Ambientes de computação ubíqua estão cada vez mais se tornando parte da vida cotidiana das pessoas. Dessa forma, é preciso entender como as pessoas interagem com eles e como os exploraram. É necessário entender também de que forma a interação do usuário contribui para a formação destes ambientes.

As pessoas enfrentam desafios ao tentar estabelecer uma interação útil ou agradável com tecnologias de computação ubíqua. Esta interação precisa ser ajustada com o contexto, interesse e objetivos dessas pessoas. As capacidades de tecnologias digitais diferentes, configurações de uso e experiências passadas dos indivíduos agem como recursos e restrições na formação desta interação.

Reduzir estas diversas formas de interação é fundamental e central no projeto de ambientes ubíquos. Um desafio importante para computação ubíqua é descobrir como as pessoas chegam a padrões de interação que são produtivos ao invés de problemáticos.

2.1.3.3 Privacidade e Confiança

Privacidade e confiança é uma preocupação tanto para quem cria ambientes ubíquos, quanto para quem os utiliza. Para que o público em geral possa confiar em um ambiente com informação pessoal, primeiro eles têm que considerar que este ambiente é confiável.

O problema de como descobrir quais as garantias que são requeridas pelas pessoas e como estas garantias devem ser providas pela infra-estrutura, se pelo uso das tecnologias pessoais ou até mesmo por uma entidade reguladora, ainda não possui solução, sendo necessário a realização de pesquisas para suprir essas dificuldades em ambientes ubíquos.

2.1.3.4 Comunicações sem fio

O fato de que dispositivos ubíquos serão incorporados em entidades móveis como pessoas e veículos, como também serão conectados milhares de dispositivos a uma infra-estrutura, significa dizer que a comunicação sem fio é essencial. Porém, comunicações sem fio requerem freqüentemente muito mais energia que outras formas de

processamento. Há muitos padrões diferentes de comunicação sem fio sendo utilizados em vários ambientes, tais como: telefones celulares, Bluetooth, Wi-Fi, Wi-Max, Zigbee etc.; e cada um com suas próprias tarifas. Os usuários móveis podem precisar trocar dados com outros dispositivos com tipos diferentes de comunicação, dependendo dos seus contextos atuais ou gerenciando custos ou preocupações com segurança e privacidade.

2.1.3.5 Auto-configuração

A escala de sistemas de computação ubíqua demanda sistemas autonômicos (auto-organização, auto-gerenciamento e auto-recuperação) para simplificar a instalação e evolução de dispositivos, serviços e aplicações. Não será possível para os usuários não-técnicos instalarem software e configurarem milhares de dispositivos em suas casas, carros e empresas. Por outro lado, se um sistema pode se auto-configurar, surge o seguinte problema: como um agente (humano ou digital) que está migrando pode descobrir os recursos e serviços dos quais precisa em seu novo local.

Além disso, pode acontecer do sistema ter que ser parcialmente autonômico como, por exemplo, necessitar da intervenção humana em circunstâncias excepcionais, ou quando os usuários desejarem intervir nas decisões. Um desafio chave de projeto é permitir uma variação entre graus diferentes de autonomia, baseado em uma compreensão de quando e como envolver as pessoas para adaptar o comportamento.

2.2 Considerações

Esse capítulo apresentou os principais conceitos inerentes à área da computação ubíqua, dentre eles, um breve histórico, principais características e desafios, dando assim, uma visão geral ao assunto.

O próximo capítulo discorre sobre o assunto de privacidade em computação ubíqua. Dentre todas as características e desafios existentes na área de ubíqua, vistos nesse capítulo, a privacidade é considerada uma característica não somente desejada, mas essencial para ambientes ubíquos e, por isso, abordada com mais detalhes.

3 Privacidade

Atualmente, privacidade é um direito de qualquer pessoa, e muitas nações têm em suas constituições leis que garantem ao cidadão o direito de possuí-la. Entretanto, privacidade não pode ser apenas garantida por leis, principalmente quando se trata de dados digitais. Este problema vem sendo abordado em computação há algum tempo e a solução que tem sido utilizada é o uso de criptografia. Essa solução tem sido satisfatória para o paradigma atual, a computação pessoal.

A computação ubíqua é um novo paradigma onde os ambientes possuem dispositivos e sensores computacionais com capacidade de computação e comunicação. O usuário se comunica com esses ambientes através de seus dispositivos pessoais e vice-versa. Em computação ubíqua, a privacidade ganha novas dimensões, as quais foram muitas vezes idealizadas por livros e filmes, e que nos tempos atuais, estão se tornando realidade.

Nesse capítulo são apresentadas e discutidas as novas dimensões de privacidade, no contexto da computação ubíqua, assim como as questões que estão sendo abordadas pela comunidade científica.

3.1 História da Privacidade

Como é mostrado em (LANGHEINRICH, 2001), a privacidade já era um assunto preocupante no século 19, quando Samuel Warren e Louis Brandeis escreveram o artigo "*The Right to Privacy*" (WARREN; BRANDEIS, 1890), motivados em grande parte pelo advento da fotografia moderna e pelos novos métodos de impressão. Os autores definiram a privacidade como "o direito de poder ficar sozinho", como uma forma de criticar os repórteres que tiravam fotos das pessoas sem permissão. Ultimamente a privacidade é tida mais como "o direito de selecionar quais informações

pessoais podem ser reveladas e quais pessoas terão permissão de receber essas revelações"(WESTIN, 1967).

A privacidade se tornou um assunto importante nos anos 60 quando os governos descobriram o processamento de dados automatizado como um meio efetivo de catalogar seus cidadãos. O exemplo da exploração do nazismo feita sobre os registros públicos detalhados na Segunda Guerra Mundial (o que facilitava a perseguição à população judia) fez com que muitas nações européias passassem a criar várias leis de proteção aos dados para prevenir qualquer abuso de tal informação armazenada (LANGHEINRICH, 2001). Ultimamente, a proteção à privacidade se destaca como o uso de informações pessoais relativas ao sigilo bancário, segredos de justiça, dados da área médica, mas um assunto preocupante devido ao uso crescente da Internet, vem sendo as transações eletrônicas, como também informações pessoais divulgadas inadequadamente.

Os objetivos da proteção à privacidade foram sendo alterados conforme os desenvolvimentos de novas tecnologias. Podem ser encontrados, na literatura, assuntos sobre privacidade desde 1361, quando o *Justices of the Peace Act* na Inglaterra estabeleceu o arresto quanto ao ato de curiosidades sobre informações alheias e intromissões, estabelecendo assim a primeira noção de *privacidade nos meios de comunicação* (MICHAEL, 1994). No século 18, apareceu a primeira noção de *privacidade territorial* (LANGHEINRICH, 2001), onde foi exemplificada uma situação que demonstra que, em nenhuma situação, um cidadão pode ultrapassar os limites de uma moradia. Com o uso crescente do sistema telefônico nos anos 30, houve uma grande preocupação com a legalidade de interceptores de conversas telefônicas utilizados pelo governo dos Estados Unidos (LANGHEINRICH, 2001). A *privacidade pessoal* foi violada seriamente, e após alguns anos, na década de 40, quando a liderança do nazismo decidiu conduzir a esterilização compulsória, assim como experiências médicas imorais, o abuso quanto à privacidade apareceu novamente. Dos anos 60 em diante, o uso crescente do processamento de dados eletrônicos deu origem à *privacidade da informação* (LANGHEINRICH, 2001).

Essas categorias de privacidade estão estabelecidas quanto aos seus aspectos legais em todo o mundo, os quais são freqüentemente definidos como direitos constitucionais. Porém, é a *privacidade de informação* que dá origem à maioria dos desafios atualmente (BHASKAR; AHAMED, 2007). Embora as leis de privacidade de

informação tivessem origem há mais de 30 anos, o progresso rápido da tecnologia, como o sucesso comercial da *World Wide Web*, desafia a legislação que foi inventada no tempo onde se utilizava *mainframes* (IBM 360, IBM 370, IBM 3090, IBM 4341, Burroughs 3700, Burroughs 6700) e cartões perfurados (IBM 029) (LANGHEINRICH, 2001).

3.2 Definição

Alan F. Westin afirma que "nenhuma definição de privacidade é possível, porque assuntos de privacidade são questões de valores, interesses e poder"(WESTIN, 1995) (WESTIN, 1967).

Dessa forma, não é possível elaborar uma definição única para privacidade que seja capaz de expressar seu significado nas mais diversas situações em que é considerada. Na maioria dos casos em que se discutem aspectos relativos à privacidade, as discussões se encaminham para a visão relativa da situação e dos indivíduos envolvidos.

A partir dessa afirmação, é possível isolar duas variáveis essenciais para que se tenha uma questão de privacidade: o indivíduo e a informação (SALTZER; SCHROEDER, 1975). Indivíduo é a pessoa que se sentiu lesada tendo algo que é privado (particular) tornado público ou acessado por alguém sem o seu consentimento. Informação é uma abstração genérica para expressar algo que é tido como privado para um indivíduo.

3.3 Classificação

Conforme já introduzida na seção 3.1 e oficializada pela *A Global Internet Liberty Campaign* (GILC, 2008), uma organização formada por diversas outras organizações envolvidas com direitos humanos e de liberdade, são quatro as grandes categorias de privacidade:

Privacidade de informação: refere-se ao gerenciamento da informação pessoal, tais como registros médicos e informação de crédito;

Privacidade corporal: refere-se à proteção física das pessoas contra procedimentos invasivos, tais como testes de drogas;

Privacidade de comunicação: refere-se à proteção e privacidade dos meios de comunicação, tais como o correio convencional, o correio eletrônico e o telefone;

Privacidade territorial: refere-se à definição do limite de intrusão nos ambientes domésticos e no espaço público.

A próxima seção discute algumas legislações mais influentes de privacidade, e como elas podem influenciar no projeto de sistemas de processamento de dados e sua infra-estrutura.

3.4 Legislação

A maioria dos países reconhecem em suas constituições algum nível de privacidade. No mínimo, os direitos da inviolabilidade do lar (privacidade territorial) e a confidencialidade das comunicações estão declaradas (RIGHTS, 2008).

No Brasil, o artigo 5 da Constituição declara que são invioláveis a intimidade, a vida privada, a honra e a imagem das pessoas; o lar é inviolável e ninguém pode entrar sem o consentimento do morador; a correspondência e a comunicação eletrônica são sigilosas (CONSTITUICAO, 1988).

A proteção da informação é garantida por declarações e legislações em diversos graus, dependendo do país. Entretanto, todas concordam que a informação pessoal deve ser obtida amigavelmente e dentro da lei; usada somente pelo propósito especificado originalmente; adequada, relevante e não excessiva ao propósito; precisa e atualizada; destruída depois que seu propósito é completado.

A constituição e a legislação brasileira não abordam essa questão com mais profundidade. Diversas propostas de leis já foram submetidas, mas não foram aprovadas ou estão tramitando no Congresso Nacional. Conseqüentemente, o Brasil não possui leis que protegem os dados e os direitos da privacidade individual. Diferentemente, a Argentina possui um ato específico e detalhado abordando esta questão (ARGENTINA, 2008).

Contudo, as mais influentes legislações de privacidade, mundialmente conhecidas, são: *US Privacy Act* de 1974 (USPRIVACYACT, 1974) e *EU Directive 95/46/EC* de 1995 (EUDIRECTIVE1995/46/EC, 1995).

A legislação *US Privacy Act* de 1974 influenciou o desenvolvimento de políticas de privacidade do mundo inteiro. Essa legislação define alguns princípios de privacidade, os quais são baseados no trabalho de Alan Westin (WESTIN, 1995) (WESTIN, 1967). Esses princípios são os seguintes:

Abertura e transparência: nenhum registro deve ser secreto. Isso inclui tanto a publicação da sua existência como também o seu conteúdo;

Participação individual: o assunto de um registro deve ser visível e sua correção deve ser permitida;

Limitação de coleção: a coleção de dados deve ser proporcional e não excessiva ao propósito da coleção;

Qualidade dos dados: os dados devem ser relevantes aos propósitos para os quais são coletados e devem estar sempre atualizados;

Limitação de uso: os dados devem ser usados para o seu propósito específico e por pessoas autorizadas;

Segurança aceitável: proteções de segurança adequadas devem ser usadas de acordo com a importância dos dados coletados;

Responsabilidade: os gerenciadores dos registros são os responsáveis por assegurar os demais princípios.

Embora esses princípios de privacidade estivessem incorporados em várias legislações no mundo inteiro, o *US Privacy Act* de 1974 não era um sucesso nos Estados Unidos (GELLMAN, 1998). Em 1980, a *Organization for Economic Cooperation and Development* (OECD) classificou as informações de privacidade nas Diretrizes da OECD (OECD, 1980) para evitar uma proliferação de leis de proteção à privacidade que poderiam prejudicar o crescimento econômico através da criação de barreiras comerciais (LANGHEINRICH, 2001).

Os países europeus continuaram a desenvolver e refinar seus documentos de proteção à privacidade abrangendo tanto coleção de dados governamentais como privados. No entanto, a legislação dos Estados Unidos seguiu com a definição de documentos mais específicos, como por exemplo: *Fair Credit Reporting Act* de 1970, *Video Privacy Protection Act* de 1988, *Family Education Rights and Privacy Act* de 1994 (LANGHEINRICH, 2001).

Somente em 1995 foi divulgada uma nova legislação tão influente quanto à dos Estados Unidos, só que dessa vez na Europa. De acordo com Langheinrich (2001), a *EU Directive 95/46/EC* da União Européia sobre a proteção de indivíduos no que diz respeito ao processamento de dados pessoais e no movimento livre de tais dados (EUDIRECTIVE1995/46/EC, 1995) é a legislação de privacidade do fim do século 20.

A *EU Directive 95/46/EC* apresentou alguns impactos (LANGHEINRICH, 2001). Em primeiro lugar, seu artigo 25/1 limita transferências de dados a países que não são da União Européia, somente se eles tiverem "um nível adequado de proteção à privacidade". A ameaça de ficar fora dos fluxos de dados europeus fez com que muitos países revisassem as suas legislações de privacidade para obedecer às exigências da *EU Directive 95/46/EC*. Em segundo lugar, a *EU Directive 95/46/EC* não só providencia e refina as práticas de privacidade descritas acima, como também acrescenta no seu artigo 7, noções de consentimentos explícitos: dados pessoais somente podem ser processados se o usuário não apresentar ambigüidades nas informações fornecidas por ele (podem haver exceções para propósitos legais e contratuais). Essa consideração praticamente repudia todos os tipos de coleções de dados (com exceção das que são exigidas por lei) e requer um consentimento explícito caso a caso conforme a importância dos dados.

Como muitos profissionais da computação podem ignorar assuntos legais no momento de projetar sistemas computacionais e se concentram somente nas possibilidades técnicas atuais, a *EU Directive 95/46/EC* criou em 1998 uma referência importante para a proteção de privacidade, que não pode ser ignorada. A *EU Directive 95/46/EC* destaca tanto a relevância da proteção de privacidade na era de processamento de dados digitais quanto a importância de cooperação internacional para alcançar essa proteção.

3.5 Privacidade em Computação Ubíqua

Em computação ubíqua, a tecnologia está muito próxima dos indivíduos e reside nos mais diversos cenários reais que possam ser considerados. Segundo este paradigma, a computação deve ser invisível, ou seja, causar o mínimo possível de distração ao usuário.

Baseado nesta idéia, não é aceitável que os usuários sejam interrompidos freqüentemente com opções e alertas para configurar, aceitar ou rejeitar algum tipo de comunicação do ambiente que seja intrusiva. Nesta seção, são apresentados serviços que podem ser intrusivos em ambientes de computação ubíqua, as questões que devem ser consideradas e as restrições que podem ser impostas a esses serviços.

3.5.1 Serviços

A maioria dos serviços em computação ubíqua ainda não são muito difundidos ou aplicados em ambientes reais (AOYAMA, 2008). A seguir são descritos alguns desses serviços e suas respectivas implicações à privacidade.

3.5.1.1 Serviço de identificação de produtos

Através desse serviço é possível contabilizar e rastrear os produtos que um consumidor está adquirindo. As implicações são a falta de controle dos tipos e do uso das informações que estão sendo coletadas e a possibilidade de uma entidade externa rastrear o conteúdo da compra fora da loja.

3.5.1.2 Serviço de alerta de proximidade

Este serviço informa quando uma pessoa conhecida e cadastrada no seu dispositivo está próxima de sua localização física ou em um ambiente comum. A problemática é que nem sempre as pessoas desejam ser localizadas em determinadas situações ou horários.

Além disso, tem-se as seguintes questões: (a) O sistema deve informar a localização exata da pessoa ou apenas emitir um alerta de sua presença? (b) Como bloquear a localização para determinadas pessoas ou em determinadas situações sem

quebrar a noção de invisibilidade da tecnologia?

3.5.1.3 Serviço de propaganda

Este serviço envia propagandas e anúncios de produtos aos dispositivos de usuários quando estes se aproximam de seus estabelecimentos comerciais. Neste caso, há diversas questões: (a) Como definir as políticas para restringir as propagandas? (b) Como evitar os protocolos desgastantes para obter estas informações? (c) Como mapear os interesses de um cliente? (d) Como definir os limites para realizar esse mapeamento?

3.5.2 Restrições

Os serviços introduzidos com a computação ubíqua tem o objetivo de facilitar a execução de tarefas do usuário. Porém, os serviços precisam respeitar certas restrições para que não se tornem intrusivos.

Uma classificação para estas restrições é apresentada a seguir (MYLES; FRIDAY; DAVIES, 2003):

Temporal: determina períodos de tempo em que o serviço está disponível ou desabilitado. Por exemplo, um usuário não gosta de ser localizado no período do almoço;

Localização: restringe o acesso às informações ou ao dispositivo baseado na localização do usuário. Por exemplo, em um restaurante o usuário pode permitir a um serviço obter seu nome para um atendimento personalizado;

Organização: define quem e quando uma entidade pode localizar o usuário. Por exemplo, um trabalhador de uma empresa deseja somente ser localizado enquanto estiver nos limites físicos da empresa;

Serviço: define quais serviços um dispositivo cliente pode acessar. Por exemplo, ao entrar em um ambiente com serviços de anúncios, pode-se restringir quais tem permissão para o acesso;

Pedido: define quais informações podem ser reveladas para um determinado serviço.

Por exemplo, ao preencher um cadastro o cliente define quais dados são relevantes para serem transmitidos do seu dispositivo para o cadastro;

Situação: define as situações em que as políticas definidas para um serviço podem ser sobrepostas. Este tipo de serviço requer um nível de inteligência dos dispositivos. Por exemplo, um usuário não quer acessar nenhum serviço, enquanto estiver na sala de reuniões com seu chefe;

Grupo: define um grupo comum que pode acessar um conjunto de informações do usuário. Essa restrição se aplica a dispositivos de outros usuários. Por exemplo, um usuário quer compartilhar as informações de trabalho com todos que fazem parte do seu setor;

Interesse: define se serviços ou informações transmitidas para o dispositivo são de interesse do usuário. Por exemplo, um usuário pode desejar receber resultados de jogos de futebol, logo ele pode definir como interesse esse tipo de informação.

Na próxima subseção são apresentadas algumas razões para a invasão de privacidade, obtidas em (BERESFORD, 2005).

3.5.3 Razões para invasão de privacidade

As razões para invasão de privacidade são muitas e variadas. Os governos normalmente querem obter informações dos cidadãos, como por exemplo, salários, padrão familiar, religião e qualificações. O objetivo declarado dos dados coletados é que favorece o crescimento da sociedade; em muitos casos o benefício está claro (por exemplo, a regulamentação de médicos), mas em alguns casos os benefícios da sociedade são menos óbvios, por exemplo, o registro de origem racial.

Organizações comerciais estão relacionadas principalmente aos lucros, e são muito mais invasivas que os governos. As propagandas ou os cartões de fidelidade foram sugeridos como a principal motivação mas, mais recentemente a discriminação de preço [1] foi sugerida como um fator motivador forte. Exemplos de discriminação de preço incluem vôos, computadores e até mesmo DVDs. A automatização da coleção e análise de perfis do consumidor (freqüentemente chamados de mineração

[1] A discriminação de preço é o ato de oferecer aos indivíduos um preço personalizado baseado na quantia que eles estão dispostos a pagar

de dados) fez crescer a habilidade das corporações em atribuir dinamicamente preços aos produtos, conforme o perfil do consumidor. Odlyzko (ODLYZKO, 2003) mostra através da teoria econômica que a discriminação de preço combinada com um perfil eficiente do consumidor resulta em um mercado mais eficiente e como ainda não possui regulamento, seu uso será difundido.

A mineração de dados, independente do contexto onde for aplicada, irá impactar diretamente na privacidade do consumidor. Nem todos os usuários parecem estar interessados com a coleção de dados e políticas de retenção; e, ao contrário, a maioria dos usuários está satisfeito com tal processo, fazendo com que consumidores menos satisfeitos sejam forçados a participar, ou por falta de escolha ou por ser caro demais para os indivíduos que optam por ter privacidade.

Exemplos atuais de ameaças à privacidade da informação são os sistemas de identificação (biometria, cartões de identificação); a vigilância das comunicações (Internet, telefone); a vigilância por vídeo, satélite ou áudio; o comércio eletrônico (levantamento de perfis, spam, dados coletados); o monitoramento em locais de trabalho; rastreamento de localização, entre outras.

3.5.4 Métodos de proteção à privacidade

Lessig (LESSIG, 1999) descreve os quatro métodos básicos de regular o comportamento de indivíduos: arquitetura, costumes sociais, mercado e lei. A figura 3.1 apresenta a interação entre o indivíduo e os quatro métodos básicos de regulamentação, onde cada método representa um custo distinto mas interdependente no indivíduo. As restrições desses métodos podem ser complementares ou não, contudo podem ser regidas por lei.

A arquitetura e o mercado regulam o comportamento antes que uma ação venha a acontecer, por exemplo: uma porta fechada previne a entrada não desejada e o custo do cigarro intimida o seu consumo. Costumes sociais e leis regulam o comportamento após a execução de uma ação, por exemplo: os restaurantes podem ignorar o fato de um cliente fumar durante uma refeição, mas a polícia pode prendê-lo por ter infringido uma propriedade privada. A maioria dos indivíduos se sente ameaçado pelos costumes sociais ou pelas leis, antes da execução de uma ação, embora os costumes sociais ou leis não possam evitar que ele a pratique, ao contrário das forças

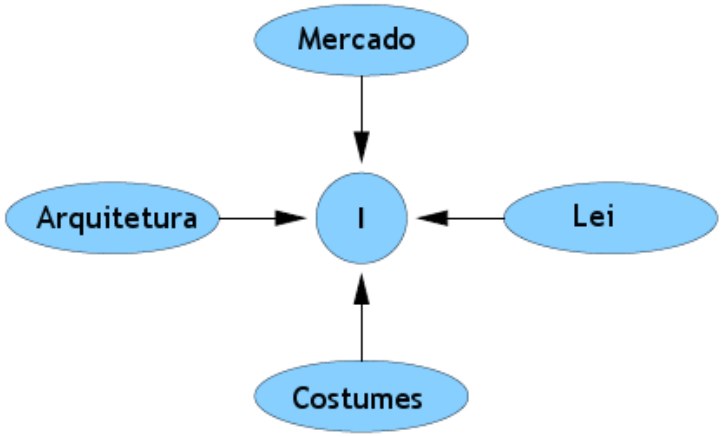

Figura 3.1: Os quatro métodos de regulamentação de um indivíduo I (BERESFORD, 2005).

arquiteturais e de mercado (BERESFORD, 2005).

Dessa forma, pode-se dizer que a lei, por si só não é suficiente como também não é o método mais eficiente para prevenção de intrusão não desejada nas vidas privadas dos indivíduos porque (BERESFORD, 2005): (1) a privacidade é muito subjetiva, ou seja, o que é aceitável a uma pessoa é inaceitável a outra; (2) a aplicação de uma lei após a execução de uma ação pode não ser uma boa solução, pois a informação já foi descoberta. Isto não significa que os legisladores não provêem nenhuma contribuição à proteção de privacidade; a lei tem um papel crucial no controle arquitetural, forças de mercado e costumes sociais.

3.5.5 Métodos para a regulamentação de privacidade

Técnicas para proteger a privacidade pessoal diferem notadamente pelo mundo afora. Porém, quatro abordagens se destacam (BERESFORD, 2005):

Amplo: a Europa, Austrália, Nova Zelândia e Canadá têm um modelo regulador amplo com um funcionário público responsável por garantir a aplicação da legislação de proteção de dados;

Setorial: os Estados Unidos não definiram uma legislação de proteção de dados geral, mas ordenou uma série de leis específicas ao invés de lidar com problemas conforme o seu surgimento;

Auto-regulamentação: várias indústrias tentaram fazer uma auto-regulamentação através de códigos de prática, porém muitos desses não foram efetivos e têm menos impacto que uma legislação; além disso, consensos industriais não resultam necessariamente em uma boa privacidade ao consumidor, embora a ameaça de intervenção do governo possa ter algum efeito positivo;

Tecnológico: a tecnologia pode prover soluções como também ameaças à privacidade pessoal. *Pretty Good Privacy* (PGP) é um exemplo bem conhecido de tecnologia que habilita a privacidade de comunicações, contudo, o governo dos Estados Unidos proibiu seu uso e a sua distribuição.

3.5.6 Algumas soluções técnicas para proteção à privacidade

Esta seção descreve algumas soluções técnicas emergentes para proteção à privacidade de informação pessoal. O maior problema encontrado em privacidade é que as ferramentas de coleta de informações não são projetadas para prover o direito do usuário à privacidade (BUENNEMEYER; MARCHANY; TRONT, 2006).

O desenvolvimento de sistemas de coleta de informações contribui para que dados pessoais sejam utilizados de forma abusiva e compartilhados de maneira imprópria. Bayardo (BAYARDO; SRIKANT, 2003) afirma que casos de revelação imprópria e completamente abusivos de informação pessoal afetam o comportamento individual e coletivo. Dessa forma, soluções técnicas devem ser desenvolvidas para ajudar a proteger a nossa privacidade. O desafio é permitir que algumas minerações de dados melhorem a vida dos usuários e, ao mesmo tempo, proteja suas informações pessoais. Há poucas ferramentas para gerenciar a privacidade dos dados pessoais no momento. Porém, mais tecnologias de proteção à privacidade vêm sendo desenvolvidas.

As tecnologias de proteção de privacidade envolvem basicamente 5 áreas gerais (BAYARDO; SRIKANT, 2003): bancos de dados de Hippocrates, codificação da política de privacidade, anonimato, mineração de dados com a opção de preservar a privacidade, e compartilhamento de informações por repositórios privados. As próximas subseções apresentam brevemente cada uma dessas áreas baseadas em (BAYARDO; SRIKANT, 2003).

3.5.6.1 Bancos de dados de Hippocrates

Os bancos de dados de Hippocrates assumem responsabilidade pela privacidade dos dados que eles gerenciam como uma premissa essencial por meio do emprego de 10 princípios de privacidade, conforme citado em (AGRAWAL et al., 2002):

Especificação do propósito: para cada informação pessoal armazenada no banco de dados, serão associados os propósitos para os quais a informação foi coletada;

Consentimento: os propósitos associados com a informação pessoal terão consentimento do proprietário da informação;

Limitação de coleção: são coletadas o mínimo possível de informações pessoais para realizar os propósitos especificados;

Limitação de uso: o banco de dados executará somente as consultas que são consistentes com os propósitos para os quais a informação foi coletada;

Revelação limitada: as informações pessoais armazenadas no banco de dados não serão reveladas para propósitos diferentes dos quais houve consentimento do proprietário da informação;

Retenção limitada: as informações pessoais somente permanecerão retidas o tempo necessário para o cumprimento dos propósitos para os quais foram coletadas;

Precisão: as informações pessoais armazenadas no banco de dados serão precisas e atualizadas;

Segurança: as informações pessoais serão protegidas por técnicas de segurança contra furtos e outros desvios;

Abertura: um proprietário poderá ter acesso a toda a informação armazenada sobre ele no banco de dados;

Conformidade: um proprietário poderá verificar a conformidade com os princípios anteriores.

De acordo com (BAYARDO; SRIKANT, 2003), esses bancos de dados obrigam automaticamente a utilização dos princípios de privacidade através da verificação

de informações de privacidade relativas à consulta autorizada de usuários, as quais são estritamente baseadas em uma política de privacidade. O banco de dados confere explicitamente se o usuário tem ou não acesso aos campos dos dados, e então permite o acesso aos registros que têm os seus atributos autorizados.

3.5.6.2 Codificação da política de privacidade

Uma política de privacidade pode ser entendida como um conjunto de regras criado por uma organização ou indivíduo para determinar quais entidades ou serviços podem ter acesso às suas informações e, em alguns casos, para determinar também em quais contextos as suas informações podem ser utilizadas (BAYARDO; SRIKANT, 2003).

A codificação de políticas de privacidade permite que uma organização codifique sua coleção de dados e práticas usadas em um modelo automatizado. Os usuários podem estabelecer um perfil de privacidade que é conferido automaticamente quando eles entram no ambiente. Há dois exemplos importantes de codificação de política de privacidade: *IBM's Enterprise Privacy Authorization Language* (EPAL) e o World Wide Web Consortium's (W3C) *Platform for Privacy Preferences Project* (P3P) (W3C, 2007). O objetivo da EPAL é permitir que os sistemas de privacidade como o *Tivoli Privacy Manager* obriguem o uso de políticas de privacidade em empresas.

Já a W3C desenvolveu a iniciativa P3P para permitir a combinação entre políticas de privacidade e preferências do usuário. Essa iniciativa está emergindo como um padrão de fato para disponibilizar meios simples e automatizados para os usuários ganharem mais controle sobre o uso de suas informações pessoais que são coletadas em sua maior parte em *sites Web*. O P3P engloba um conjunto padrão de questões que cobrem a maioria dos aspectos da política de privacidade de um *site Web*. A idéia é que um sistema habilitado possa comparar automaticamente a política de privacidade do *site Web* com o perfil de privacidade do usuário. O usuário retém o controle da privacidade de suas informações baseado no seu perfil e, mais importante, o P3P permite que o sistema notifique o usuário em um formato compreensível e, dessa forma, ele pode tomar decisões com relação à liberação de informações adicionais. Essa parece ser a primeira tentativa de padronizar e categorizar a privacidade das informações pessoais em um formato de interface que é compreensível tanto pelos usuários quanto pelos sistemas.

3.5.6.3 Anonimato

Ferramentas que estabelecem o anonimato de um usuário são comuns atualmente, mas elas são ferramentas tipicamente baseadas na Web. No futuro, essas ferramentas irão envolver e migrar para os ambientes ubíquos. De acordo com Bayardo (BAYARDO; SRIKANT, 2003), ferramentas de anonimato permitirão aos usuários a possibilidade de impedir que organizações coletem informações sobre eles devido ao fato dessa tecnologia prevenir a coleta de dados pois esconde a identificação da informação. Neste caso, por exemplo, ao entrar em um ambiente ubíquo, o usuário poderá se conectar ao sistema por um *proxy* de privacidade o qual permitirá que ele faça compras anonimamente sem ter que compartilhar informações pessoais.

Um exemplo de anonimato são as zonas de mixagem (*mix zones*) (BERESFORD; STAJANO, 2003). Zonas de mixagem são regiões espaciais conectadas de tamanho máximo definido, onde nenhum dos usuários pode ser localizado por serviços. Durante a permanência nessas zonas a identidade do usuário é "mixada", ou seja, o usuário recebe um novo pseudônimo. Com isso, dispositivos que conheciam sua identificação anterior não saberão mais sua identidade ou localização no ambiente.

As zonas de mixagem permitem introduzir um grau de anonimato e evitar o rastreamento do usuário por um determinado dispositivo. Elas devem estar localizadas em pontos movimentados, onde grupos de usuários se concentrem para que a mixagem seja mais eficiente.

3.5.6.4 Mineração de dados com a opção de preservar a privacidade

A mineração de dados com a opção de preservar a privacidade combina aspectos de várias áreas de tecnologias de privacidade emergentes. O anonimato pode impedir que companhias conheçam a base de dados de seus clientes e dessa forma impede também que elas se esforcem para melhorar seus produtos e serviços (BAYARDO; SRIKANT, 2003). A mineração de dados com a característica de privacidade deveria permitir que as operações comerciais derivassem as informações necessárias para compreender os hábitos de compra do cliente sem coletar qualquer informação pessoal. Essa abordagem impede a retenção de informações pessoais significantes sobre o usuário, mas permite que operações comerciais construam modelos de informações do cliente agregadas para melhorar as decisões comerciais.

3.5.6.5 Compartilhamento de informações por repositórios privados

O compartilhamento de informações entre os repositórios privados representa um grande desafio, pois as políticas de segurança não são perfeitamente alinhadas entre os diferentes bancos de dados. O fato dos consumidores poderem, em alguns casos, optar por revelar suas informações pessoais, não significa necessariamente que eles queiram que as suas informações sejam combinadas para gerar dossiês de consumidores com um elevado nível de detalhe. Quando a informação é distribuída entre esses bancos de dados, o problema é como permitir que as companhias desenvolvam modelos agregados para compartilhar informações sem ter que revelar os dados de privacidade individuais (BAYARDO; SRIKANT, 2003). No futuro, *frameworks* para compartilhamento de informações entre banco de dados deverão ser desenvolvidos para alcançar esta possibilidade de compartilhar informações e ainda proteger a privacidade pessoal.

3.5.6.6 Considerações

Nessa seção foram descritas algumas tecnologias existentes e que ainda estão sendo desenvolvidas para permitir o compartilhamento de informações e ainda para proteger a privacidade individual. Acredita-se que com os avanços tecnológicos e com a resolução dos problemas de ataques à privacidade será possível superar o desafio de proteger a privacidade individual e se beneficiar de um maior compartilhamento de informações.

3.5.7 Implicações sociais da computação ubíqua

Para se entender o que faz a computação ubíqua diferente dos outros domínios de informática, no que diz respeito à privacidade, e porque os cientistas da computação nesse domínio particular deveriam se preocupar mais com tais noções vagas de liberdade e privacidade, é preciso conhecer quatro propriedades fundamentais (LANGHEINRICH, 2001):

Onipresença: a computação ubíqua está em todos os lugares, essa é a sua essência, seu objetivo explícito. Por conseqüência, as decisões feitas nos sistemas de computação ubíqua irão afetar grande ou até mesmo todas as áreas de nossas

vidas, desde atravessar uma rua até se sentar na sala de estar ou entrar em um edifício comercial;

Invisibilidade: os computadores não somente devem estar em todos lugares, como também devem ser imperceptíveis. Com o desenvolvimento de dispositivos de comunicação e computação cada vez menores, esse objetivo parece longe de ser pura ficção científica e começa a se tornar realidade;

Sensibilidade: com o aumento do poder de processamento computacional de dispositivos cada vez menores, os sensores ganharam em precisão das informações obtidas do ambiente onde se encontram. Sensores simples de temperatura, luz ou som foram os sensores mais completos durante muito tempo. Porém, novas gerações de sensores permitirão a obtenção de dados de áudio e vídeo de alta qualidade a partir de máquinas fotográficas e microfones menores que simples botões. Até mesmo os aspectos emocionais de nossas vidas, como estresse, medo, ou excitação, poderiam ser sentidos por sensores embutidos em nossas roupas ou em nosso ambiente;

Amplificação de memória: os avanços no processamento de áudio e vídeo, combinados com os novos equipamentos de sensores fazem perceber a importância de amplificadores de memória que podem continuamente registrar todas as nossas ações, expressões verbais e os nossos movimentos, alimentando-os em um sistema sofisticado que usa o processamento de áudio e vídeo o qual nos permite pesquisar sobre o nosso passado.

A tecnologia de banco de dados e a Internet já deram aos pesquisadores e implementadores um gosto da responsabilidade social que esses sistemas exigem. Lessig (LESSIG, 1999) argumenta que as decisões técnicas feitas durante o projeto de qualquer sistema computacional constituem em implicações legais do que é e o que não é possível de obrigar ou gerenciar em tais sistemas. Com o crescimento e onipresença da *World Wide Web*, as tecnologias computacionais passaram a afetar muito mais que a elite de acadêmicos com habilidades técnicas, pois elas atingem também todos os demais cidadãos.

Conforme (LANGHEINRICH, 2001), a computação ubíqua fará com que as tecnologias computacionais dêem um passo a frente. Com um mundo densamente povoado

de dispositivos de computação e comunicação pequenos, inteligentes e invisíveis, nenhuma única parte da vida dos seres humanos irá escapar da digitalização. Tudo o que se fala, faz ou sente poderá ser digitalizado, armazenado e pesquisado a qualquer momento.

São os cientistas da computação quem precisam entender o potencial e o perigo dos avanços tecnológicos e desenvolver convenções seguras e diretrizes de acordo com princípios bem estabelecidos que irão guiar a tecnologia em uma direção responsável e socialmente aceitável (LANGHEINRICH, 2001).

3.6 Considerações

Nesse capítulo foram apresentados os principais conceitos, tecnologias, preocupações e desafios relacionados à privacidade.

O metamodelo apresentado em (CAMPIOLO, 2005) não permite especificar os problemas relativos à privacidade em ambientes ubíquos. O próximo capítulo abordará uma especificação de privacidade com base nos conceitos de (1) anonimato (PFITZMANN; KöHNTOPP, 2001), (2) o uso de pseudônimos (BERESFORD; STAJANO, 2004), (3) o perfil de preferências do usuário (LEDERER; DEY; MANKOFF, 2002) e (4) a criação de zonas de mixagem (BERESFORD, 2005), no caso de necessidade de existência dessa no ambiente ubíquo.

A figura 4.1 mostra todas as classes construídas em (CAMPIOLO, 2005). Em destaque (retângulo vermelho) são criadas as três classes apropriadas que especificam os itens (1), (2), (3) e (4) focalizados nessa dissertação. Uma especificação formal em Object-Z foi concebida para abordar esses aspectos de privacidade dos itens acima.

4 *Especificando Privacidade*

Esse capítulo apresenta uma extensão da especificação de ambientes de computação ubíqua, criada em (CAMPIOLO, 2005), no sentido de se estabelecer os aspectos relacionados à privacidade. Tais aspectos continuam sendo apresentados, detalhados e discutidos informalmente, através de sentenças explicativas, e formalmente, através do modelo em Object-Z, como feito em (CAMPIOLO, 2005). Além disso, as notações matemáticas como as que podem ser utilizadas no Object-Z são normalmente adotadas, pois descrevem de forma precisa as propriedades de um sistema computacional (DUKE et al., 1991).

4.1 Componentes da Especificação

O desenvolvimento dos componentes da especificação foram baseados em dois objetivos: (1) permitir que a especificação desenvolvida em (CAMPIOLO, 2005) permaneça genérica, porém estendida com a característica de privacidade; (2) permitir a utilização dessa especificação em simulações de ambientes de computação ubíqua, como apresentado no capítulo 5.

O diagrama 4.1 apresenta os principais elementos da especificação classificados em nove categorias. Para sintetizar o entendimento da especificação, apenas as relações mais importantes em (CAMPIOLO, 2005) são apresentadas, com o objetivo de fornecer uma visão geral da interação entre esses elementos.

Como se pode observar, foi utilizada a mesma notação dos diagramas de classe UML (*Unified Modeling Language*), com algumas semânticas redefinidas. Como apresentado por (CAMPIOLO, 2005) são utilizadas as notações de herança, associação, agregação e dependência. As caixas representam as classes essenciais da especificação, herança representa a expansão das classes, a associação representa a

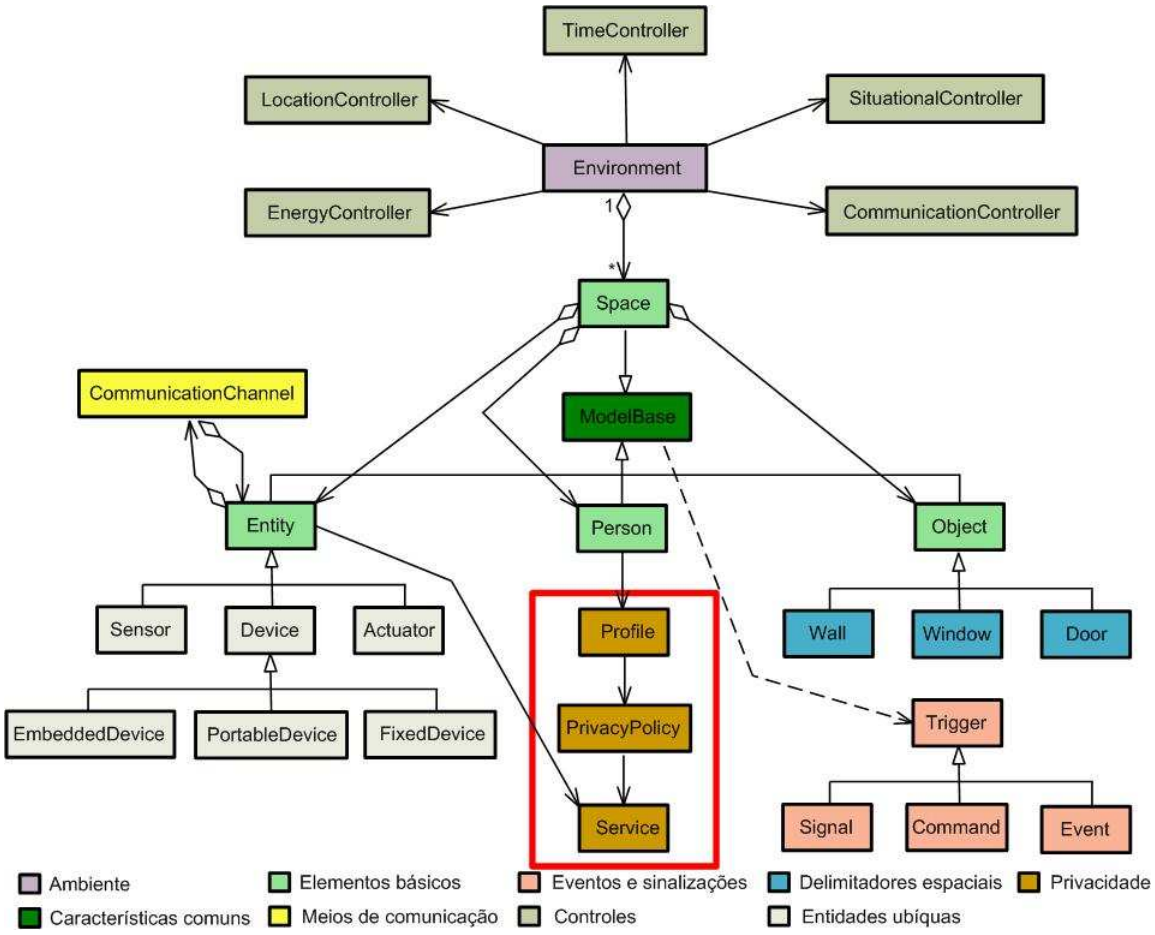

Figura 4.1: Principais componentes da especificação

ligação entre as classes e a dependência indica a participação direta ou indireta sobre o estado das classes.

Os elementos desse metamodelo têm como objetivo permitir a construção de modelos que possam ser simulados através de estruturas computacionais. A categoria de privacidade foi acrescentada, com o objetivo de permitir que ambientes de computação ubíqua sejam simulados com os aspectos de privacidade. As nove categorias são detalhadas a seguir (CAMPIOLO, 2005):

Ambiente: se refere aos cenários de computação ubíqua. Todos os demais elementos fazem parte de sua composição;

Características comuns: definem as propriedades compartilhadas pelos principais elementos físicos que compõem o ambiente;

Elementos básicos: são os elementos físicos presentes nos ambientes;

Meios de comunicação: especificam as comunicações entre as entidades;

Eventos e sinalizações: são responsáveis por um tipo especial de comunicação e transporte de informação entre os elementos;

Controles: são estruturas para tratar questões de gerenciamento de energia, localização, comunicação, tempo e situações na especificação;

Delimitadores espaciais: são objetos que possuem propriedades para definir a delimitação dos espaços físicos;

Entidades ubíquas: correspondem às entidades que possuem capacidade de computação e comunicação

Privacidade: categoria que estende o metamodelo, e que permite aos usuários a criação de políticas de privacidade para que os serviços em ambientes ubíquos não sejam invasivos ou utilizem suas informações inadequadamente. Também permite a criação de perfis de usuários, onde eles especificam as informações que podem ser compartilhadas com outros usuários/serviços.

As próximas seções apresentam as principais classes da especificação apresentada por Campiolo (2005), bem como a especificação completa das classes da categoria de privacidade, propostas neste trabalho. Para a especificação completa feita por Campiolo (2005), recorra ao apêndice A.

4.2 Definições axiomáticas, tipos básicos e tipos livres

Um tipo básico é um recurso da linguagem formal Z que não possui a necessidade de descrever através de estruturas matemáticas ou formais o seu significado. De acordo com (SPIVEY, 1989), o nome de um tipo básico deve ter uma definição global única e seu escopo se estende da definição até o final da especificação. A definição de tipos livres não adiciona nenhum poder à linguagem Z, mas facilita a descrição de estruturas recursivas tais como as listas e árvores. Já as definições axiomáticas permitem especificar tipos através de declarações formais e atribuir restrições aos valores especificados. Esta seção apresenta e descreve os tipos básicos e as definições axiomáticas utilizadas na especificação de privacidade proposto neste trabalho.

Há sete tipos básicos definidos, conforme pode ser observado em (CAMPIOLO, 2005):

$$[TEXT, TIME, DATE, SHAPE, DATA, ACTION, STATE]$$

TEXT: representa um conjunto de caracteres para descrever através de linguagem natural algum tipo de propriedade ou característica, ou para ser utilizado como um identificador;

TIME: corresponde a hora, minutos, segundos e centésimos de segundo;

DATE: corresponde a dia, mês e ano;

SHAPE: tem por objetivo estabelecer limites para forma de propagação e alcance de comunicação, delimitação de acesso, entre outros;

DATA: representa qualquer tipo de informação digital;

ACTION: representa uma ação que modifica o estado de um objeto físico, ou seja, representa a execução de uma tarefa;

STATE: representa os estados que os elementos possuem ou podem assumir.

Foi acrescentado à especificação a definição do tipo livre *MODE*, que define um modo de operação de um serviço em um ambiente ubíquo, como sendo um conjunto definido pelos três valores:

$$MODE ::= allow \mid deny \mid ask$$

Assim, uma política de privacidade pode ser criada, especificando-se um serviço em um dos modos de permissão existentes: permitido (*allow*), negado (*deny*), ou em alguns casos, pode ser questionado (*ask*) se pode ou não ser executado.

Há duas definições axiomáticas globais:

$$\begin{aligned} &Properties : TEXT \nrightarrow DATA \\ &Direction : \{North, South, East, West, Northeast, Northwest, \\ &\qquad Southeast, Southwest\} \end{aligned}$$

O tipo *Properties*, representado por uma função parcial, associa um identificador a uma informação. Por exemplo, pode-se definir uma função f : *Properties*, que

associa nomes com imagens. Já o tipo *Direction* é um conjunto que define vetores para orientação no ambiente. Os vetores definidos são baseados nos pontos cardeais (norte, sul, leste e oeste) e nos pontos colaterais (noroeste, nordeste, sudoeste e sudeste).

4.3 Pessoas

A classe utilizada para especificar pessoas em ambientes ubíquos é a classe *Person*. A figura 4.2 apresenta os principais elementos dessa classe. O tipo *POSITIONAL* representa as posições das pessoas no ambiente, assim como as características que se relacionam com a posição, tais como velocidade, direção e orientação. Já o tipo *PHYSICAL* representa as características físicas como, por exemplo, sexo, idade, peso, entre outros. O tipo *PSYCHOLOGICAL* representa as características relacionadas ao psicológico, tais como emoções, interesses, temperamento, entre outros.

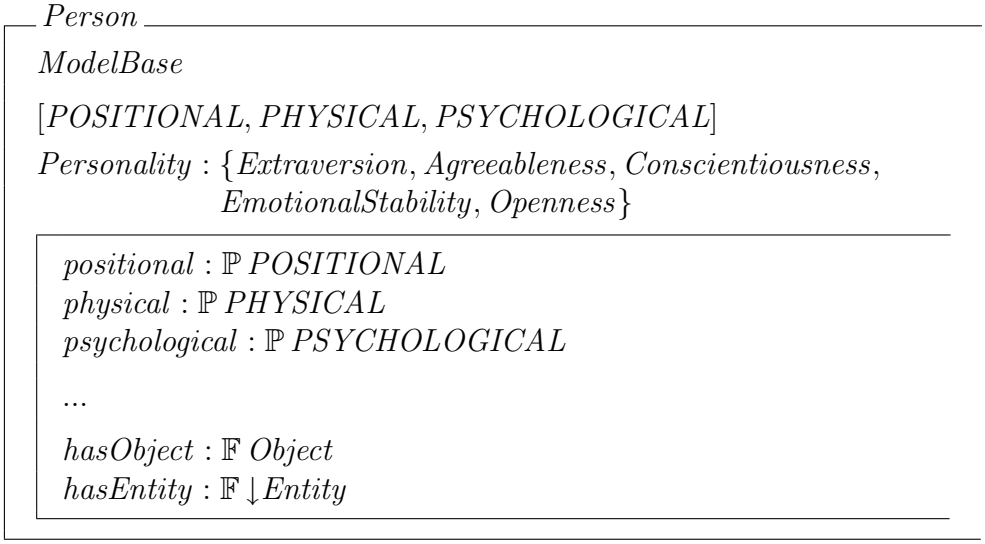

Figura 4.2: Características da classe *Person*

Por fim, as propriedades *hasObject* e *hasEntity* representam o relacionamento das pessoas com os dispositivos e objetos no ambiente.

4.4 Entidades

Na especificação, o termo entidade é utilizado para representar os elementos dos ambientes ubíquos que possuem capacidades de computação e comunicação. A classe *Entity* (figura 4.3) representa a estrutura base para a especificação dos sensores, atuadores e dispositivos.

Entity

$ModelBase$

$[PROCESS, PROTOCOL]$

$enabled : \mathbb{B}$

$connections : \text{seq} \downarrow Entity$
$communication : \text{seq}\, PROTOCOL$
$channel : \text{seq}\, CommunicationChannel$

Figura 4.3: Características da classe *Entity*

A propriedade *enabled* define se a entidade está ativa ou inativa. As propriedades *connections*, *communication* e *channel* especificam, respectivamente, com quais entidades são mantidas conexões, os protocolos e os canais físicos de comunicação.

A classe *Sensor* (figura 4.4) representa os sensores em ambientes ubíquos e herda todas as características da classe *ModelBase* e *Entity*. A propriedade *type* especifica uma identificação numérica para tratar sensores que reagem simultaneamente ao mesmo sinal. As propriedades *perceptionShape* e *perceptionDistance* definem, respectivamente, a forma e a distância que um determinado sensor percebe a presença de um sinal. A propriedade *knownSignal* especifica o tipo de sinal reconhecido e as propriedades *internalState* e *stateTransition* gerenciam a mudança interna de estado originada pela percepção de um sinal.

Os atuadores do ambiente podem ser representados utilizando a classe *Actuator* (figura 4.5). A propriedade *object* define o conjunto de objetos que o atuador tem acesso, enquanto a propriedade *action* define as ações que o atuador pode executar. Por último, a propriedade *actuation* é uma função que associa um comando, recebido por uma conexão, a um objeto e a ação que modifica seu estado.

Sensor

Entity

$type : \mathbb{N}$
$perceptionShape : SHAPE$
$perceptionDistance : \mathbb{N}$
$knownSignal : SignalType$
$internalState : \mathbb{P}\, STATE$
$stateTransition : Signal \nrightarrow (STATE \leftrightarrow STATE)$

$\#knownSignal = 1$

Figura 4.4: Características da classe *Sensor*

Actuator

Entity

$object : \mathbb{P} \downarrow Object$
$action : \mathbb{P}\, ACTION$
$actuation : Command \nrightarrow (Object \leftrightarrow ACTION)$

Figura 4.5: Características da classe *Actuator*

A classe *Device* (figura 4.6) é um modelo genérico de dispositivo que acomoda os principais requisitos para a simulação. Os tipos básicos *INPUT_DEVICE* e *OUTPUT_DEVICE* definem os dispositivos que fornecem entradas e saídas de dados para um dispositivo ubíquo. Como os dispositivos podem ter diversas entradas e saídas, as propriedades *input* e *output* especificam o conjunto de dispositivos de entradas e saídas. As propriedades *entry* e *exit* definem uma associação entre os dispositivos de entrada e saída com o conjunto de dados. A propriedade *aliveProcess* especifica os processos que estão em execução. A propriedade *state* define o conjunto de variáveis de estados e a propriedade *stateTransition* define como as alterações sobre essas variáveis devem ocorrer. A propriedade *event* descreve o conjunto de eventos gerados pelo dispositivo. A propriedade *eventMap* mapeia os eventos para os dispositivos interessados. Assim, é necessário que todo dispositivo que tem interesse em um evento remoto seja registrado ao dispositivo que origina o evento para ser notificado.

As propriedades e funcionalidades da classe *Device* são a base para a definição das seguintes classes de dispositivos: embutidos, portáteis e fixos. A classe *Em-*

Device

Entity

$[INPUT\_DEVICE, OUTPUT\_DEVICE]$
$pid, mid : \mathbb{N}$

$model : TEXT$

$input : \mathbb{P}\, INPUT\_DEVICE$
$output : \mathbb{P}\, OUTPUT\_DEVICE$
$entry : INPUT\_DEVICE \pfun \mathbb{P}\, DATA$
$exit : OUTPUT\_DEVICE \pfun \mathbb{P}\, DATA$

$process : pid \pfun PROCESS$
$aliveProcess : \mathbb{P}(PROCESS)$

$state : \mathbb{P}\, STATE$
$stateTransition : Event \pfun (STATE \rel STATE)$

$memory : \mathbb{P}\, DATA$
$addressMemory : mid \pfun DATA$

$event : \mathbb{P}\, Event$
$eventMap : Event \pfun \mathbb{P}\, Device$

Figura 4.6: Características da classe *Device*

beddedDevice através da propriedade *embeddedIn* especifica onde o dispositivo está embutido. A classe *PortableDevice* através da propriedade *owner* especifica o dono do dispositivo. A classe *FixedDevice* através da propriedade *space* indica o espaço do ambiente no qual a estação fixa se encontra.

4.5 Ambiente

A classe *Environment* (figura 4.7) representa o nível mais alto da hierarquia da especificação e é responsável por agrupar e gerenciar os espaços no modelo.

Um ambiente é identificado pela propriedade *name*. Já a propriedade *space* permite o agrupamento de todos os espaços que compõem o cenário modelado. A condição para a criação de um ambiente, é que ele deve ser composto por, no mínimo, um espaço. O ambiente utiliza os controladores de tempo, situação, localização e energia. O controlador de tempo é responsável por controlar e sincronizar o relógio de simulação e para definir os eventos dependentes do tempo. O controlador de

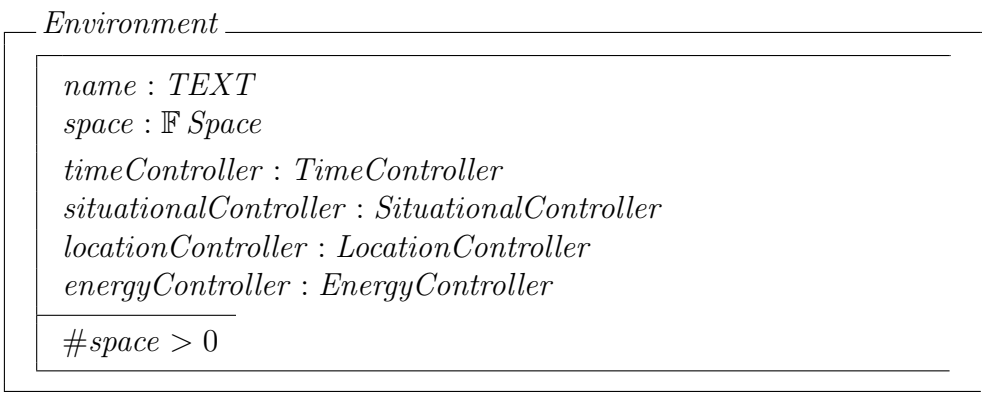

Figura 4.7: Características da classe *Environment*

situação é responsável por registrar situações que dependem de eventos de elementos presentes em espaços comuns ou distintos. Os controladores de localização e energia registram a localização e a energia de todas as entidades para prover com eficiência acesso e controle sobre elas.

4.6 Privacidade

Nesse trabalho, o metamodelo em (CAMPIOLO, 2005) foi estendido e características relativas à privacidade da informação foram incorporadas.

Em ambientes de computação ubíqua fechados[1], a questão da privacidade pode ser garantida internamente por um sistema local. Assim, as violações de privacidade e os problemas originados pela comunicação devem ser protegidos, amenizados e garantidos pelo ambiente.

Partindo desse pressuposto, no presente estudo, são considerados os problemas que envolvem o ambiente e o indivíduo dentro dos limites desse ambiente. Logo, a interação entre dispositivos de diferentes indivíduos no ambiente não é abordada.

Um dos problemas iniciais se refere à comunicação do dispositivo do usuário com os dispositivos do ambiente. Além disso, toda a comunicação entre dispositivos consome energia. Logo, é necessário evitar protocolos desgastantes e tentativas de comunicação repetitivas.

Levando-se em consideração as restrições apresentadas, foram criadas três novas

[1]Ambientes de computação ubíqua fechados são ambientes que são delimitados fisicamente e onde a comunicação e computação se restringem a esses limites.

classes para o metamodelo, as quais são apresentadas na figura 4.8.

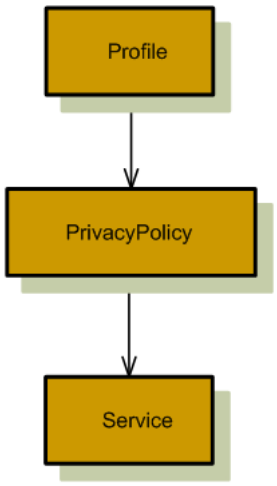

Figura 4.8: Classes da especificação de privacidade

A classe *Service* representa os serviços em ambientes ubíquos e é melhor detalhada na figura 4.9. Esses serviços são fornecidos por dispositivos e sensores e, como discutido na seção 3.5.1, podem ser intrusivos e incômodos, o que pode representar sérias ameaças à privacidade de tais ambientes.

Service

ServiceState

$identification : \mathbb{N}$
$description : TEXT$
$created : TimeStamp$

$\forall x, y : identification \bullet x \neq y$

CreateService

$\Delta(identification, description, created)$
$identification? : \mathbb{N}$
$description? : TEXT$
$created? : TimeStamp$

$identification' = identification?$
$description' = description?$
$created' = created?$

Figura 4.9: Características da classe *Service*

A propriedade *identification* identifica de forma única um serviço no ambiente.

Já a propriedade *description* permite fornecer alguma informação suficiente o bastante para descrever as funcionalidades do serviço. Por último, a propriedade *created* possui o registro da data e hora de criação do serviço.

O que pode ameaçar a privacidade dos indivíduos em ambientes ubíquos são os serviços abusivos, como por exemplo, o envio de anúncios e propagandas que não são de interesse do usuário, monitoramento da localização do usuário, coleta de informações sem permissão, identificação não autorizada, assim como outros exemplos citados na seção 3.5.3.

Para evitar que qualquer serviço tenha acesso às informações pessoais das pessoas de um determinado ambiente ubíquo, foi criada a classe *PrivacyPolicy* (figuras 4.8 e 4.10) para que um indivíduo possa especificar quais serviços podem ter acesso ou não às suas informações.

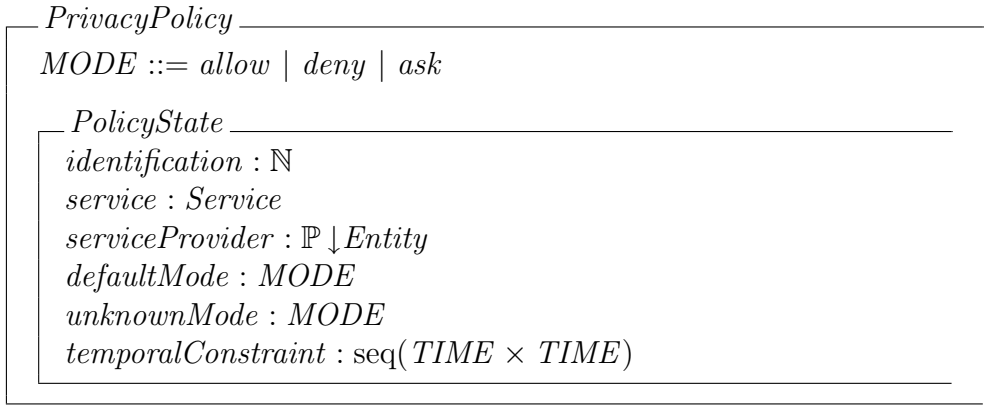

Figura 4.10: Características da classe *PrivacyPolicy*

A propriedade *identification* da classe *PrivacyPolicy* permite especificar unicamente uma política de privacidade no modelo. Para informar sobre qual serviço se trata a política de privacidade, foi criada a propriedade *service*. A propriedade *serviceProvider* permite informar um conjunto parcial de provedores de serviços. Para a execução do serviço provido por esses provedores, a propriedade *defaultMode* deve ser consultada, sendo que seus respectivos modos de operação podem ser: *allow*, *deny* ou *ask* (ver seção 4.2). Caso um provedor não seja relacionado, o modelo permite especificar qual o modo padrão de execução de serviço através da propriedade *unknownMode*. Se o modo for *allow*, todos os serviços desconhecidos terão acesso às informações pessoais; se for *deny*, todos os serviços desconhecidos não terão acesso

e, por último, se for *ask*, o usuário terá que ser consultado sobre o novo provedor de serviços e poderá determinar se deseja ou não compartilhar suas informações. Por último, baseado na seção 3.5.2, foi criada a propriedade *temporalConstraint*, que permite determinar quais os períodos de tempo em que o serviço estará disponível ou não.

Para que o modelo tenha uma especificação mais completa das informações pessoais dos usuários e suas preferências, foi elaborada a classe *Profile*, figuras 4.8 e 4.11.

Profile

ProfileState

$identification : \mathbb{N}$
$pseudonym : TEXT$
$personalProperties : \mathbb{P}(TEXT \times \mathbb{F}\ TEXT)$
$servicePolicies : \mathbb{P}\ PrivacyPolicy$
$mixingRate : \mathbb{N}$

$\forall x, y : pseudonym \bullet x \neq y$

Figura 4.11: Características da classe *Profile*

A propriedade *identification* da classe *Profile* identifica de forma única um perfil de um usuário. Utilizando-se do conceito de mixagem, apresentado na seção 3.5.6.3, a propriedade *pseudonym* armazena o pseudônimo usado para identificar o dispositivo do usuário e permitir a mixagem. A propriedade *personalProperties* permite especificar as preferências pessoais, tais como, na categoria de esportes: futebol, vôlei, basquete, entre outros; e na categoria de filmes: de ação, comédia, romance, entre outros. A propriedade *servicePolicies* relaciona o perfil do usuário com as suas políticas de privacidade, ou seja, o conjunto de serviços que podem ter ou não acesso às informações do seu perfil pessoal. Por fim, a propriedade *mixingRate* define uma taxa mínima de indivíduos que devem estar presentes para que ocorra uma mixagem. Se a taxa for zero, significa que o usuário não quer participar da mixagem.

Além dessas considerações, para integrar os aspectos de privacidade à especificação feita por (CAMPIOLO, 2005), foi criada a propriedade *profile* na classe *Person*, que permite associar a classe *Profile* à uma pessoa; assim como a propriedade *offeredServices* na classe *Entity*, que permite informar quais serviços são fornecidos

por uma determinada entidade.

4.7 Considerações

Esse capítulo apresentou os principais elementos para representação de um ambiente de computação ubíqua, com a preocupação de preservar a privacidade das pessoas contidas nesses ambientes. A especificação de privacidade é limitada aos ambientes de computação ubíqua fechados, mas pode ser estendido para os demais tipos de ambientes, desde que seja levada em consideração a complexidade do ambiente que deseja-se estudar.

Com a elaboração da especificação, os conceitos de privacidade foram melhor discutidos, esclarecidos e entendidos. Dessa forma, o próximo capítulo apresenta a aplicação da especificação obtida no ambiente de um shopping center.

5 *Prova de Conceito: Cenário de um Shopping Center*

Esse capítulo apresenta a aplicação do metamodelo apresentado no capítulo 4 para o cenário de um shopping center. A escolha deve-se ao fato de um shopping center ser um cenário complexo, por ser composto por diversos outros cenários fechados e abertos (lojas, áreas de exposição, salões, escadas rolantes, etc.), mas que estão bem delimitados fisicamente pela estrutura do shopping.

O objetivo é destacar a importância e aplicabilidade do metamodelo elaborado, bem como discutir os diversos problemas eminentes desse cenário, além de propor algumas soluções viáveis. Apenas as principais classes são apresentadas nesse capítulo. Para obter a aplicação completa, é necessário consultar o apêndice C. Além disso, ao final do capítulo é apresentado o cenário do shopping modelado na ferramenta Opnet e uma análise dos resultados é feita com base na simulação obtida.

5.1 O Cenário

Conforme previamente apresentado em (CAMPIOLO, 2005), o shopping center (figura 5.1) corresponde a um cenário composto por uma grande quantidade de pessoas com seus dispositivos e por diversos espaços fechados e abertos, bem delimitados fisicamente pela estrutura do shopping. Há sensores e dispositivos distribuídos, monitorando e fornecendo serviços para os indivíduos nos limites internos do ambiente. Os serviços têm como objetivo conquistar e fornecer comodidade aos usuários. Para os serviços não serem invasivos, isto é, não cometerem abusos, o ambiente dispõe de uma infra-estrutura para proteger a privacidade de seus usuários.

Os dispositivos e sensores podem estar localizados no espaço de lojas. Eles podem se comunicar internamente na loja, isto é, o dispositivo de um cliente é

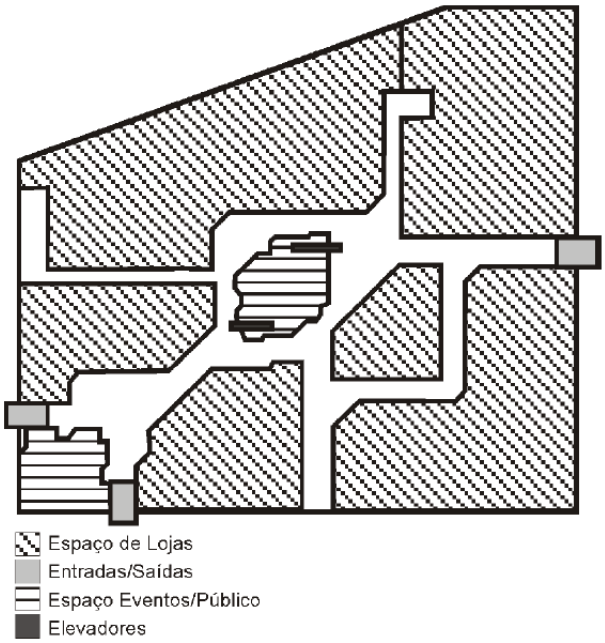

Figura 5.1: Ilustração do cenário shopping. Fonte: (CAMPIOLO, 2005)

detectado dentro dos limites da loja, ou ainda se comunicar a uma determinada distância externa à loja. O mesmo princípio é válido para os sensores. Além disso, sensores e dispositivos podem ser de propriedade do shopping e estarem distribuídos em outros pontos, sendo compartilhados por diversas lojas.

5.2 Problema específico

O cenário a seguir ilustra um problema específico de privacidade no ambiente de estudo. Alice possui um perfil que é composto de duas partes: um conjunto de proposições fornecido por Alice (perfil A) e um conjunto de proposições inferido pelo sistema ou por outras entidades (perfil B).

Alice pode determinar de que forma o perfil A pode ser utilizado pelo sistema, no todo ou para um propósito específico. Já sobre o perfil B, seu controle é bastante limitado, pois Alice pode nem saber de sua existência. Por exemplo, considere que Alice prepara o perfil A indicando que gosta de ginástica, filmes de ação e livros de auto-ajuda, conforme mostra a figura 5.2. A estrutura nomeada **alice**, representa a especificação da classe *Person* que instancia Alice no ambiente. Nesse caso, a identificação **alice** foi utilizada como valor da propriedade *pseudonym*, pois Alice optou por não participar de mixagem (propriedade *mixingRate* é zero), ou seja, não

deseja manter seu anonimato. Esse assunto volta a ser discutido na seção 5.6.

Profile : *profile_A*

$identification = 14$
$pseudonym = alice$
$personalProperties = \{sports, \{gymnastics\}\}, \{films, \{action\_movies\}\}, \{books, \{self\_help\}\}$
$servicePolicies = marketing\_policy$
$mixingRate = 0$

Figura 5.2: Perfil A fornecido por Alice

A figura 5.3 apresenta a política de privacidade de Alice, onde ela determina que suas informações utilizadas pelo serviço de propaganda (**marketing_service**), não devem ser repassados a terceiros, apenas para as livrarias L e M. Além disso, através da propriedade *temporalConstraint*, Alice definiu uma restrição temporal alegando que deseja ser abordada por esses serviços somente no horário das 8h às 18h.

PrivacyPolicy : *marketing_policy*

$identification = 16$
$service = marketing\_service$
$serviceProvider = sen\_bookstore\_L, sen\_bookstore\_M$
$defaultMode = allow$
$unknownMode = deny$
$temporalConstraint : (8h00m00s, 18h00m00s)$

Figura 5.3: Política de privacidade para o serviço de propaganda

Agora, suponha que Alice realize algumas compras de livros sobre viagens na livraria L, uma ou mais vezes por mês num período de seis meses. Pode-se supor que esses livros são para ela mesma, pois é difícil que sejam presentes (a não ser que Alice conheça vários amigos que gostem de viajar e façam aniversário dentro deste período). Assim, esta livraria tem condições de criar um segundo perfil para Alice, o perfil B da figura 5.4, que é uma cópia do perfil A fornecido por ela, com um valor a mais na propriedade *personalProperties*, dizendo que ela gosta de livros de viagens, fato que ela não revelou pessoalmente.

É importante considerar que esta informação é produzida sem o conhecimento

Profile : *profile_B*

$identification = 15$
$pseudonym = alice$
$personalProperties = \{sports, \{gymnastics\}\}, \{films, \{action\_movies\}\},$
$\{books, \{self\_help, traveling\}\}$
$servicePolicies = marketing\_policy$
$mixingRate = 0$

Figura 5.4: Perfil B inferido pelo sistema

de Alice, não se enquadra na restrição de não ser fornecida a terceiros e constitui um perfil B ainda mais preciso do que o perfil A fornecido pessoalmente por ela.

Agora, suponha que Alice vai ao shopping portando um *handheld* (estrutura **handheld**) com a sua identificação não ambígua e seu perfil A armazenado. Já na entrada, Alice é identificada, o perfil A é lido e o perfil B é resgatado pelo sistema.

Ao passar próxima à livraria M, Alice recebe uma mensagem informando de algumas promoções de livros de viagens. Isso pode ser abordado de vários pontos de vista. Alice não colocou a informação de que gosta de livros de viagens no perfil A, porque prefere pesquisar pessoalmente quando tem uma necessidade específica.

Suponha que ela viaje muito a trabalho, mas não pode determinar seu destino com antecedência. Neste caso, não adianta receber avisos de promoções. Alice sabe que as informações colocadas em seu perfil A são utilizadas para que diversas empresas possam fazer ofertas personalizadas. Por outro lado, essas ofertas podem ser interessantes para Alice, pois podem ser mais de acordo com seus interesses, além de proporcionar uma boa economia em alguns casos.

De qualquer modo, o objetivo das empresas é vender, o que necessariamente não significa satisfazer Alice completamente, nem respeitar sua vontade.

5.3 Modelo e arquitetura do ambiente

A estrutura física do ambiente permanece idêntica a apresentada na figura 5.1. São adicionados apenas novos sensores e dispositivos em algumas áreas do ambiente.

Nas entradas e saídas do ambiente, encontram-se os sensores responsáveis por

coletar as políticas de privacidade do usuário. Estes sensores são chamados de sensores E/S. Um exemplo desse tipo de sensor na especificação é o **senIO**. Somente os sensores E/S recolhem as políticas de privacidade. Logo, necessita-se de uma entidade para armazenar essas políticas.

A entidade responsável por armazenar as políticas de segurança é o servidor central **server**. Este servidor recebe dos sensores E/S as informações coletadas do dispositivo do usuário. Para simplificar o entendimento, essas informações são representadas na propriedade *memory*. Os sensores e dispositivos locais acessam as políticas de privacidade de um cliente através de uma conexão com o servidor central. Assim, os protocolos de comunicação com os dispositivos tornam-se menos desgastantes e economizam a bateria do dispositivo do cliente. Os sensores **sen_bookstore_L** e **sen_bookstore_M** são os sensores locais das livrarias L e M, respectivamente, que possuem essa responsabilidade.

O mesmo efeito é alcançado com a coleta do arquivo de políticas de privacidade na entrada. A coleta é realizada por comunicação sem fio (rádio). A distância entre o dispositivo coletor e o dispositivo do cliente é pequena. Logo, a energia despendida e a perda de pacotes são bem menores.

Os sensores E/S devem ser isolados e ter um alcance de extensão, ou seja, não devem permitir um sensor externo recuperar ou atrapalhar a coleta das políticas de privacidade e devem garantir que o dispositivo do cliente permaneça na faixa de transmissão até o término do protocolo.

O último elemento de arquitetura adicionado são as zonas de mixagem (seção 3.5.6.3), onde nenhum dos usuários pode ser localizado por serviços. No ambiente estudado, as zonas de mixagem encontram-se na região central e nos ambientes de exposição do shopping (figura 5.5), pois, nesses locais, há uma constante movimentação de pessoas.

5.4 Modelo da política de privacidade

Neste modelo, o foco são os serviços disponibilizados. Cada serviço é representado por uma marca no arquivo e contém as ações padrões, negar ou permitir o serviço, para os estabelecimentos comerciais listados, além de uma ação para lojas não listadas.

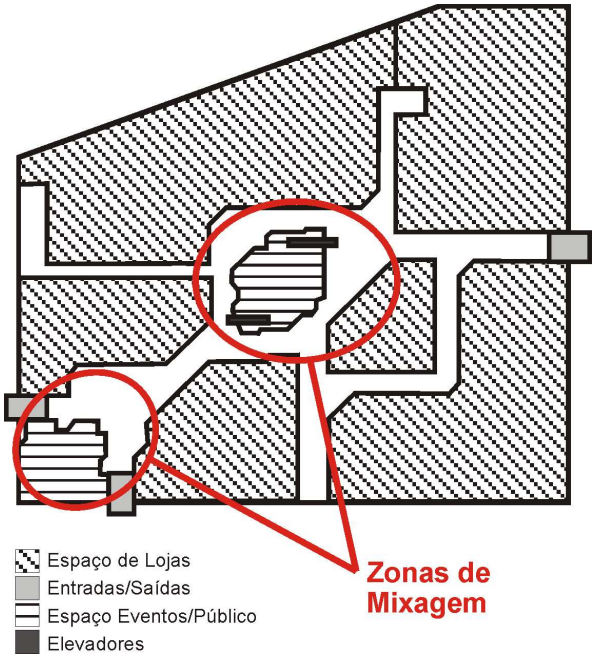

Figura 5.5: Ilustração do cenário shopping. Fonte: (CAMPIOLO, 2005)

Um exemplo de um serviço que é permitido a duas lojas é apresentado na figura 5.6. Caso alguma outra tente fornecer o serviço, o usuário poderá ser consultado se deseja ou não, tal serviço. No segundo serviço (figura 5.7), o usuário já configurou que o serviço seja negado para qualquer outra loja não listada.

A figura 5.8 apresenta a situação onde o usuário deseja sempre receber um serviço. Já, para o caso contrário, a figura 5.9 apresenta a situação onde o usuário nunca deseja receber um determinado serviço.

A próxima seção apresenta o funcionamento dos protocolos de comunicação utilizados no ambiente de estudo.

$PrivacyPolicy : Example1$

$identification = 1$
$service = service\_1$
$serviceProvider = store\_1, store\_2$
$defaultMode = allow$
$unknownMode = ask$

Figura 5.6: Política de privacidade que dá permissão de um serviço à duas lojas, permitindo que para as lojas desconhecidas o usuário seja consultado

PrivacyPolicy : *Example*2

identification = 2
service = *service_2*
serviceProvider = *store_1*, *store_2*
defaultMode = *allow*
unknownMode = *deny*

Figura 5.7: Política de privacidade que dá permissão de um serviço à duas lojas, mas que não dá permissão à nenhuma outra

PrivacyPolicy : *Example*3

identification = 3
service = *service_3*
unknownMode = *allow*

Figura 5.8: Política de privacidade que dá sempre permissão de um serviço à qualquer loja

PrivacyPolicy : *Example*4

identification = 4
service = *service_4*
unknownMode = *deny*

Figura 5.9: Política de privacidade que não dá acesso de nenhum serviço à nenhuma loja

5.5 Entendendo os protocolos

Para facilitar a compreensão, as implicações e o funcionamento dos protocolos, estes são descritos baseados nas possíveis situações que um cliente encontra ao entrar em um ambiente de computação ubíqua fechado.

5.5.1 Entrada de um cliente/usuário

Um cliente, ao entrar no ambiente portando seu dispositivo de computação ubíqua, inicia a comunicação com os sensores E/S. O arquivo de política de privacidade e a identificação do cliente são transmitidos. O servidor gera um pseudônimo

para o registro do cliente, o qual funciona como uma chave para recuperação das políticas. O dispositivo do cliente recebe essa identificação única e a armazena no campo correspondente. A comunicação é encerrada.

5.5.2 Detecção de um cliente por um sensor

Ao detectarem a presença de um dispositivo, os sensores do ambiente ou estabelecimento, através de um protocolo simples, obtêm o seu pseudônimo. Após isso, consultam o servidor central para determinar as políticas de privacidade do usuário portador do dispositivo. Se estiver registrado o interesse do usuário por algum serviço fornecido, inicia-se um protocolo para atendê-lo.

5.5.3 Cliente finaliza a comunicação com um dispositivo

Um cliente, ao sair de um estabelecimento ou do alcance de um sensor ou dispositivo, finaliza seus vínculos e a marca de mixagem é redefinida. Logo, ao transitar por uma zona de mixagem, este dispositivo receberá um novo pseudônimo e sua marca de mixagem será novamente redefinida.

5.6 Aplicando o modelo a um problema

Nesta seção, o problema da Alice (seção 5.2) é retomado e as situações descritas no problema são aplicadas ao modelo proposto. A geração do perfil B, em que é registrado o interesse de Alice por livros de viagens, é inevitável, uma vez que para a realização das compras, Alice se identifica, seja por meio do crédito, ou outro qualquer, tal como a lembrança do vendedor ou até mesmo sistemas de reconhecimento facial usado nas câmeras de segurança da loja.

Dada a inevitabilidade da geração de um perfil B, a solução para se manter a privacidade de Alice é dissociá-la de seu perfil B. Isto pode ser feito atribuindo a Alice um pseudônimo que será usado para identificá-la. Assim, ao passar pelos sensores da loja, Alice não será identificada e, conseqüentemente, não ligada ao seu perfil B. Partindo da idéia de que Alice deseja ter anonimato, o novo perfil A de Alice com o seu novo pseudônimo gerado automaticamente pelo servidor central é apresentado na figura 5.10.

Profile : *profile_A*

$identification = 14$
$pseudonym = 4fasd452$
$personalProperties = \{sports, \{gymnastics\}\}, \{films, \{action\_movies\}\}, \{books, \{self\_help\}\}$
$servicePolicies = marketing\_policy$
$mixingRate = 1$

Figura 5.10: Perfil A fornecido por Alice

Entretanto, há situações em que Alice deverá se identificar, como por exemplo, ao realizar um pagamento com cartão de crédito. Neste momento, os sistemas podem localizar Alice, uma vez que ela foi relacionada com o pseudônimo usado. A solução é trocar o seu pseudônimo novamente.

Obrigar um usuário a sair e entrar novamente no ambiente visando a mudança do pseudônimo não é algo plausível. Para isso, existem as zonas de mixagem, em que Alice pode literalmente se misturar a multidão, emergindo desta zona com um novo pseudônimo não rastreável pelos estabelecimentos comerciais.

Assim, mesmo nos casos em que a identidade de Alice possa ser inferida pelos seus hábitos (hábitos como a visita específica a determinadas lojas numa ordem em um determinado horário), basta que Alice passe por uma zona de mixagem para o seu anonimato permanecer.

No caso do shopping, as zonas de mixagem são pontos centrais em que o usuário sempre transita, seja para mudar de nível, seja para ver outras lojas. Assim, as chances de rastreamento e identificação passam a ser muito baixas, salvo nas ocasiões em que as taxas de mixagem forem muito baixas.

5.7 Simulando o cenário

O modelo elaborado para tratar das questões de privacidade em ambientes de computação ubíqua fechados apresenta uma solução, baseada em restrições de serviço através de políticas de privacidade definidas pelo usuário, satisfatória para solucionar os problemas de invasão da privacidade causados por serviços oferecidos em tais ambientes.

O modelo da arquitetura e de dados desenvolvido atendeu aos requisitos, evitando tráfego de pacotes desnecessários e desperdício de bateria dos dispositivos do cliente.

A discussão das questões de privacidade permitiu uma reflexão sobre quais as futuras preocupações e prevenções que os usuários e as aplicações devem considerar para a utilização de dispositivos e ambientes de computação ubíqua.

Essa seção apresenta uma simulação realizada para esse cenário de estudo, comprovando a eficácia da solução apresentada.

5.7.1 O simulador Opnet

O Opnet (*Optimized Network Engineering Tool*) *Modeler*, ou simplesmente Opnet, foi introduzido em 1986 pelo MIT e permite projetar e estudar redes de comunicação, dispositivos, protocolos e aplicação.

Os modelos no Opnet são hierárquicos. Como mostra a figura 5.11, no nível mais baixo, o comportamento de um algoritmo ou protocolo é codificado por um diagrama de estados finitos com código embutido baseado na linguagem C/C++. No nível intermediário, funções discretas como processamento, transmissão e recepção de pacotes de dados são executados por objetos separados, os quais se comportam como definido no seu modelo de processo. Esses objetos, chamados módulos, são conectados para formar o modelo da rede, que na hierarquia, é o modelo de mais alto nível. Esse modelo, por fim, é o que define o escopo de uma simulação.

Todos os componentes são modelados utilizando a abordagem orientada a objeto, o que facilita o mapeamento para os sistemas reais. Também possui uma plataforma flexível para testar novas soluções de redes e com baixo custo.

A ferramenta Opnet, se comparada com outras como NS2, se destaca pela facilidade de uso, qualidade da documentação e pelas plataformas de simulação. Além disso, o Opnet possui maior suporte às tecnologias de redes sem fio, o que satisfaz as necessidades para a simulação do ambiente do shopping.

O Opnet é bem aceito no meio acadêmico devido à confiança nos resultados que oferece aos seus usuários e também porque permite uma análise realística de métricas de desempenho (RAZAK; ZHOU; LANG, 2002).

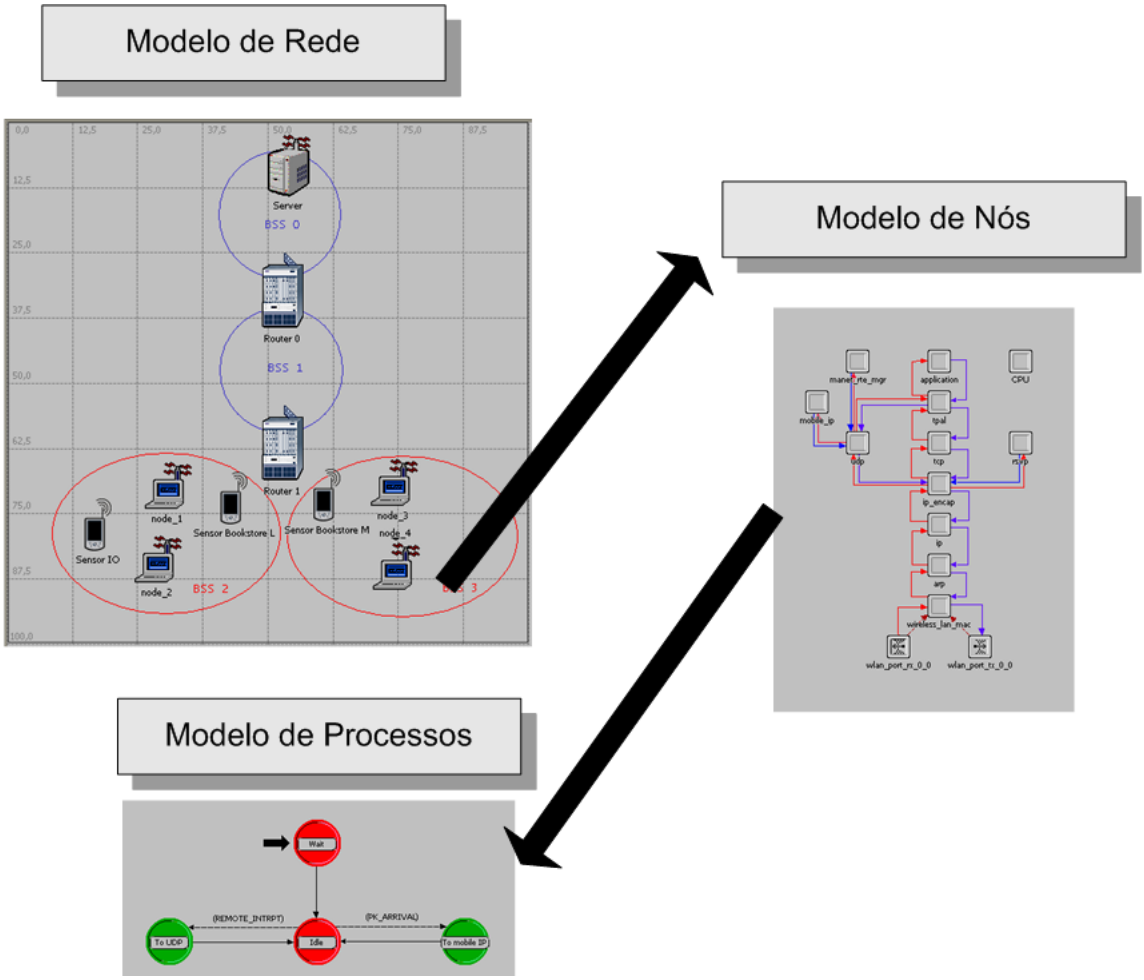

Figura 5.11: A estrutura hierárquica dos modelos no Opnet

Uma das características mais importantes do Opnet é o fato de utilizar modelos que são especificados em termos de objetos, os quais possuem um conjunto de atributos configuráveis permitindo assim, a definição de novos objetos com características programáveis e fácil extensão. Essa característica foi fundamental para a criação dos objetos utilizados no cenário do shopping. Além disso, permite que o próprio usuário defina qualquer tipo de formato de pacote para que seja usado em novos protocolos. Assim, foi possível criar os pacotes do cenário do shopping e os protocolos de comunicação.

As principais características do Opnet podem ser resumidas em (CHANG, 1999):

Ciclo de Modelagem e Simulação: o Opnet provê ferramentas para ajudar o usuário a construir modelos, executar a simulação e analisar os resultados, conforme mostra a figura 5.12;

Modelagem hierárquica: o Opnet emprega uma estrutura hierárquica para modelagem. Cada nível da hierarquia descreve os diferentes aspectos do modelo completo a ser simulado;

Especializado em redes de comunicação: modelos de bibliotecas detalhadas provêem o suporte para protocolos existentes e permitem aos usuários e desenvolvedores modificar estes modelos existentes ou desenvolver novos modelos;

Geração de simulação automática: os modelos do Opnet podem ser compilados em código executável. O núcleo do Opnet consiste em códigos C/C++, permitindo que uma simulação de evento discreto possa ser depurada ou simplesmente executada, gerando dados para serem analisados.

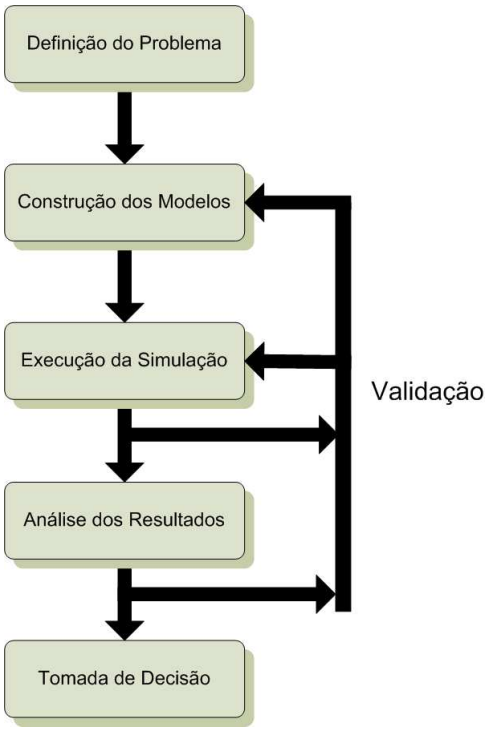

Figura 5.12: Ciclo de Modelagem e Simulação do Opnet baseada em (CHANG, 1999)

Por fim, é importante destacar que o Opnet não é um software de código aberto, porém, permite que muitos parâmetros de simulação sejam alterados, tendo um efeito significante na simulação. O Opnet *Modeler* é apenas uma das muitas ferramentas do conjunto de tecnologias Opnet e a versão utilizada nessa dissertação é a 11.5.

5.7.2 O ambiente representado no Opnet

Para a simulação foi considerado um ambiente simplificado do shopping descrito na seção 5.1. No ambiente simulado há duas livrarias, representadas pelos sensores **Bookstore L Sensor** e **Bookstore M Sensor**; 3 cenários: o primeiro com 50 usuários, o segundo com 100 usuários e o terceiro com 300 usuários os quais são representados pelos seus dispositivos móveis; o servidor central ou **Server**; uma zona de mixagem representada pelo **Mix Sensor** e um sensor E/S que é o **IO Sensor**. Esses elementos são apresentados no modelo de rede do Opnet, na figura 5.13.

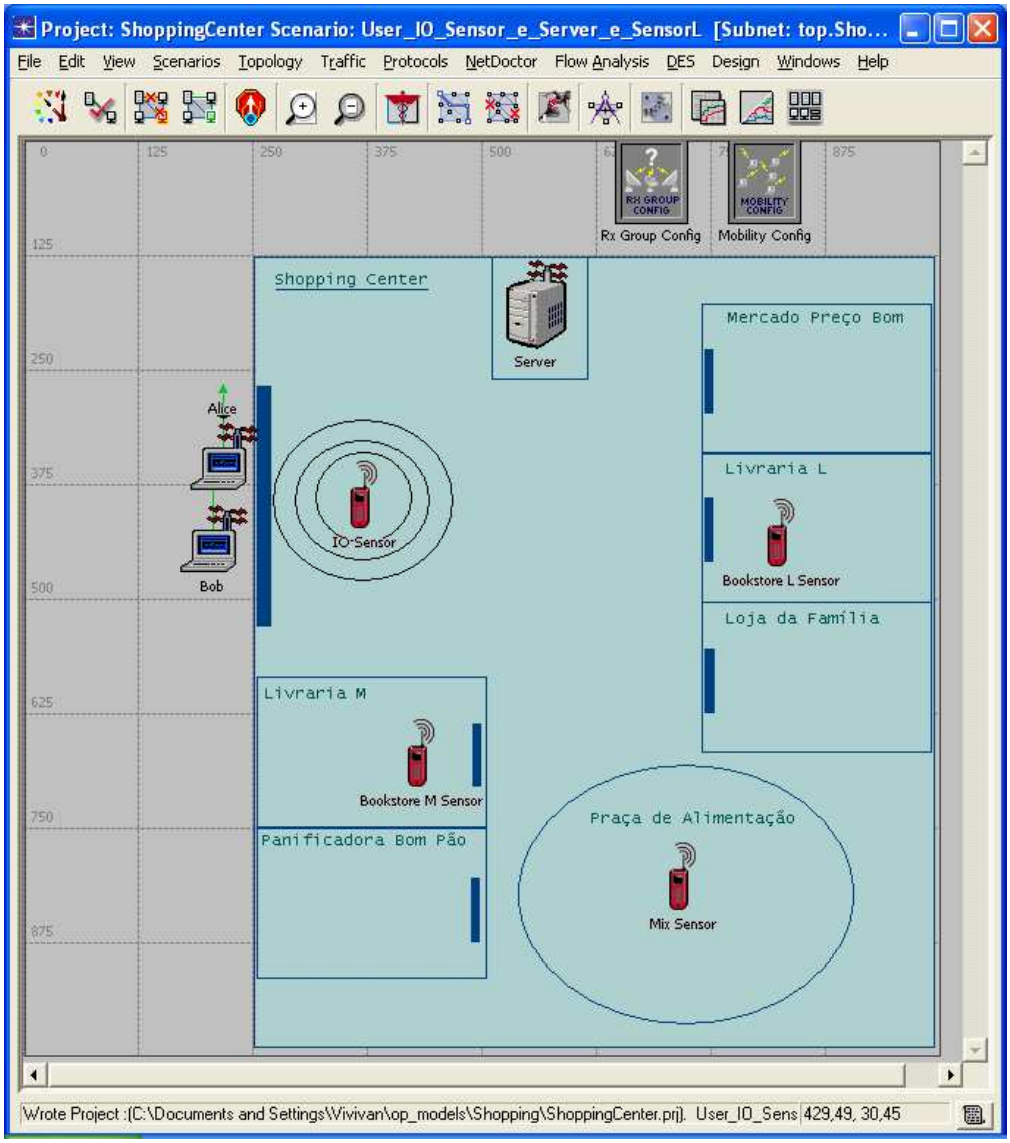

Figura 5.13: Cenário do shopping no Opnet.

A seguir é descrita a forma da troca de mensagens bem como os módulos dos elementos do shopping.

5.7.2.1 Pacotes para troca de mensagens

Para a troca de mensagens entre os elementos do shopping foram desenvolvidos os seguintes pacotes:

- shopping_sensor_requests_pck ou pacote de requisição;
- shopping_profile_pck ou pacote de perfil;
- shopping_pseudonym_pck ou pacote de pseudônimo;
- shopping_new_pseudonym_pck ou pacote de pseudônimo novo, criado pelo sensor de mixagem e útil apenas para os usuários que desejarem participar da mixagem;
- shopping_service_policies_of_user_pck ou pacote de políticas de serviço do usuário;
- shopping_mixing_pck ou pacote de mixagem;
- shopping_marketing_pck ou pacote de propaganda.

A seguir, cada um desses pacotes e seus campos são apresentados e melhor detalhados.

Pacote de requisição

O pacote de requisição é utilizado pelos sensores das livrarias, pelo sensor de mixagem e pelo sensor E/S para fazer as solicitações ao usuário. Como mostra a figura 5.14, esse pacote possui três campos: **description**, **IP source** e **text**.

O campo **description** poderá ter um dos seguintes valores:

- *Profile*: quando a requisição for feita pelo sensor E/S, indicando a solicitação do perfil do usuário;
- *Pseudonym*: quando a requisição for feita pelo sensor de alguma livraria, indicando a solicitação do pseudônimo do usuário;

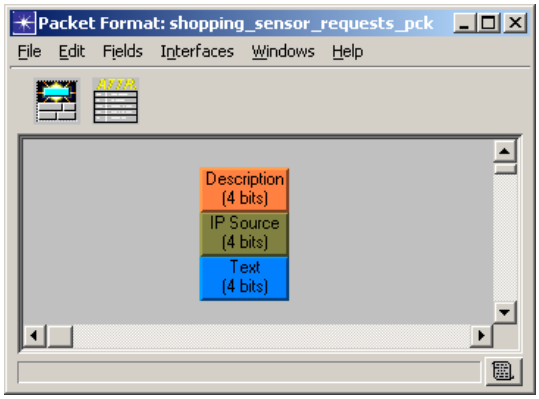

Figura 5.14: Pacote de requisição

- *Mix*: quando a requisição for feita pelo sensor de mixagem, indicando a solicitação da indicação do usuário sobre participar ou não da mixagem;
- *New Pseudonym*: quando a requisição for feita pelo sensor de mixagem ao servidor, indicando a solicitação da geração de um novo pseudônimo. Nesse caso, no campo **Text** é enviado o pseudônimo atual do usuário.

Já o campo **IP source** serve para armazenar o IP que originou a requisição para que o usuário ou o servidor possam enviar a resposta.

Pacote de perfil

Quando o usuário recebe um pacote de requisição do tipo *Profile*, envia imediatamente ao sensor E/S um pacote com o seu perfil. Esse pacote é então recebido pelo sensor E/S e encaminhado ao servidor, que o armazena para futuras consultas pelos sensores das livrarias.

De acordo com a figura 5.15, o pacote perfil possui os seguintes campos: **IP user**, **pseudonym**, **personalProperties**, **servicePolicies**, **mixingRate**, **name**, **lastName**, **IP source** e **IP destination**.

Os campos **IP user** e **pseudonym** possuem os valores do IP do dispositivo do usuário e o seu pseudônimo, respectivamente. O campo **personalProperties**, figura 5.16, é uma lista, onde cada elemento possui os campos **type** e **value_list**. O campo **type** indica a categoria da propriedade pessoal como, por exemplo: esportes, filmes e livros. Já o campo **value_list** é um campo complementar ao campo **type**

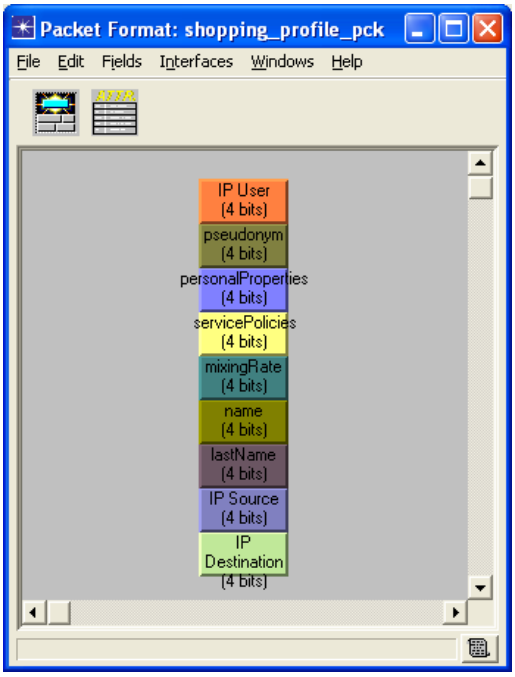

Figura 5.15: Pacote de perfil

e possui os valores daquela categoria. Por exemplo, na categoria esportes pode-se ter: ginástica, vôlei e futebol.

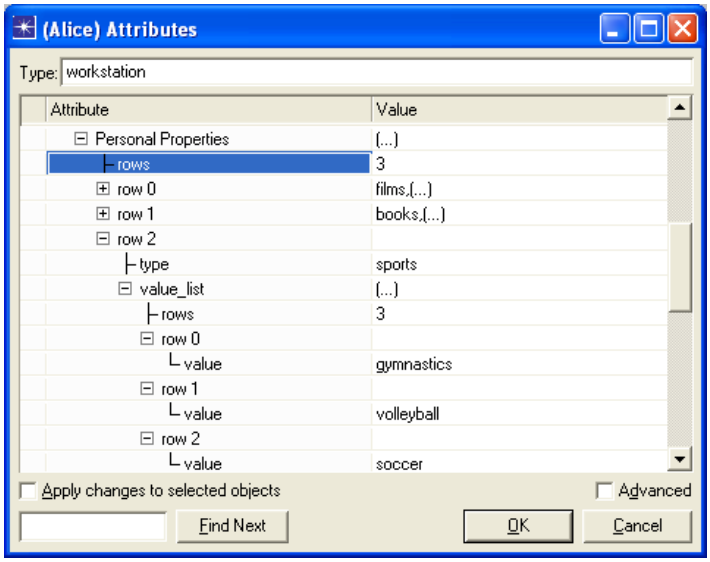

Figura 5.16: Campo **personalProperties** do pacote perfil.

O campo **servicePolicies** do pacote perfil possui as políticas de acesso dos serviços de interesse do usuário ou políticas de privacidade. Assim como o campo **personalProperties**, esse também é uma lista. Cada elemento dessa lista possui os seguintes atributos: **identification**, **service**, **service_provider_list**,

default_mode, **unknown_mode**, **hour_ini**, **min_ini**, **hour_end** e **min_end**. O campo **identification** serve para identificar unicamente a política de privacidade. O campo **service** é uma descrição para o serviço oferecido pela loja como, por exemplo, o serviço de propaganda. Já **service_provider_list** é uma lista que identifica quais os provedores do serviço que têm permissão para o usuário em questão. Os campos **default_mode** e **unknown_mode** definem qual a permissão padrão e qual o nível de acesso permitido para provedores desconhecidos, respectivamente. Para esses dois campos os valores possíveis são: *allow*, *deny* e *ask*, conforme apresentado na seção 4.6. Além disso, os campos **hour_ini**, **min_ini**, **hour_end** e **min_end** permitem que o usuário determine em qual horário ele deseja ser abordado pelo provedor de serviço. Por exemplo, no horário comercial seria das 8h às 18h.

O campo **mixingRate** permite que o usuário determine se deseja ou não participar da mixagem ao entrar em uma zona de mixagem. Caso este campo esteja como zero, significa que ele não deseja fazer parte da mixagem.

Por fim, os campos **name**, **lastName**, **IP source** e **IP destination** identificam o nome e sobrenome do usuário, IP origem e destino do pacote perfil que será enviado.

Pacote de pseudônimo

Ao entrar no shopping o dispositivo do usuário é abordado pelo sensor E/S, o qual solicita o seu perfil. Recebendo esse perfil, o sensor envia ao servidor o perfil do usuário para que seja armazenado em sua lista interna de usuários. Nesse momento, o servidor gera um pseudônimo para o usuário que é enviado ao sensor E/S e o próprio sensor se encarrega de encaminhar o pseudônimo ao usuário.

Ao passar próximo à uma livraria, o sensor dessa livraria inicia uma comunicação com o usuário para obter o seu pseudônimo. Após recebido o pacote com o pseudônimo do usuário, o sensor envia-o ao servidor que o encaminha as políticas de privacidade do usuário. Se for verificado que o sensor possui permissão para enviar propagandas ao usuário, esse serviço é executado.

Conforme apresenta a figura 5.17, o pacote de pseudônimo possui o campo **pseudonym**, que identifica o pseudônimo do usuário; o campo **IP user**, que indica sempre o IP do usuário, utilizado apenas pelo servidor, quando este gera o pseudônimo do usuário e envia ao sensor E/S; e os campos **IP source** e **IP destination**, que são,

respectivamente, o IP de origem e destino do pacote. O dispositivo do cliente nunca preenche o campo **IP source**, apenas os campos **pseudonym** e **IP destination**.

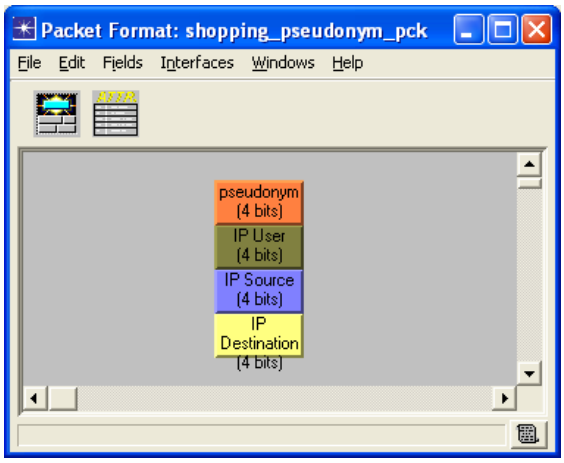

Figura 5.17: Pacote de pseudônimo.

Pacote de pseudônimo novo

O pacote de pseudônimo novo, figura 5.18, é utilizado pelo servidor e pelo sensor de mixagem. Ao passar próximo ao sensor de mixagem, o usuário é questionado sobre o seu interesse em participar ou não da mixagem. Caso essa verificação seja positiva, o sensor de mixagem solicita ao servidor um novo pseudônimo para o cliente. O servidor então gera um novo pseudônimo, atualiza-o na sua lista de usuários para o usuário em questão e envia o novo pseudônimo gerado ao sensor de mixagem. O sensor de mixagem então encaminha o novo pseudônimo ao cliente, campo **new pseudonym**, que é identificado pelo pseudônimo antigo, campo **pseudonym**. Além disso, possui o campo **IP source** para identificar quem enviou o pacote.

Pacote de mixagem

Quando o sensor de mixagem detecta um usuário novo, esse faz uma requisição ao usuário para que envie o pacote de mixagem, apresentado na figura 5.19. Esse pacote terá o valor da taxa de mixagem, indicando se o usuário deseja ou não participar da mixagem. Se o campo **mixing** for zero, indica que o usuário não deseja participar da zona de mixagem. Caso tenha o valor 1, indica que o usuário deseja ter o seu pseudônimo alterado, ao entrar em uma zona de mixagem.

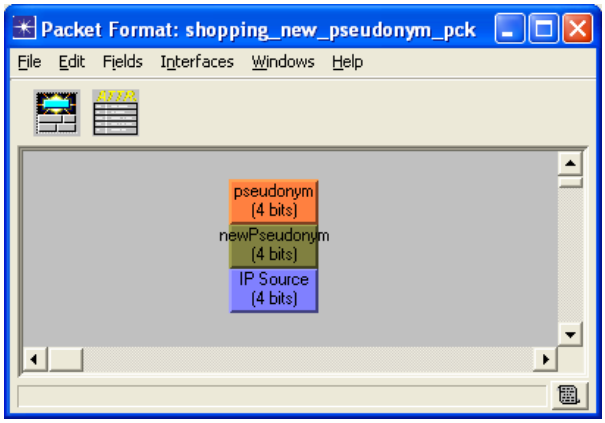

Figura 5.18: Pacote de pseudônimo novo.

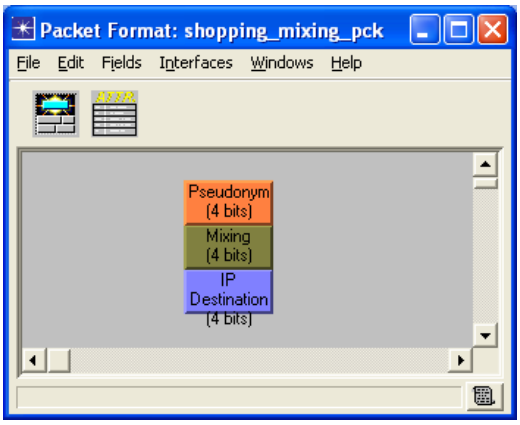

Figura 5.19: Pacote de mixagem.

O campo **pseudonym** indica o pseudônimo do usuário e o campo **IP destination** indica o IP destino do pacote.

Pacote de políticas de serviço do usuário

O pacote de políticas de serviço do usuário, apresentado na figura 5.20, é o pacote que é enviado do servidor ao sensor de uma livraria, para que este possa identificar quais as preferências pessoais do usuário com o objetivo de prestar um serviço mais personalizado a ele, de acordo com os seus interesses. Vale destacar que, este pacote só é enviado ao sensor, caso o servidor verifique nas políticas de privacidade do usuário que o sensor possui permissão para prestar serviços para aquele usuário.

O campo **pseudonym** identifica o pseudônimo do usuário. Os campos **servicePolicies** e **personalProperties** têm a mesma estrutura apresentada para esses

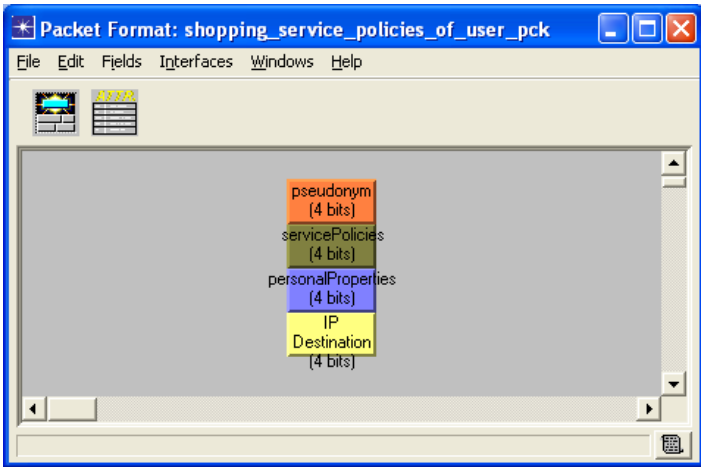

Figura 5.20: Pacote de políticas de serviço do usuário.

mesmos campos no pacote perfil. Por fim, o campo **IP destination** indica para qual sensor o pacote será enviado.

Pacote de propaganda

Após verificada as preferências pessoais do usuário, o pacote de propaganda, figura 5.21, é enviado com as propagandas que são do interesse do usuário. O usuário é identificado pelo campo **pseudonym** e a lista de produtos pelo campo **productList**.

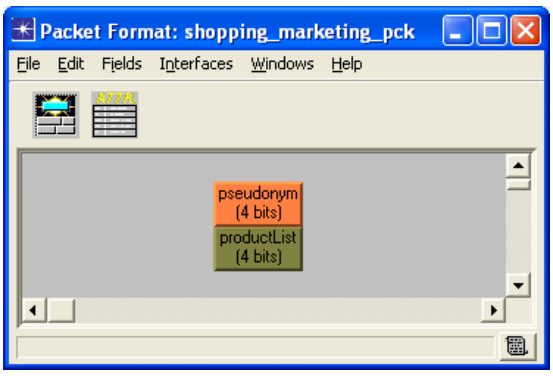

Figura 5.21: Pacote de propaganda.

O campo **productList** é uma lista com os campos **name** e **price**, que identificam respectivamente, o nome e o preço do produto.

5.7.2.2 Modelo do nó sensor E/S

Como apresentado na figura 5.11, cada nó de rede representado no Opnet consta de um *modelo de nó* e de um *modelo de processo*. O modelo do nó sensor E/S é apresentado na figura 5.22. Como pode-se perceber há dois fluxos de entrada (**stream 0** e **stream 1**) e um fluxo de saída.

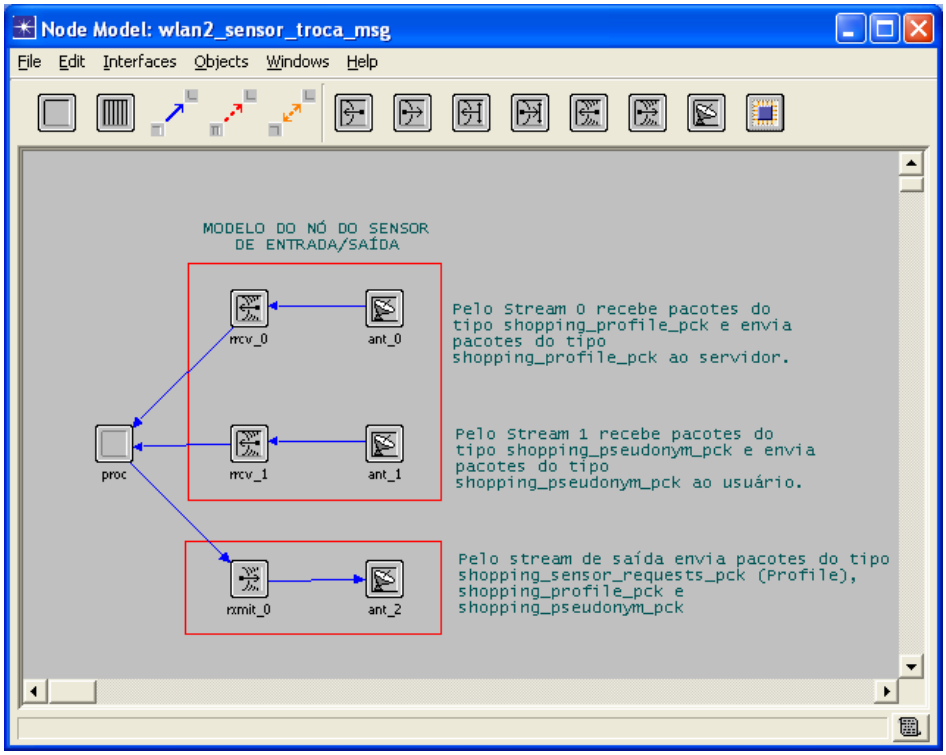

Figura 5.22: Modelo do nó sensor E/S.

O nó sensor é o responsável por capturar as informações dos usuários e encaminhá-las ao servidor, e vice-versa. O nó sensor envia periodicamente pacotes de requisição ou **shopping_sensor_requests_pck** do tipo **Profile** aos novos usuários do shopping para dar início à troca de mensagens. Caso algum usuário responda à essas requisições, é pelo fluxo 0 que o sensor irá receber os pacotes do tipo **shopping_profile_pck** do usuário e os enviará pelo fluxo de saída ao servidor. O servidor então gerará um pseudônimo para esse usuário, e enviará ao sensor E/S. É pelo fluxo 1 que o sensor receberá os pacotes do tipo **shopping_pseudonym_pck** do servidor e os enviará pelo fluxo de saída ao usuário.

A figura 5.23 apresenta o modelo de processo do sensor E/S. Há quatro estados: **start**, **send**, **wait** e **end**. O estado **start** carrega a variável de estrutura **Address**

(figura 5.24) com os valores IP informados na interface do nó, exibida na figura 5.25. Como pode-se observar, o nó sensor E/S, assim como todos os outros sensores do ambiente, trabalha com duas BSS: 0 e 1. A BSS 1 é para comunicação com o usuário e a BSS 0 é para a comunicação entre os sensores e o servidor.

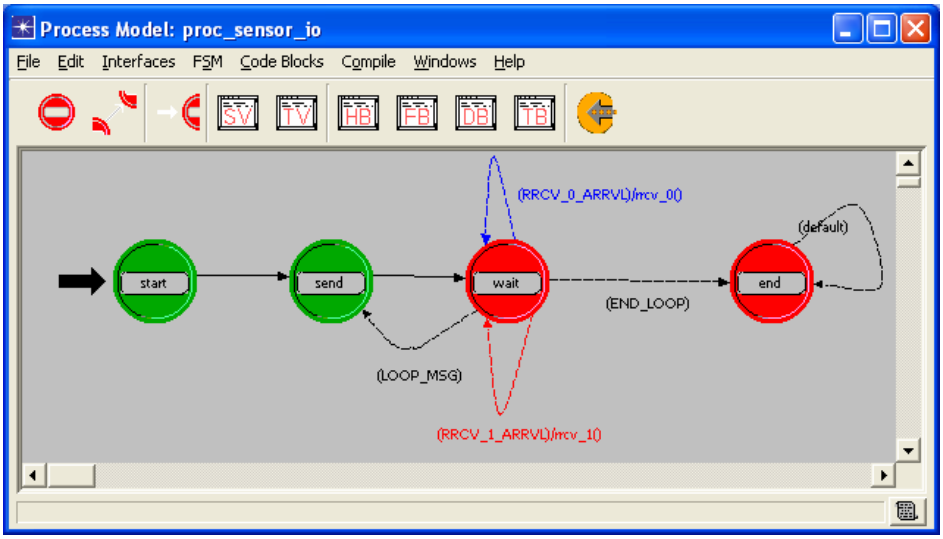

Figura 5.23: Modelo de processo do nó sensor E/S.

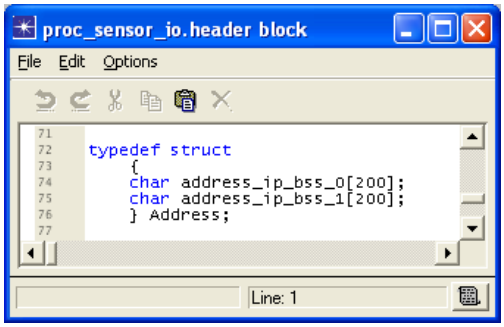

Figura 5.24: Estrutura **Address**.

O estado **send** é o que envia pacotes de requisição aos novos usuários do shopping, que são os pacotes **shopping_sensor_requests_pck** do tipo **Profile**, como pode ser visto na figura 5.26.

Após enviar essa requisição, a máquina de estados assume o estado **wait**. Nesse estado o sensor aguarda pacotes do usuário (fluxo 0) ou do servidor (fluxo 1). Esse estado também pode voltar ao estado **send** para o envio de novas solicitações (transição **LOOP_MSG**) ou ir para o estado final **end** com a transição **END_LOOP**.

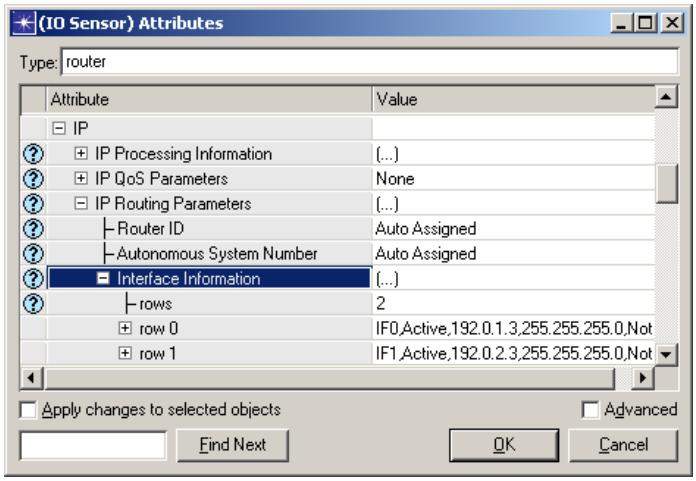

Figura 5.25: Atributos do nó sensor E/S.

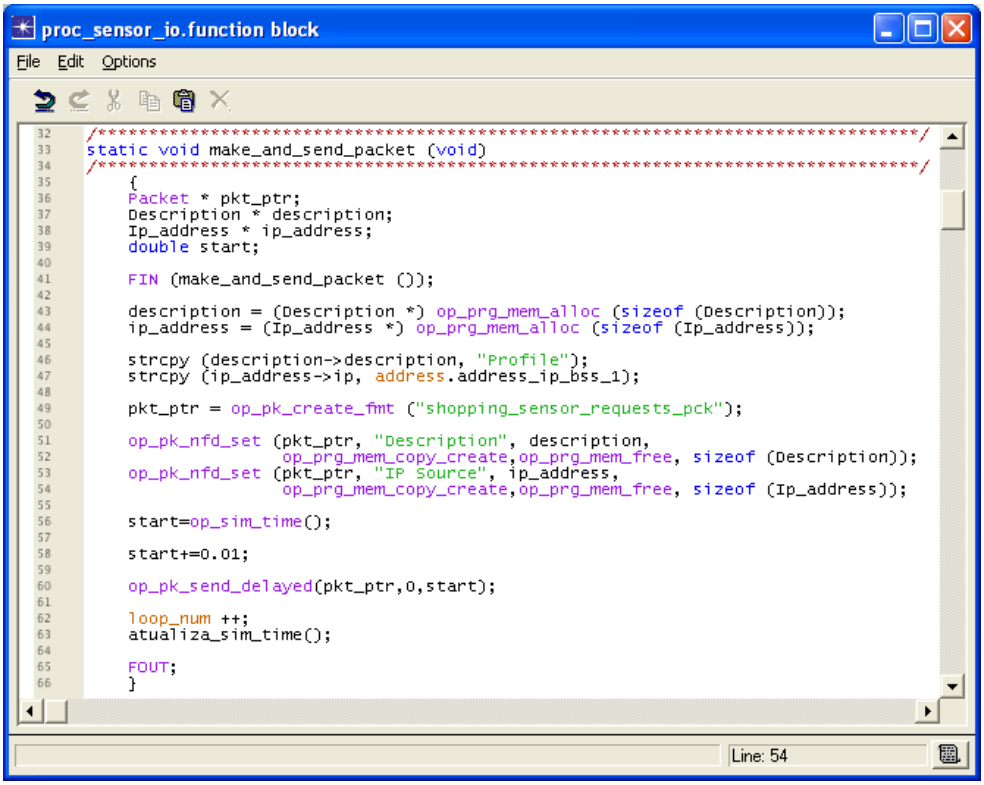

Figura 5.26: Função **make_and_send_packet()** do processo **send** do sensor IO.

5.7.2.3 Modelo do nó de usuários

O modelo de nó de usuários é o modelo mais importante do cenário de estudo, pois representa o elemento chave da dissertação, que é o usuário e como manter a sua privacidade. Como mostra a figura 5.27, o dispositivo do usuário tem seis fluxos de entrada (**stream 0**, **stream 1**, **stream 2**, **stream 3**, **stream 4** e **stream 5**) e

um fluxo de saída.

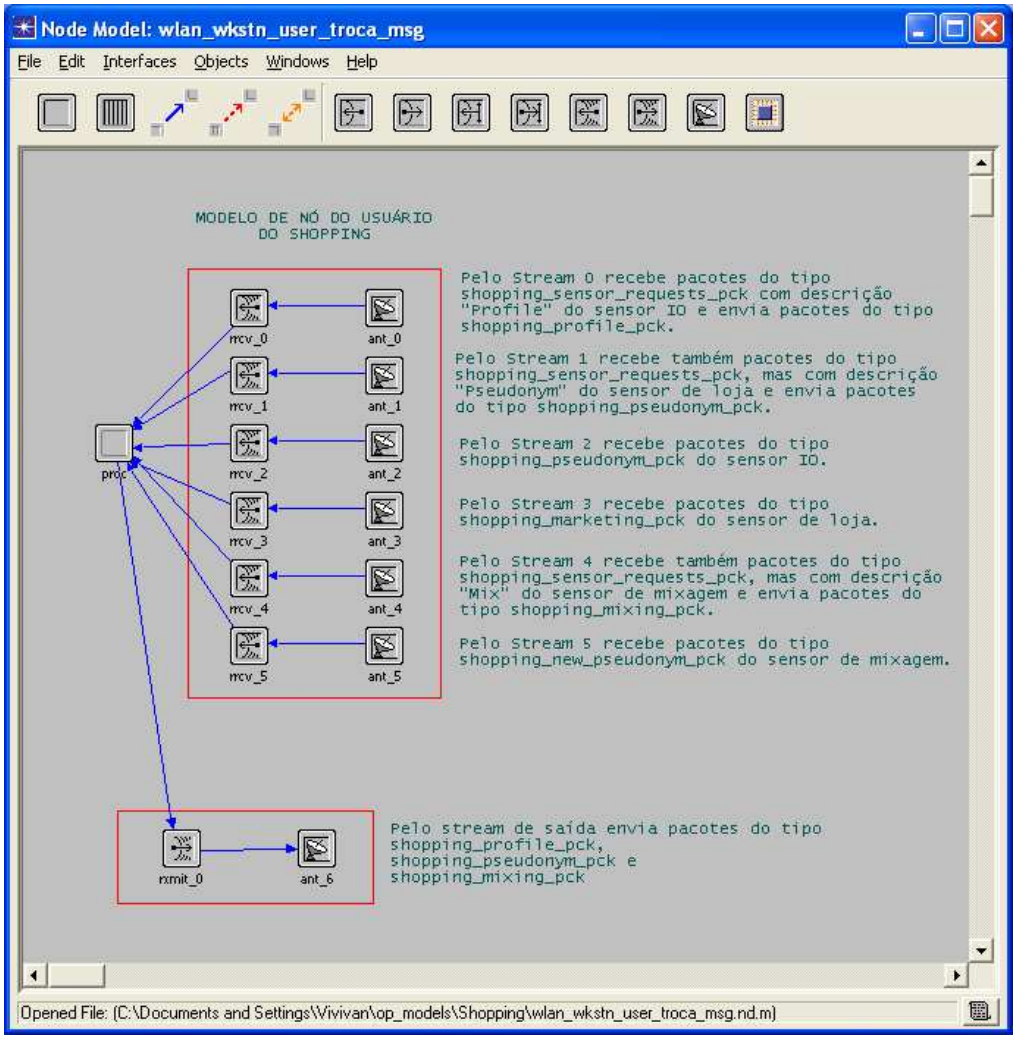

Figura 5.27: Modelo do nó de usuários.

Pelos fluxos 0, 1 e 4, o usuário recebe pacotes de requisição, que são os pacotes do tipo **shopping_sensor_requests_pck**, do sensor E/S, sensor de livraria e sensor de mixagem, respectivamente. Após receber o pacote de requisição do sensor E/S, o usuário envia o pacote com seu perfil **shopping_profile_pck** ao sensor, para que possa receber um pseudônimo. Como já explicado na seção 5.7.2.2, o sensor envia esse pacote ao servidor, que gera um pseudônimo para o usuário, e que é enviado pelo sensor E/S. O pseudônimo é recebido pelo usuário por meio do fluxo 2. Ao receber o pseudônimo, o dispositivo do usuário grava em sua memória interna o valor para que possa ser utilizado em outras comunicações durante sua estadia no shopping.

Ao receber o pseudônimo, o usuário fica apto a responder as solicitações dos

sensores das livrarias, que são as solicitações feitas pelo fluxo 1. Feito isso, o sensor da livraria irá consultar o servidor para verificar se tem permissão para enviar propagandas ao usuário e para saber quais as preferências do usuário. Se for possível o envio de propagandas, o usuário as receberá pelo fluxo 3, e se houver alguma promoção que seja do seu interesse, com certeza irá até a livraria.

Caso o usuário caminhe próximo à uma zona de mixagem, ele receberá pelo fluxo 4 uma requisição sobre a sua taxa de mixagem para que o sensor da zona de mixagem determine se o usuário deseja participar ou não da mixagem. O valor da sua taxa de mixagem é enviado no pacote **shopping_mixing_pck**. Caso o valor da taxa de mixagem seja 1, o sensor irá solicitar ao servidor um novo pseudônimo que é então enviado ao usuário. Esse novo pseudônimo é recebido pelo fluxo 5 no pacote **shopping_new_pseudonym_pck**. Isso faz com que qualquer perfil paralelo gerado pelas lojas seja perdido e apenas o perfil gerado pelo usuário seja respeitado, mantendo a sua privacidade.

Além do modelo de nó do usuário, há também o modelo de processo, mostrado na figura 5.28. Como pode-se observar, há apenas dois estados: **start** e **wait**. O estado **start** é o que carrega internamente o endereço IP do usuário e o estado **wait** é o estado que aguarda as comunicações em todos os fluxos descritos acima.

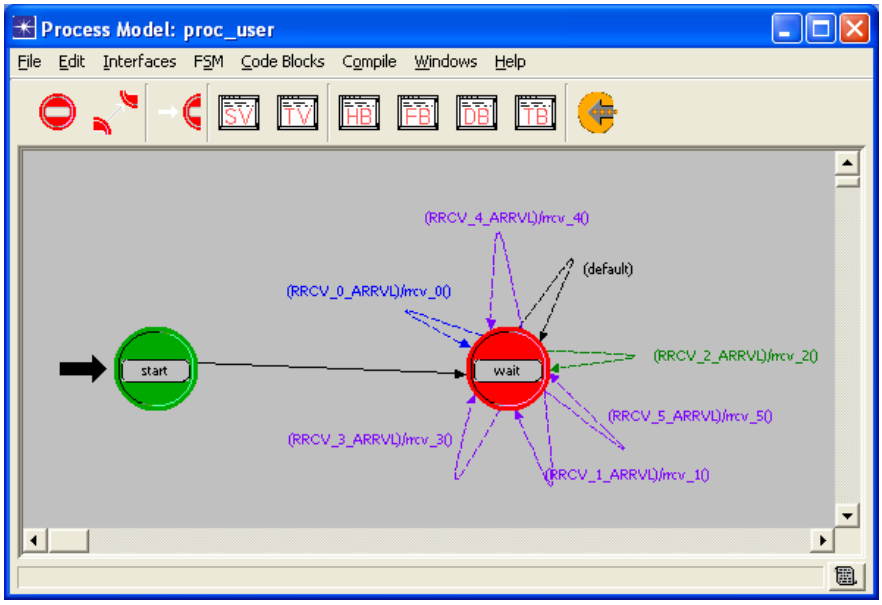

Figura 5.28: Modelo de processo do nó de usuários.

Para exemplificar, caso seja cumprida a condição **RRCV_1_ARRVL**, que é defina no *header block* como mostra a figura 5.29, a função **rrcv_1()** é executada.

Como explicado acima, nesse fluxo é solicitado ao usuário o seu pseudônimo. A figura 5.30 apresenta o código da função **rrcv_1()**.

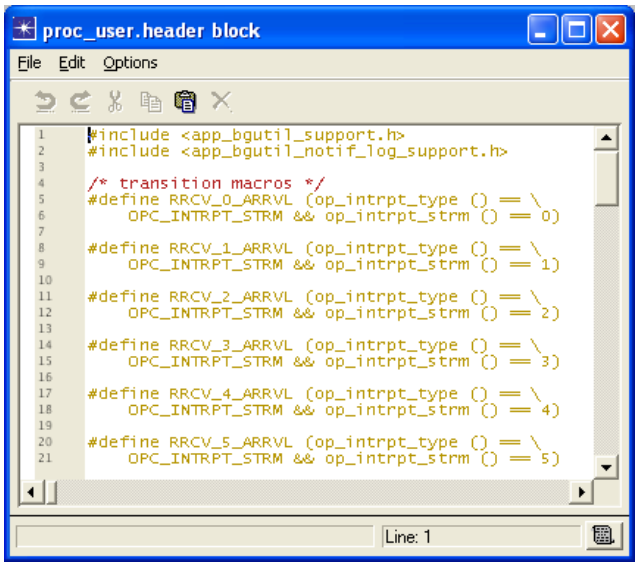

Figura 5.29: *Header Block* do processo de usuário, onde são definidas as condições das transições de estado.

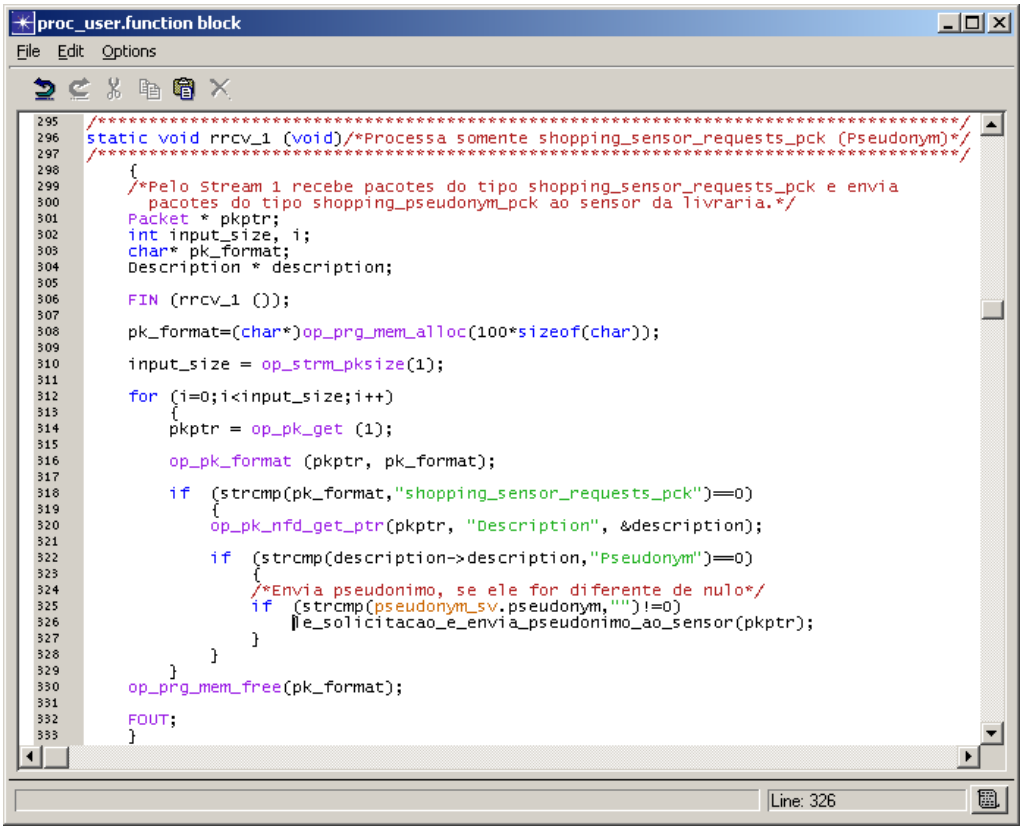

Figura 5.30: Definição da função **rrcv_1()**.

Na linha 322, o usuário verifica se a requisição do pacote é do tipo **Pseudonym** e

se for, a função **le_solicitacao_e_envia_pseudonimo_ao_sensor()** (figura 5.31) é chamada. Nessa função, na linha 351 é criado o pacote **shopping_pseudonym_pck** que é carregado com o valor do pseudônimo do usuário e com o IP destino do pacote, por meio da função **op_pk_nfd_set** do Opnet. Após isso, na linha 360, o pacote é enviado com o comando **op_pk_send_delayed**.

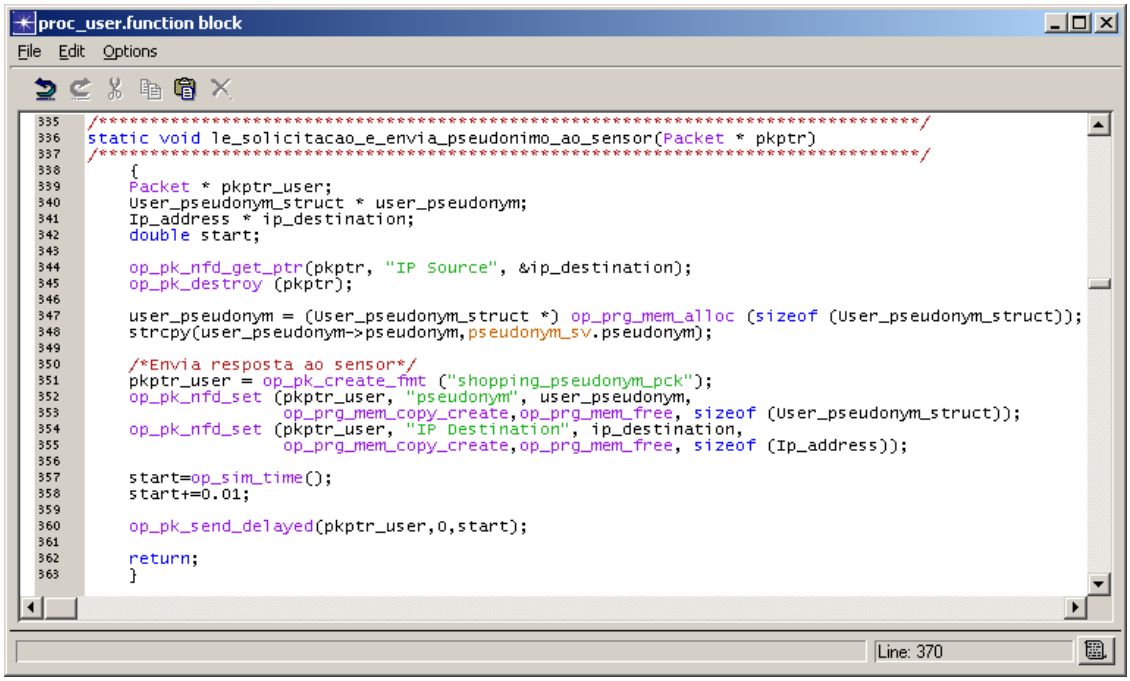

Figura 5.31: Definição da função **le_solicitacao_e_envia_pseudonimo_ao_sensor()**.

5.7.2.4 Modelo do nó servidor

O nó servidor é o responsável por armazenar o perfil do usuário e por gerar um pseudônimo para esse usuário, assim que ele entra no shopping. A figura 5.32 apresenta o modelo do nó servidor, que consta de três fluxos de entrada (**stream 0**, **stream 1**, e **stream 2**) e um de saída.

Pelo fluxo 0, o servidor recebe do sensor E/S o perfil do usuário no pacote **shopping_profile_pck** e o armazena em sua lista de usuários, a qual é apresentada na figura 5.33.

Após o armazenamento do usuário, o servidor gera o seu pseudônimo, armazena-o no registro do usuário e envia-o ao usuário por meio do sensor E/S no pacote **shopping_pseudonym_pck**.

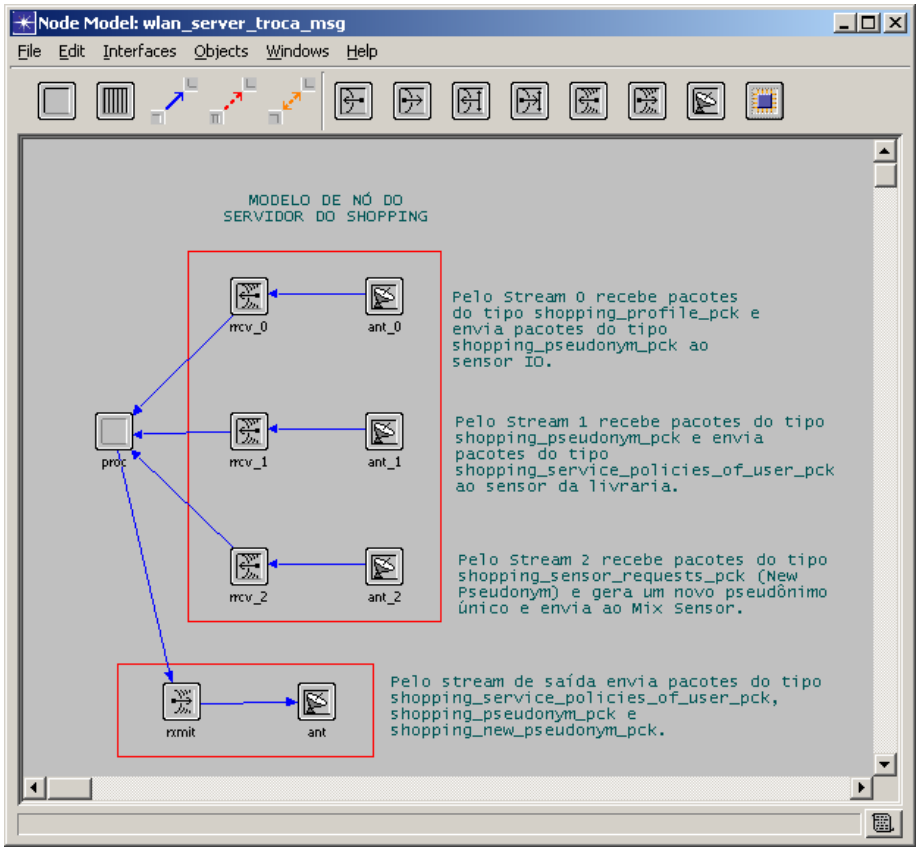

Figura 5.32: Modelo do nó servidor.

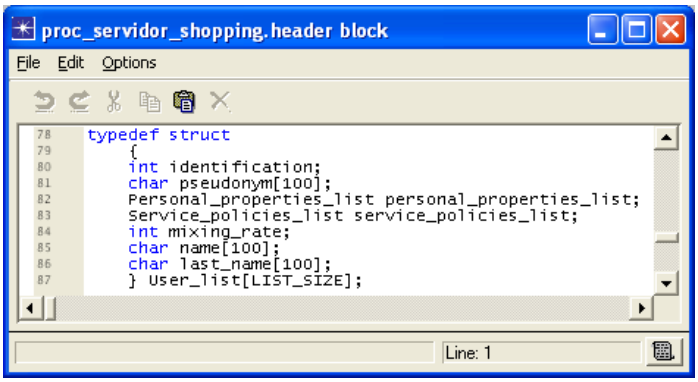

Figura 5.33: *Header block* do nó servidor, onde é definida a sua lista de usuários.

O fluxo 1 é para as comunicações entre os sensores das livrarias com o servidor central. O sensor da livraria envia para o servidor o pseudônimo do usuário, para que ele localize as suas preferências pessoais, políticas de privacidade e verifique se o sensor tem permissão de enviar propagandas ao usuário. Caso ele tenha permissão de enviar propagandas ao usuário, o servidor envia as preferências pessoais e as políticas de privacidade do usuário no pacote **shopping_service_policies_of_user_pck**.

Já o fluxo 2 serve para o servidor receber as requisições do sensor de mixagem para geração de novos pseudônimos para os clientes do shopping que desejam participar da mixagem. Nesse caso, o pacote recebido é o **shopping_sensor_requests_pck** do tipo **New Pseudonym**, e o servidor envia o novo pseudônimo ao sensor de mixagem no pacote **shopping_new_pseudonym_pck**.

O modelo de processo do nó servidor é mostrado na figura 5.34. Assim como o processo do usuário, o processo do servidor possui dois estados: **start** e **wait**. No estado **start** a variável de IP do servidor é carregada e no estado **wait** o servidor aguarda por pacotes para estabelecer uma comunicação com os sensores.

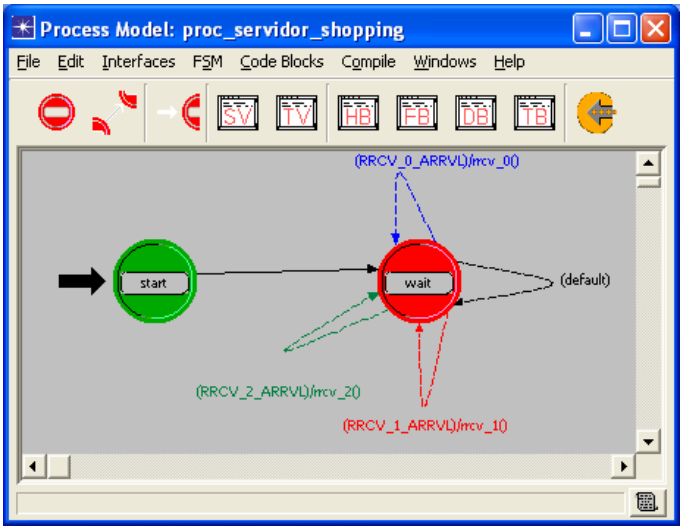

Figura 5.34: Modelo de processo do nó servidor.

A transição com condição **RRCV_0_ARRVL** verifica se chegou algum pacote para o fluxo 0. Nesse caso, a função **rrcv_0()** é acionada. Essa função se encarrega de fazer a leitura do perfil do usuário e armazená-la na lista de usuários do servidor. Feito isso, o pseudônimo do usuário é gerado. A função **cria_pseudonimo()** é responsável pela criação do pseudônimo e é apresentada na figura 5.35.

A função **cria_pseudonimo()**, cria um novo pseudônimo a partir da concatenação do retorno da função **gera_string_randomica()** com o retorno da função **gera_numero_randomico()**. A função **gera_string_randomica()** retorna uma *string* com até 3 caracteres e a função **gera_numero_randomico()** retorna um *double*, todos gerados randomicamente. Após isso, é verificado se esse pseudônimo já existe para algum dos usuários que estão no shopping. Essa verificação é feita por meio da função **pseudonimo_ja_existe()**. Se for um pseudônimo já existente,

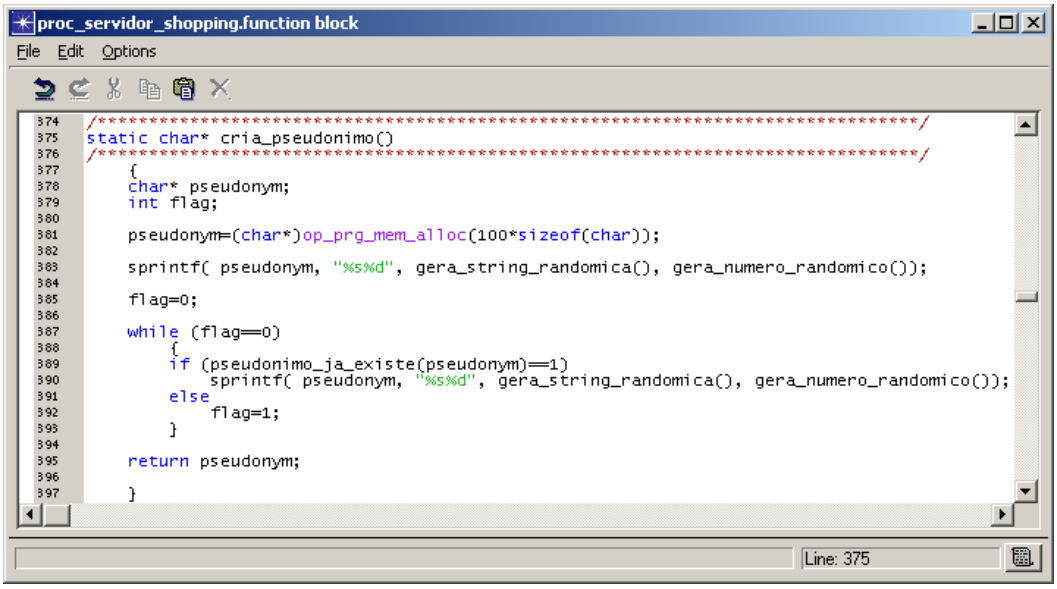

Figura 5.35: Definição da função **cria_pseudonimo()**.

um novo pseudônimo é gerado até que não haja repetição. Ao final, o pseudônimo gerado é retornado ao sensor, para que possa encaminhar ao usuário.

A função **cria_pseudonimo()** é utilizada também na função **rrcv_2()**, que é acionada quando o servidor recebe solicitações do sensor de mixagem para gerar um novo pseudônimo ao usuário.

5.7.2.5 Modelo do nó de sensores das livrarias

O nó sensor da livraria é interessado em obter as informações pessoais do usuário com o objetivo de enviar serviços de propaganda. O seu modelo de nó é apresentado na figura 5.36, onde é possível notar dois fluxos de entrada (**stream 0** e **stream 1**) e um fluxo de saída.

O nó sensor envia periodicamente mensagens de requisição de pseudônimo, que são os pacotes **shopping_sensor_requests_pck**, aos usuários que estão nas suas proximidades. A resposta a essa requisição é recebida no fluxo 0, no pacote **shopping_pseudonym_pck**. Esse pseudônimo é então enviado ao servidor, para que o sensor possa obter as preferências e as políticas de serviço fornecidos pelo usuário. Essas informações são então recebidas por meio do fluxo 1, no pacote **shopping_service_policies_of_user_pck**. Ao receber esse pacote, o sensor verifica quais as propagandas que ele possui que possam interessar ao usuário e as envia

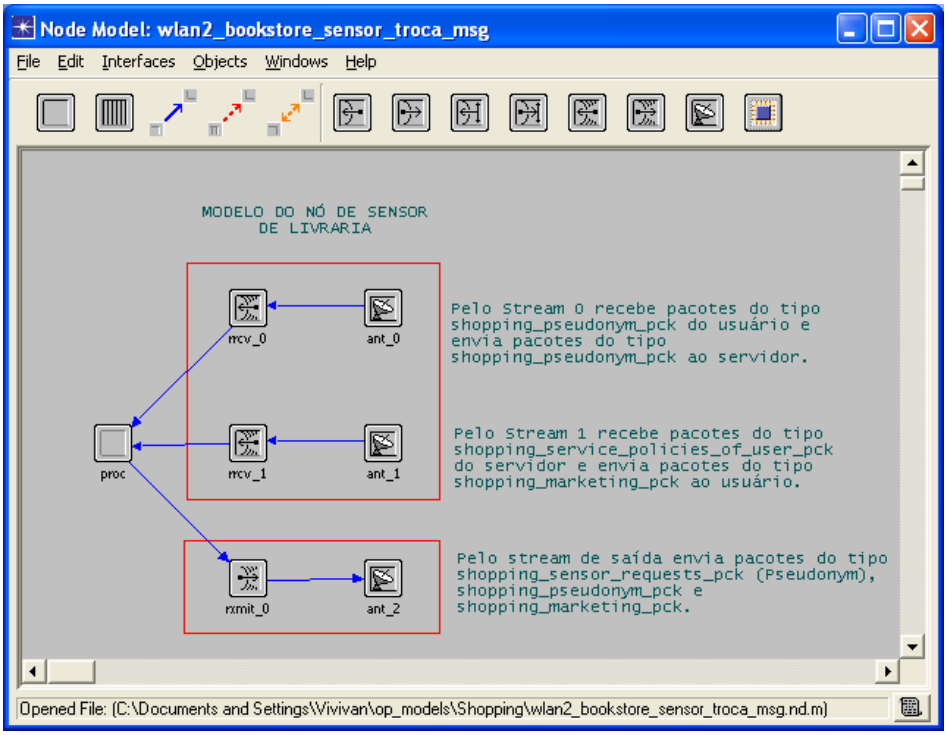

Figura 5.36: Modelo do nó de sensores das livrarias.

no pacote **shopping_marketing_pck**.

O modelo de processo do nó sensor da livraria é apresentado na figura 5.37. Como pode-se perceber, ele possui quatro estados: **start**, **send**, **wait** e **end**, similares aos quatro estados do nó sensor E/S. O estado **start** carrega o endereço IP do sensor internamente. O estado **send** envia pacotes de requisição, que são os pacotes do tipo **shopping_sensor_requests_pck**, e aguarda as respostas à essas requisições no estado **wait**. Por fim, o estado **end** finaliza o processamento.

5.7.2.6 Zona de mixagem

A zona de mixagem é representada no ambiente pelo sensor de mixagem. A função do sensor de mixagem é verificar quais usuários estão interessados em trocar o seu pseudônimo, evitando assim que ele seja associado à qualquer perfil secundário criado pelas lojas, como explicado na seção 5.6.

A figura 5.38 apresenta o modelo do nó sensor de mixagem. O sensor de mixagem envia periodicamente pacotes do tipo **shopping_sensor_requests_pck** do tipo **Mix** aos usuários próximos à sua localidade, para que respondam com a sua taxa de mixagem. O pacote **shopping_mixing_pck** é então recebido pelo fluxo de

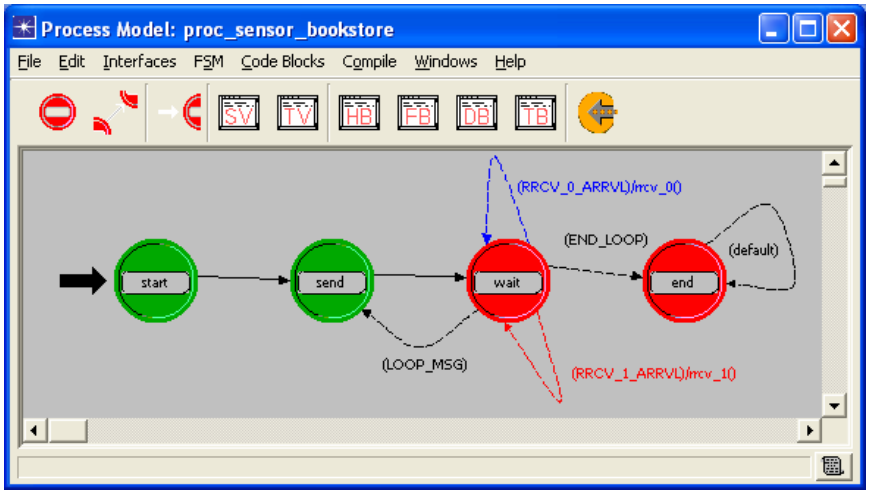

Figura 5.37: Modelo de processo do nó de sensores das livrarias.

entrada e processado. Caso a taxa de mixagem seja 1, o sensor de mixagem envia o pseudônimo do usuário ao servidor, para que seja criado um novo pseudônimo. Essa solicitação é feita por meio do envio do pacote **shopping_sensor_requests_pck** do tipo **New Pseudonym** ao servidor. Ao receber o novo pseudônimo gerado pelo servidor no fluxo 1, no pacote **shopping_new_pseudonym_pck**, o sensor o encaminha ao usuário que deseja participar da mixagem.

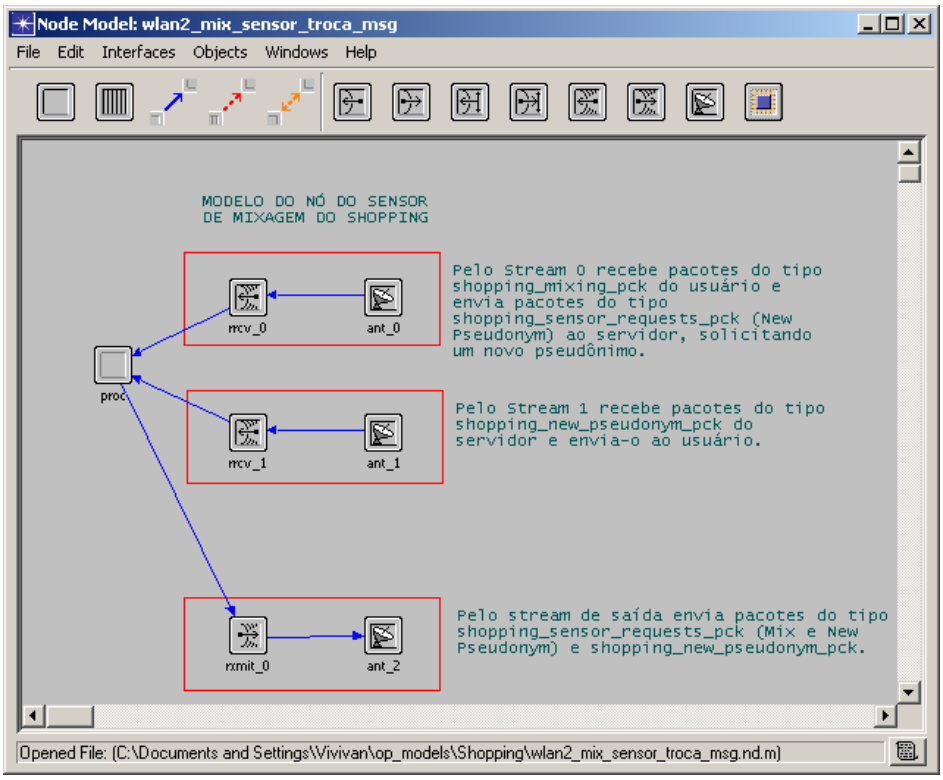

Figura 5.38: Modelo de nó do sensor de mixagem.

Na figura 5.39 é apresentado o modelo de processo do nó sensor de mixagem, o qual possui quatro estados: **start**, **send**, **wait** e **end**. O estado **start** carrega a variável interna do sensor com o seu IP. O estado **send** envia pacotes de requisição **shopping_sensor_requests_pck** aos usuários e aguarda as respostas no estado **wait**. O estado **end** é o que finaliza o processamento do sensor de mixagem.

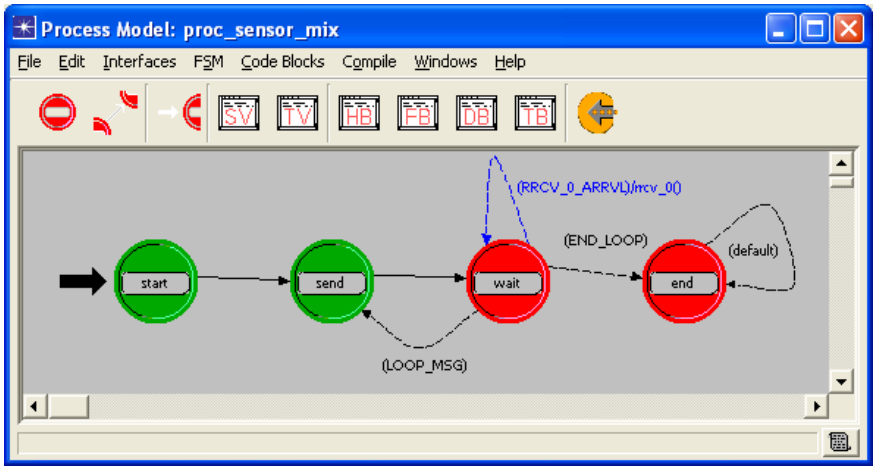

Figura 5.39: Modelo de processo do sensor de mixagem.

5.7.3 Executando uma simulação

Nessa seção é apresentada uma simulação para exemplificar o funcionamento e a interação dos elementos descritos na seção 5.7.2. O cenário exemplo é o apresentado na figura 5.40, onde tem-se somente um cliente que, nesse caso, é a **Alice**.

Como atributos, **Alice** possui os que são apresentados na figura 5.41. Para simplificação, os valores do atributo **Personal Properties** são detalhados na tabela 5.1.

Tabela 5.1: Atributo **Personal Properties** de Alice

Atributo	Valor
Films	action_movies
Books	self_help
Sports	gymnastics

O sensor **Bookstore L Sensor**, possui propagandas disponíveis nas categorias de **films**, **books** e **sports**, conforme apresenta a figura 5.42.

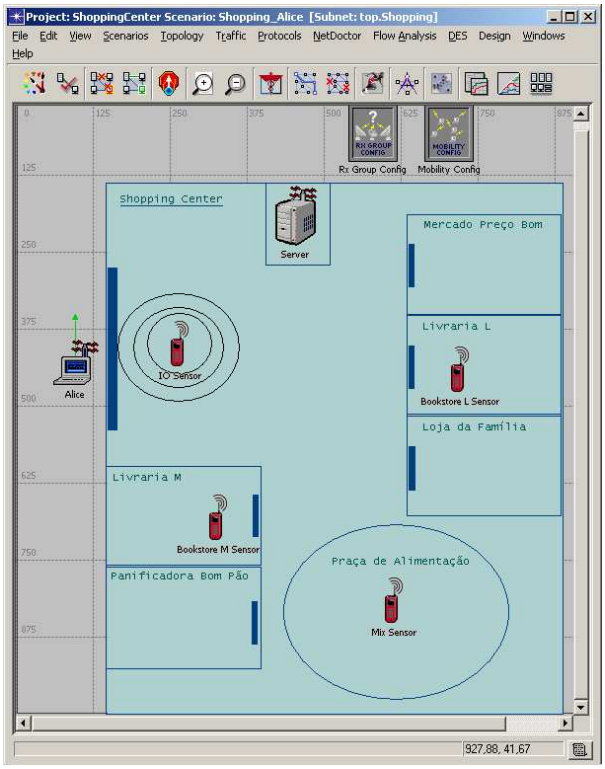

Figura 5.40: Cenário de simulação com apenas um usuário.

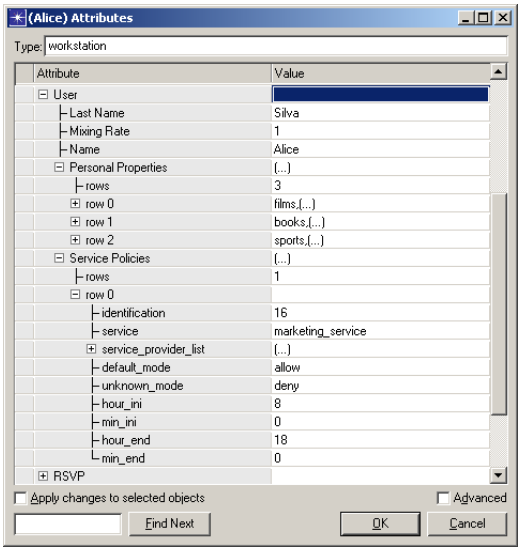

Figura 5.41: Atributos de Alice.

Para simplificar, em cada categoria do **Bookstore L Sensor**, há os produtos apresentados na tabela 5.2. Já o **Bookstore M Sensor** possui propagandas para os produtos e categorias apresentados na tabela 5.3.

O que é esperado nessa simulação é que Alice receba propagandas dos produtos

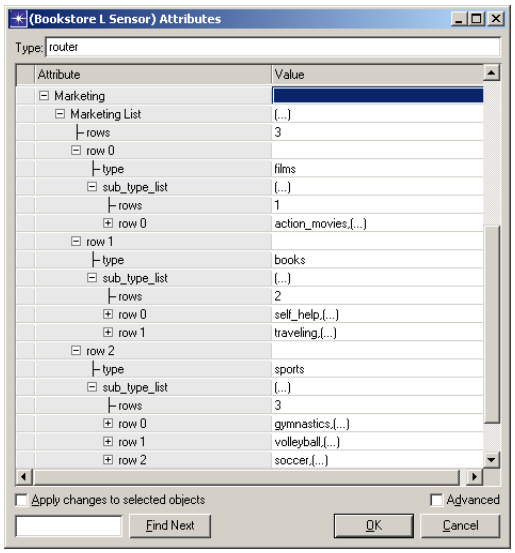

Figura 5.42: Atributos do **Bookstore L Sensor**.

Tabela 5.2: Atributos do **Bookstore L Sensor**

Classe	Categoria	Produto	Valor
Films	action_movies	Velozes e Furiosos	20,00
Books	self_help	Como melhorar a auto-estima	50,00
	traveling	Tudo sobre Joinville	65,00
Sports	gymnastics	Nado sincronizado (Livro)	27,00
	volleyball	Aprenda as principais regras do vôlei (Livro)	14,00
	soccer	O Rei Pelé (Livro)	70,00

Tabela 5.3: Atributos do **Bookstore M Sensor**

Classe	Categoria	Produto	Valor
Films	romance	Romeu e Julieta	45,00
Books	self_help	Uma luz no fim do túnel	57,00

Velozes e Furiosos, *Como melhorar a auto-estima* e *Nado sincronizado*, do **Bookstore L Sensor**, e o produto *Uma luz no fim do túnel* do **Bookstore M Sensor**, já que esses produtos se enquadram nas categorias de interesse de Alice. Além disso, as livrarias não poderão criar um segundo perfil de Alice, pois Alice optou por participar de mixagem ao passar por zonas de mixagem no shopping, atributo **Mixing Rate** como 1.

Para a simulação foi considerado o tempo de 600 segundos, com Alice caminhando na velocidade de 2 m/s. A figura 5.43 apresenta o início da simulação. Nessa etapa os objetos **IO Sensor**, **Alice**, **Mix Sensor**, **Bookstore L Sensor** e **Bookstore M Sensor** são inicializados, e os seus IPs são apresentados. Logo

após iniciar o **IO Sensor**, este envia pacotes de requisição de perfil ao usuário. Em destaque na figura 5.43, está a resposta de **Alice** à essa solicitação.

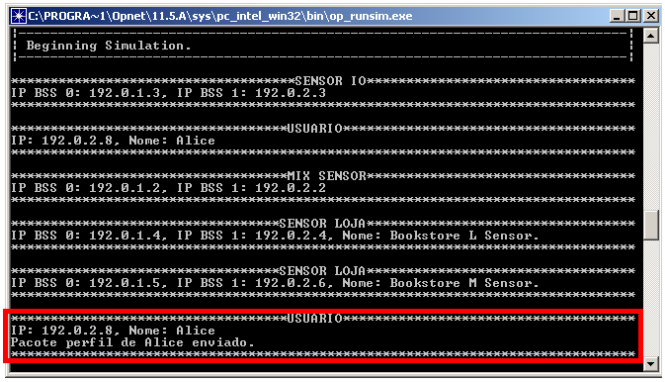

Figura 5.43: Primeira etapa da simulação.

No início da figura 5.44, é possível perceber que o **IO Sensor** recebeu o perfil de **Alice** e o encaminhou ao **Server**. Nesse momento o **Server** é inicializado e o seu IP é apresentado. Após isso, o **Server** recebe o perfil de **Alice** enviado pelo **IO Sensor** e gera um pseudônimo para ela, que é apresentado em tela como **wyw237126223**. Em seguida, esse pseudônimo é enviado ao **IO Sensor**, que o envia à **Alice**. No final dessa figura, a última comunicação realizada foi a resposta de **Alice** a uma solicitação de pseudônimo feita pelo **Bookstore M Sensor**.

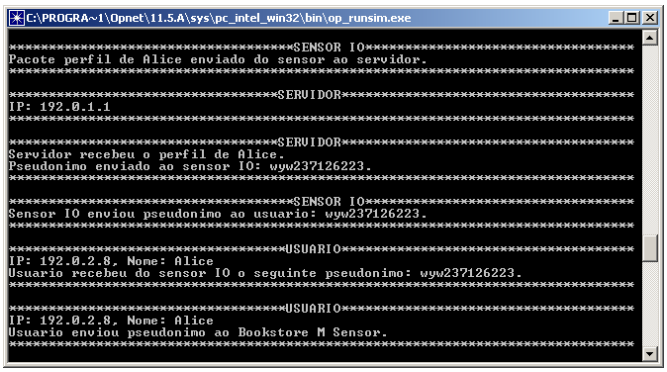

Figura 5.44: Segunda etapa da simulação.

Na figura 5.45, o **Bookstore M Sensor** envia esse pseudônimo ao **Server**, a fim de verificar se ele possui permissão para prestar serviços à **Alice**. O **Server** então recebe esse pseudônimo, verifica as permissões do sensor e envia as preferências pessoais de **Alice**. O **Bookstore M Sensor** verifica quais os produtos que ele possui que podem ser do interesse de **Alice** e os envia. **Alice** recebe em seu dispositivo a propaganda de um único produto *Uma luz no fim do túnel* com preço de 57 reais,

como previsto. Ao final dessa terceira etapa da simulação, **Alice**, ao passar próximo ao **Bookstore L Sensor**, recebe uma requisição de seu pseudônimo, o qual é, então, enviado.

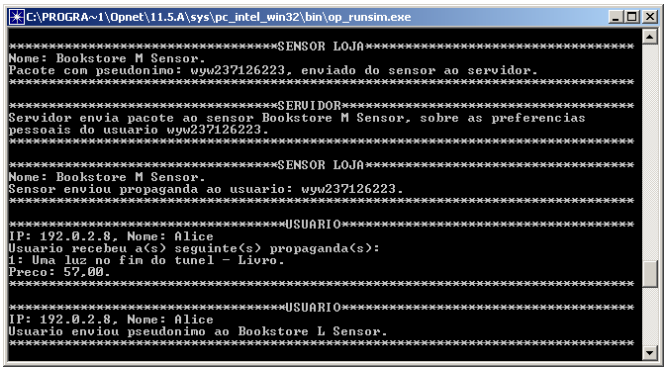

Figura 5.45: Terceira etapa da simulação.

A figura 5.46 apresenta o momento em que o **Bookstore L Sensor** envia o pseudônimo do usuário ao **Server**, para que esse verifique as suas permissões. Após a validação das permissões do **Bookstore L Sensor**, o **Server** envia as preferências pessoais de **Alice**, para que ele possa enviar propagandas. Os produtos que o **Bookstore L Sensor** possui e que são do interesse de **Alice** são: *Velozes e Furiosos*, *Como melhorar a auto-estima* e *Nado sincronizado*, como previsto. Esses produtos são então recebidos pelo dispositivo de *Alice* e exibidos em tela.

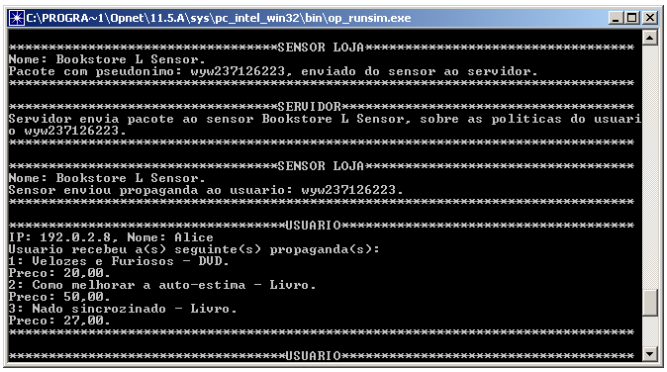

Figura 5.46: Quarta etapa da simulação.

Por fim, a figura 5.47 apresenta o momento em que **Alice** entra em uma zona de mixagem e é abordada pelo **Mix Sensor**, que solicita à ela o seu interesse em participar ou não da mixagem. **Alice** então envia um pacote ao **Mix Sensor** indicando que tem interesse em participar, campo **mixing** igual a 1. Dessa forma, o **Mix Sensor** solicita ao **Server** um novo pseudônimo para **Alice**, a qual ele

conhece por **wyw237126223**. O **Server** gera um novo pseudônimo, atualiza-o no seu cadastro interno para **Alice** e envia-o para o **Mix Sensor**. Como pode-se perceber, o novo pseudônimo gerado foi o **ywy339169902**. O **Mix Sensor** envia esse novo pseudônimo ao usuário. Nesse momento, a simulação é finalizada.

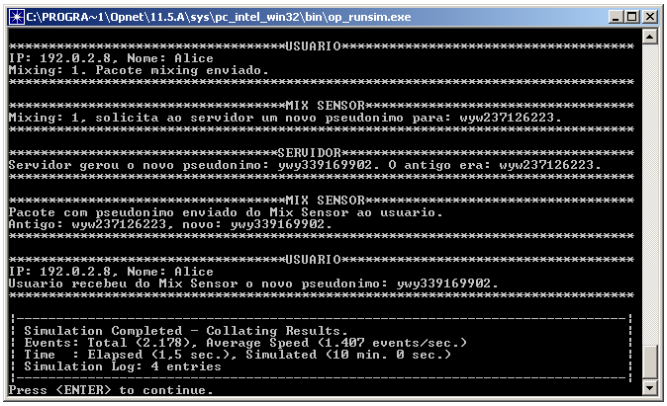

Figura 5.47: Quinta etapa da simulação.

Esse cenário de simulação foi escolhido apenas para exemplificar a interação entre os elementos do shopping. Apesar de **Alice** desejar participar da mixagem, sua identidade, nesse caso, poderia ser facilmente associada ao seu perfil, pois é a única cliente que está no shopping e, como será mostrado na seção 5.7.4.1, nesse caso, não há garantias de anonimato.

5.7.4 Resultados e análise da simulação

Essa seção apresenta como é calculado o grau de anonimato de um ambiente com base na literatura e a análise do grau de anonimato oferecido pelo shopping apresentado na seção 5.7.2.

5.7.4.1 Métricas de Anonimato

O anonimato é o estado de não ser identificável quando se faz parte de um conjunto de usuários, que é o conjunto de anonimato (PFITZMANN; KöHNTOPP, 2001). Para o exemplo do shopping, a definição dada por (NUSSBAUMER, 2007) esclarece melhor o que é o conjunto de anonimato: é o conjunto de pessoas visitando uma zona de mixagem durante o mesmo período. Quanto maior for o conjunto, maior será o grau de anonimato oferecido. Quando o conjunto se reduz para um elemento, o usuário fica completamente exposto e perde todo o seu anonimato.

Porém, o usuário pode negar a informação de sua localização para uma aplicação até que a zona de mixagem ofereça um nível mínimo de anonimato. Esse procedimento não foi implementado no cenário do shopping.

De acordo com (TOTH; HORNAK; VAJDA, 2004), as primeiras publicações tinham como objetivo quantificar o nível de anonimato provido pelo sistema estudado como sendo o tamanho do conjunto de anonimato como, por exemplo, a publicação (BERTHOLD; FEDERRATH; KPSELL, 2000). Porém, essa não é uma boa medida de anonimato, considerando que as probabilidades poderiam não ser distribuídas uniformemente.

Com a necessidade de medir o grau de anonimato, Serjantov e Danezis (SERJANTOV; DANEZIS, 2002) introduziram o conceito de entropia apropriado como uma medida de anonimato. A definição 1 introduz os elementos necessários para o entendimento do conceito de entropia.

Definição 1: Dado um modelo de ataque e um conjunto finito de todos os usuários Ψ, seja $r \in R$ um papel para um usuário ($R = \{\text{remetente, destinatário}\}$) com relação à uma mensagem M. Seja U a probabilidade do usuário $u \in \Psi$ ser atacado tendo o papel r com relação à M.

Com essa definição, a medida de anonimato tanto do remetente, quanto do destinatário pode ser definida como:

Definição 2: O tamanho S de uma distribuição de probabilidade U do anonimato r é igual à entropia da distribuição. Em outras palavras:

$$\boxed{\text{S} = \text{-} \sum_{u=1}^{\Psi} p_u \; log_2 \; p_u} \tag{5.1}$$

onde $p_u = U(u, r)$.

Esse tipo de entropia é conhecida como entropia simples (TOTH; HORNAK; VAJDA, 2004).

Em (DIAZ et al., 2002) foi seguida uma abordagem diferente, onde se considera somente o anonimato do remetente. Seja A representando o conjunto de anonimato

de uma certa mensagem M, ou seja, $A = \{\ u \mid (u \in \Psi) \wedge (p_u > 0)\ \}$. Além disso, seja N o tamanho do conjunto de anonimato, ou seja, $N = |\, A \,|$.

Definição 3: O grau de anonimato provido pelo sistema é definido por:

$$\boxed{d = \frac{H(X)}{H_M}} \tag{5.2}$$

Onde $H(X) = S$ e $H_M = log_2 N$. Para o caso particular de um usuário, d é assumido como sendo zero.

Essa métrica é conhecida como entropia normalizada. Em ambos os casos, zero significa nenhum anonimato, ou seja, o atacante conhece 100% o remetente da mensagem. No caso da entropia simples, o caso máximo de anonimato é alcançado quando $S = log_2 N$ e na normalizada quando $d = 1$ (TOTH; HORNAK; VAJDA, 2004).

Nessa dissertação será utilizada a métrica da entropia normalizada, visto que o interesse maior é no anonimato do remetente, ou seja, do usuário do shopping e não dos demais elementos. O cálculo da métrica utilizada foi introduzido com uma programação à parte para o simulador Opnet.

5.7.4.2 Cenários para simulação

Para a obtenção do grau de anonimato do shopping foram considerados 3 cenários: um com 50 clientes (figura 5.48), outro com 100 clientes (figura 5.49) e, por fim, um cenário com 300 clientes (figura 5.50). Esse é o número de usuários que podem estar presentes na zona de mixagem, mas não necessariamente desejam participar da mixagem. Para cada cenário, foram simuladas algumas situações, variando o número de usuários que desejam participar da mixagem, como mostra a tabela 5.4.

O modelo de ataque para esses cenários é similar ao apresentado por (DIAZ et al., 2002), para o caso da *Onion Routing*. Nesse modelo, N é o tamanho do conjunto de anonimato, e a entropia máxima para esses N usuários é:

$$\boxed{H_M = log_2 N} \tag{5.3}$$

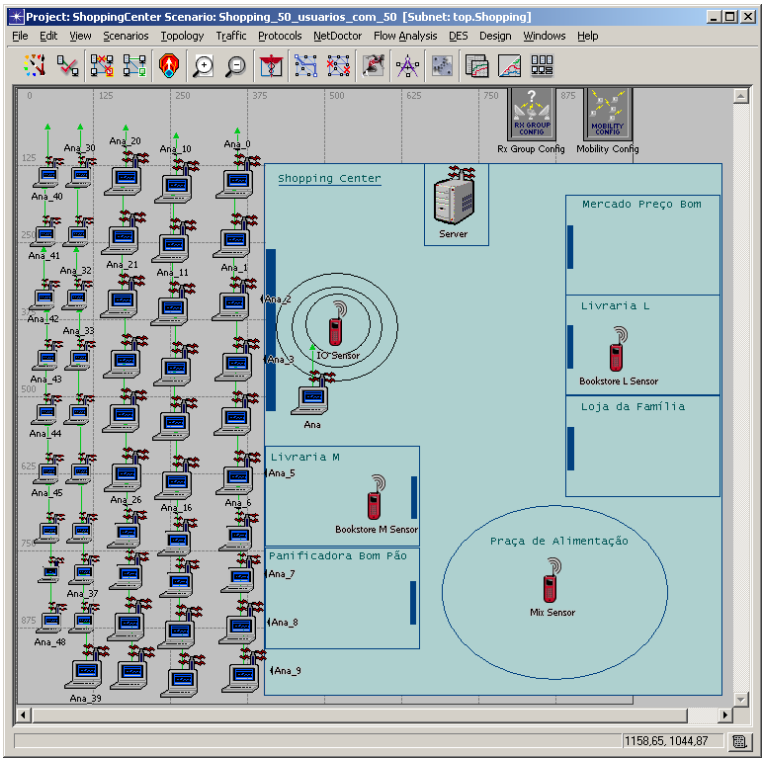

Figura 5.48: Shopping com 50 clientes.

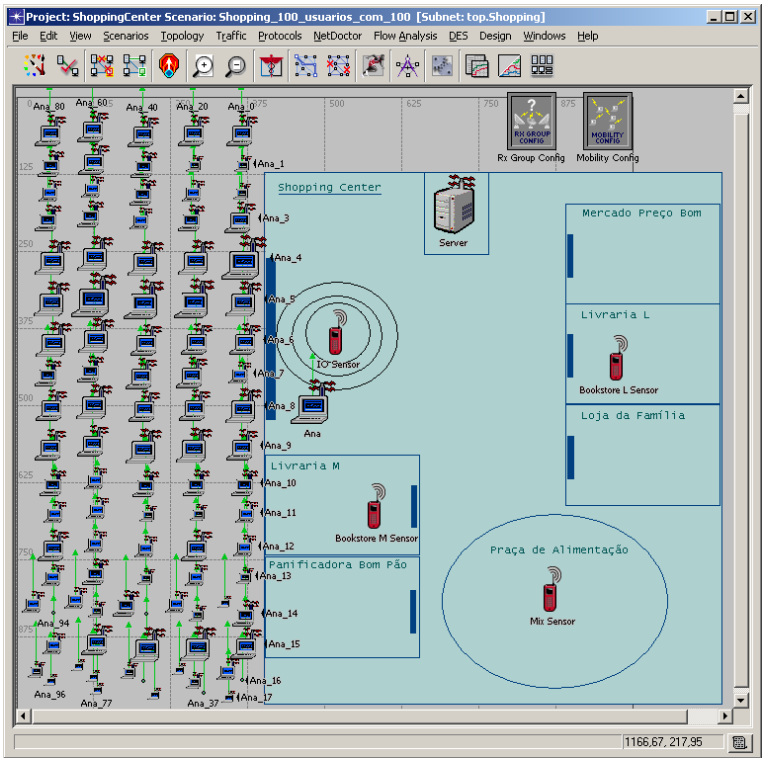

Figura 5.49: Shopping com 100 clientes.

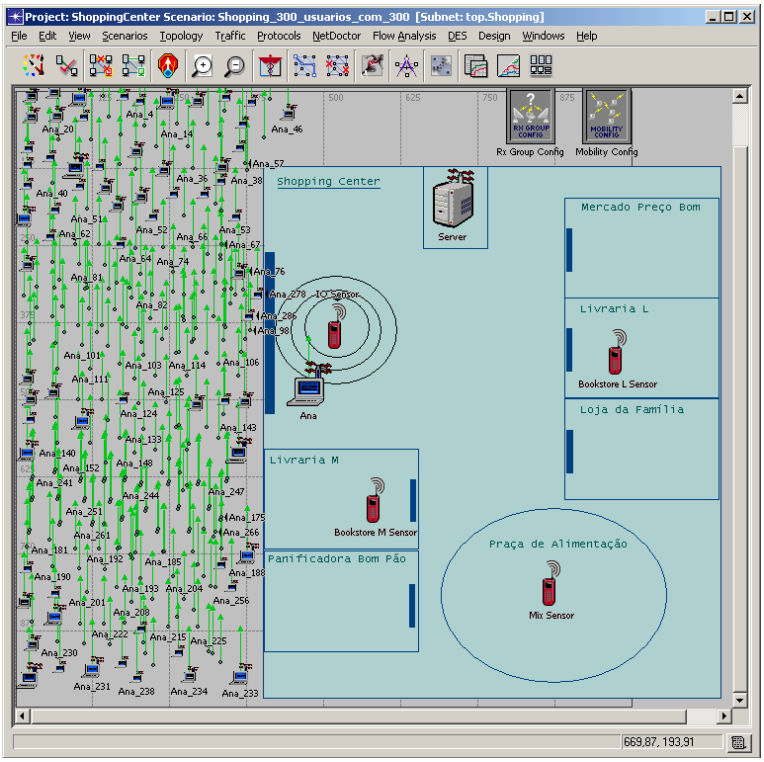

Figura 5.50: Shopping com 300 clientes.

Tabela 5.4: Situações para a simulação.

Usuários na zona de mixagem	Usuários que participam da mixagem
50	1 2 10 25 40 50
100	1 2 20 50 80 100
300	1 2 60 150 240 300

No caso do shopping, N é o número de usuários que estão na zona de mixagem do shopping independente de desejarem ou não participar da mixagem. Nesse caso, o ataque se caracteriza como um dos sensores das lojas tentando associar a identidade de um cliente com um perfil secundário criado por eles mesmos.

O conjunto que contém apenas os usuários interessados em participar da mixagem é chamado de conjunto A, onde $1 \leq A \leq N$. Nesse caso, a distribuição de probabilidades para os A usuários é uniforme:

$$\boxed{p_i = \frac{1}{A},\ 1 \leq i \leq A;\ p_i = 0,\ A\ +\ 1 \leq i \leq N} \tag{5.4}$$

Dessa forma, a entropia e o grau de anonimato são definidos como:

$$\boxed{H(X) = log_2 A,\ d = \frac{H(X)}{H_M} = \frac{log_2 A}{log_2 N}} \tag{5.5}$$

Para aplicar esse modelo de ataque aos cenários apresentados nas figuras 5.48, 5.49 e 5.50, foram considerados vários tamanhos para o conjunto A, como previamente apresentado na tabela 5.4. O grau de anonimato obtido para cada cenário é apresentado na tabela 5.5.

Tabela 5.5: Grau de anonimato obtido para os cenários

Conjunto N	Conjunto A	p_i	$log_2 N$	$log_2 A$	d
50	1	1,0000	5,6439	0,0000	0,0000
	2	0,5000	5,6439	1,0000	0,1772
	10	0,1000	5,6439	3,3219	0,5886
	25	0,0400	5,6439	4,6439	0,8228
	40	0,0250	5,6439	5,3219	0,9430
	50	0,0200	5,6439	5,6439	1,0000
100	1	1,0000	6,6439	0,0000	0,0000
	2	0,5000	6,6439	1,0000	0,1505
	20	0,0500	6,6439	4,3219	0,6505
	50	0,0200	6,6439	5,6439	0,8495
	80	0,0125	6,6439	6,3219	0,9515
	100	0,0100	6,6439	6,6439	1,0000
300	1	1,0000	8,2288	0,0000	0,0000
	2	0,5000	8,2288	1,0000	0,1215
	60	0,0167	8,2288	5,9069	0,7178
	150	0,0067	8,2288	7,2288	0,8785
	240	0,0042	8,2288	7,9069	0,9609
	300	0,0033	8,2288	8,2288	1,0000

As figuras 5.51, 5.52 e 5.53 apresentam graficamente o grau de anonimato d obtido para cada situação do conjunto A. Como pode-se perceber, em todas as situações onde há somente 1 cliente, não há anonimato garantido para esse cliente. Mesmo com dois usuários, o grau de anonimato obtido é muito baixo. Em (DIAZ et al., 2002) é sugerido um

valor intuitivo para o grau mínimo de anonimato para que um sistema forneça anonimato adequadamente. Esse valor é d $\geq$ 0,8 obtido empiricamente.

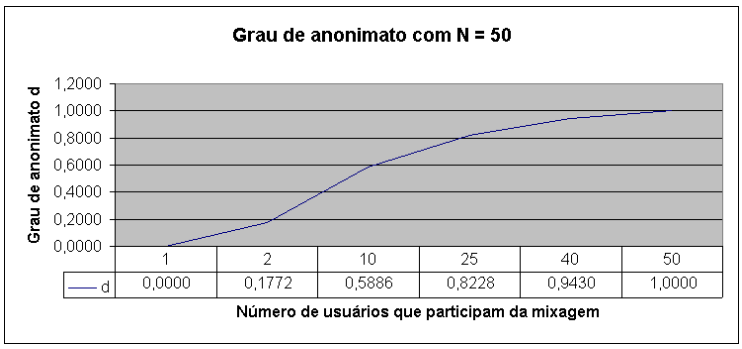

Figura 5.51: Grau de anonimato com $N = 50$.

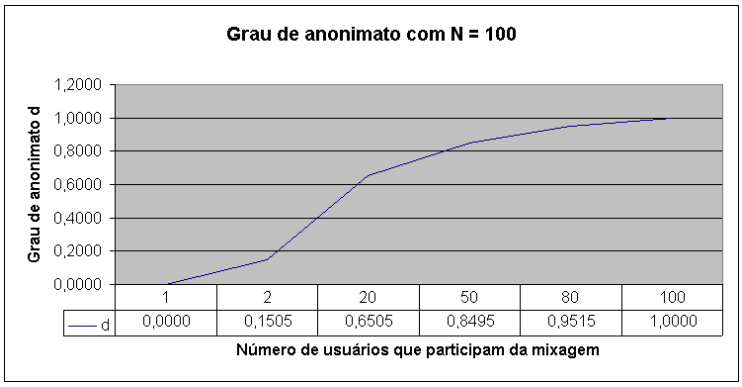

Figura 5.52: Grau de anonimato com $N = 100$.

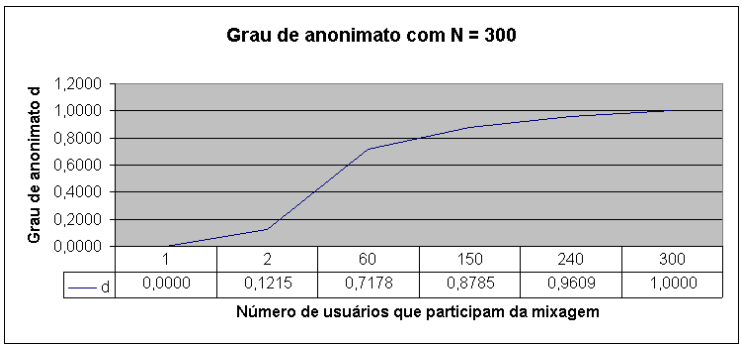

Figura 5.53: Grau de anonimato com $N = 300$.

As situações simuladas, como mostradas nas figuras 5.51, 5.52 e 5.53, para as amostras dos conjuntos A em questão, mostram que o grau de anonimato $\geq$ 0,8 (DIAZ et al., 2002) são as situações onde A $\geq$ 25, para o cenário onde tem-se N = 50 usuários, A $\geq$ 50, para N = 100 usuários e A $\geq$ 150, quando N = 300 usuários. Além disso, o grau de anonimato máximo obtido é quando N = A.

Os resultados apresentados nos gráficos das figuras 5.51, 5.52 e 5.53, foram comparados no gráfico da figura 5.54. Pode-se dizer que o grau de anonimato de um ambiente aumenta, conforme o número de elementos do conjunto A aumenta. Ou seja, quanto mais usuários desejarem participar da mixagem, mais difícil será de um atacante descobrir a identidade de um usuário.

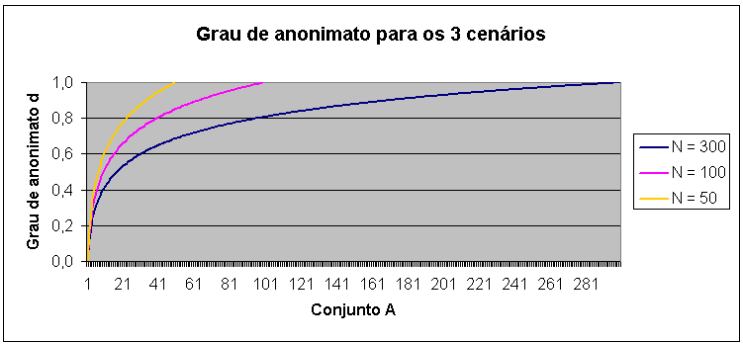

Figura 5.54: Comparativo entre os 3 cenários.

5.8 Considerações

Um problema encontrado na métrica do cálculo do grau de anonimato é que, ao considerar um ambiente onde $N = 2$ e $A = 2$, tem-se $d = 1$. O que significa que foi obtido o grau de anonimato máximo para esse ambiente, considerando-se $p_i = 0,5$. Porém, na prática, sabe-se que $N = 2$ não garante o anonimato desses dois usuários, mesmo com $A = 2$. Por esse motivo, é importante sempre considerar valores maiores para N e A. Uma alternativa seria estipular um valor mínimo para o tamanho de A, e o usuário aceitar participar da mixagem somente quando esse número mínimo de usuários para o conjunto A for atingido.

Além disso, uma das limitações desse trabalho é que ele não aborda técnicas de segurança. Dessa forma, é importante ressaltar que as zonas de mixagem foram consideradas confiáveis.

6 Conclusões e Trabalhos Futuros

Esse trabalho apresentou o metamodelo de (CAMPIOLO, 2005) evoluído com a característica de privacidade. As classes da categoria de privacidade foram desenvolvidas com base nos conceitos de zonas de mixagem, pseudônimos e perfis com as preferências pessoais dos usuários. Para a validação do metamodelo foi dada continuidade ao cenário do shopping center apresentado em (CAMPIOLO, 2005) o qual passou a abordar também a característica de privacidade. Havia dois caminhos a serem seguidos: desenvolver um simulador com base no metamodelo ou optar por um simulador já existente que permitisse a inclusão dos conceitos do metamodelo. A segunda opção foi a escolhida, devido à restrição temporal para a conclusão do presente estudo. Optou-se então pelo simulador Opnet[1], por ele ter uma vasta documentação, por sua flexibilidade na extensão dos seus objetos existentes e também na permissão da criação de novos objetos, por seu suporte à comunicação sem fio e fácil utilização.

O resultado obtido com a adaptação do metamodelo ao simulador Opnet foi apresentado no capítulo 5. Como limitação na simulação foi que o Opnet não permite comunicações síncronas entre as entidades, o que tornou mais trabalhosa a elaboração do cenário. Outra característica do Opnet é que ele exige memória do computador, fazendo com que o código fonte seja feito de maneira a se otimizar ao máximo o uso de variáveis, como também, que sejam utilizados recursos do próprio Opnet para desalocar memória assim que o objeto deixa de ser utilizado e o tamanho dos campos dos pacotes utilizados seja o mínimo possível para o funcionamento dos protocolos de comunicação.

Por meio do uso de pseudônimos os usuários tornaram-se anônimos e para garantir que a identidade de um usuário não fosse associada a um segundo perfil, foram criadas as zonas de mixagem, que são regiões onde os usuários se misturam à multidão, trocando seus pseudônimos por novos pseudônimos, melhorando assim, a sua privacidade.

[1] www.opnet.com

Por fim, utilizou-se a métrica de (DIAZ et al., 2002) para medir o grau de anonimato do ambiente do shopping center. Um modelo de ataque foi elaborado e apresentado no capítulo 5. Três cenários foram escolhidos para o cálculo do grau de anonimato, variando para cada um deles o tamanho do conjunto de clientes que desejavam participar da mixagem, ao entrar em uma zona de mixagem. Resumidamente, os resultados obtidos mostraram que, quanto mais usuários estiverem presentes na zona de mixagem, mais alto será o grau de anonimato oferecido pelo ambiente.

6.1 Trabalhos Futuros

Esse trabalho teve como principal objetivo, o estudo da característica de privacidade em ambientes de computação ubíqua. Contudo, sabe-se que a área de computação ubíqua é uma área em constante crescimento e também uma área que demanda muita pesquisa ainda a ser realizada. A seguir, algumas sugestões de trabalhos futuros são feitas:

- Abordar segurança no metamodelo seria interessante como, por exemplo, utilizar criptografia nos pacotes desenvolvidos nessa dissertação. Seria um mecanismo a mais para proteção da comunicação entre as entidades do ambiente;
- Permitir, na modelagem desenvolvida, que o usuário determine qual o número mínimo de usuários devem estar presentes na zona de mixagem para que ele aceite participar da mixagem;
- Simular outros cenários, como uma casa inteligente, uma faculdade, um escritório, entre outros, para verificar a aplicação do modelo e o grau de anonimato obtido.
- Investigar a possibilidade de se desenvolver um simulador com base no metamodelo apresentado.

Referências

AGRAWAL, R. et al. Hippocratic databases. In: *28th Int'l Conf. on Very Large Databases (VLDB), Hong Kong*. [s.n.], 2002. p. 143–154. Disponível em: <citeseer.ist.psu.edu/agrawal02hippocratic.html>.

AOYAMA, T. A new generation network - beyond ngn. *Innovations in NGN: Future Network and Services. First ITU-T Kaleidoscope Academic Conference*, Geneva, p. 3–10, May 2008.

ARGENTINA. Personal data protection act. May 2008. Disponível em: <http://www.privacyinternational.org/countries/argentina/argentine-dpa.html>.

BAYARDO, R. J.; SRIKANT, R. Technologies: Technological solutions for protecting privacy. *j-COMPUTER*, v. 36, n. 9, p. 115–118, September 2003. ISSN 0018-9162. Disponível em: <http://csdl.computer.org/dl/mags/co/2003/09/r9115.htm; http://csdl.computer.org/dl/mags/co/2003/09/r9115.pdf>.

BERENDT, B.; GUNTHER, O.; SPIEKERMANN, S. Privacy in ecommerce: State preferences vs. actual behavior. *Communications of the ACM*, v. 48, n. 4, p. 101–106, 2005.

BERESFORD, A. R. *Location privacy in ubiquitous computing*. [S.l.], January 2005.

BERESFORD, A. R.; STAJANO, F. Location privacy in pervasive computing. *IEEE Pervasive Computing*, v. 2, n. 2, p. 46–55, April-June 2003.

BERESFORD, A. R.; STAJANO, F. Mix zones: User privacy in location-aware services. *Pervasive Computing and Communications Workshops, IEEE International Conference on*, IEEE Computer Society, Los Alamitos, CA, USA, v. 0, p. 127–131, 2004.

BERTHOLD, O.; FEDERRATH, H.; KPSELL, S. Web mixes: A system for anonymous and unobservable internet access. In: *Designing Privacy Enhancing Technologies*. [S.l.]: Springer-Verlag, 2000. p. 115–129.

BHASKAR, P.; AHAMED, S. I. Privacy in pervasive computing and open issues. In: *ARES '07: Proceedings of the The Second International Conference on Availability, Reliability and Security*. Washington, DC, USA: IEEE Computer Society, 2007. p. 147–154. ISBN 0-7695-2775-2.

BUENNEMEYER, T.; MARCHANY, R.; TRONT, J. G. Ubiquitous security: Privacy versus protection. In: ARABNIA, H. R. (Ed.). *PSC*. [S.l.]: CSREA Press, 2006. p. 71–77. ISBN 1-60132-018-3.

CAMPBELL, R. et al. Towards security and privacy for pervasive computing. In: *In Proceedings of International Symposium on Software Security*. Tokyo, Japan: [s.n.], 2002. p. 1–15.

CAMPIOLO, R. *Aspectos de Modelagem de Ambientes de Computação Ubíqua*. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, Junho 2005.

CAMPIOLO, R.; SOBRAL, J. B. M. Modelagem e simulação em ambientes de computação ubíqua. Florianópolis, 2005. In: I2TS'2005 - International Information and Telecommunication Technologies Symposium.

CHALMERS, D. et al. Manifesto for the ubiquitous computing grand challenge: Experience, design and science. Version 4. UK-UbiNet. February 2006. Disponível em: <http://www-dse.doc.ic.ac.uk/Projects/UbiNet/GC/manifesto.html>.

CHANG, X. Network simulations with opnet. In: *WSC '99: Proceedings of the 31st conference on Winter simulation*. New York, NY, USA: ACM Press, 1999. p. 307–314. ISBN 0780357809. Disponível em: <http://dx.doi.org/10.1145/324138.324232>.

CHENG, H. S.; ZHANG, D.; TAN, J. G. Protection of privacy in pervasive computing environments. In: *ITCC '05: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05)*. Washington, DC, USA: IEEE Computer Society, 2005. v. 2, p. 242–247. ISBN 0-7695-2315-3.

CONSTITUICAO. Constituição da república federativa do brasil. artigo 5. Parágrafos X, XI e XII. 1988.

CULLER, D. E.; HONG, W. Wireless sensor networks. *Communications of the ACM*, v. 47, n. 6, p. 30–33, June 2004.

DIAZ, C. et al. Towards measuring anonymity. In: DINGLEDINE, R.; SYVERSON, P. (Ed.). *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*. [S.l.]: Springer-Verlag, LNCS 2482, 2002. p. 54–68.

DUKE, R. et al. *The Object-Z Specification Language: Verson 1*. [S.l.], 1991.

EUDIRECTIVE1995/46/EC. Directive 95/46/ec of the european parliament and of the council of 24 october 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. An unofficial text: http://www.datenschutz-berlin.de/gesetze/europa/den.htm. For the authoritative text of the Directive, reference should be made to the Official Journal of the European Communities of 23 November 1995 No L. 281 p. 31. November 1995.

FAHRMAIR, M.; SITOU, W.; SPANFELNER, B. Security and privacy rights management for mobile and ubiquitous computing. Tokyo, Japan, September 2005. Workshop on UbiComp Privacy: Privacy in Context.

FILHO, P. J. de F. *Introdução à Modelagem e Simulação de Sistemas*. [S.l.]: Visual Books, 2001.

GELLMAN, R. Does privacy law work? In: *Technology and Privacy: The New Landscape, ed. Philip E. Agre and Marc Rotenberg*. [S.l.: s.n.], 1998. p. 193–218. MIT Press, Cambridge, Massachusetts.

GILC. Privacy and human rights: An international survay of privacy laws and practice. May 2008. Disponível em: <http://www.gilc.org/privacy/survey/>.

HANSMANN, U. et al. *Pervasive Computing*. 2. ed. [S.l.]: Springer, 2003.

JIANG, X.; LANDAY, J. A. Modeling privacy control in context-aware systems. *IEEE Pervasive Computing*, IEEE Computer Society, Los Alamitos, CA, USA, v. 1, n. 3, p. 59–63, 2002. ISSN 1536-1268.

JIANG, X.; LANDAY, J. A. Modeling privacy control in context-aware systems. *IEEE Pervasive Computing*, v. 1, n. 3, p. 59–63, July-September 2002.

KOBSA, A.; SCHRECK, J. Privacy through pseudonymity in user-adaptive systems. *ACM Transactions on Internet Technology*, v. 3, n. 2, p. 149–183, 2003.

LANDAY, J. A.; DAVIS, R. C. Making sharing pervasive: Ubiquitous computing for shared note taking. *IBM Systems Journal*, v. 38, n. 4, p. 531–550, 1999.

LANGHEINRICH, M. Privacy by design — principles of privacy-aware ubiquitous systems. *j-LECT-NOTES-COMP-SCI*, v. 2201, p. 273–??, 2001. ISSN 0302-9743.

LANGHEINRICH, M. A privacy awareness system for ubiquitous computing environments. In: *Ubicomp, Lecture Notes in Computer Science*. [S.l.]: Springer-Verlag, 2002. v. 2498, p. 237–245.

LEDERER, S.; DEY, A. K.; MANKOFF, J. *A Conceptual Model and a Metaphor of Everyday Privacy in Ubiquitous Computing Environments*. [S.l.], June 2002. Disponível em: <http://www.eecs.berkeley.edu/Pubs/TechRpts/2002/5464.html>.

LESSIG, L. *Code and Other Laws of Cyberspace*. New York, NY: Basic Books, 1999. 31, 33, 34 p.

MICHAEL, J. Privacy and human rights: An international and comparative study, with special reference to developments in information technology. Dartmouth Pub Co. / UNESCO. 1994.

MYLES, G.; FRIDAY, A.; DAVIES, N. Preserving Privacy in Environments with Location-Based Applications. *IEEE Pervasive Computing*, v. 2, n. 1, p. 56–64, 2003.

NUSSBAUMER, M. Location privacy. July 2007. Disponível em: <www.sec.informatik.tu-darmstadt.de/pages/lehre/SS07/sem_misc/papers/nussbaumer.pdf>.

ODLYZKO, A. Privacy, economics, and price discrimination on the internet. In: *ICEC'03: Proceedings of the 5th international conference on Electronic commerce*. New York, NY, USA: ACM Press, 2003. p. 355–366. ISBN 1-58113-788-5.

OECD. Recommendation of the council concerning guidelines governing the protection of privacy and transborder flows of personal data. Organisation for Economic Co-operation and Development (OECD). September 1980. Disponível em: <http://webdomino1.oecd.org/horizontal/oecdacts.nsf/linkto/C(80)58>.

PFITZMANN, A.; KöHNTOPP, M. Anonymity, unobservability, and pseudonymity - a proposal for terminology. In: *Designing Privacy Enhancing Technologies*. Springer-Verlag, 2001. p. 1–9. Disponível em: <http://dx.doi.org/10.1007/3-540-44702-4_1>.

RAZAK, S.; ZHOU, M.; LANG, S. Network Intrusion Simulation Using OPNET. In: *OPNETWORK2002 conference*. [S.l.: s.n.], 2002.

RIGHTS. Privacy & human rights. May 2008. Disponível em: <http://www.privacyinternational.org/survey/>.

SAHA, D.; MUKHERJEE, A. Pervasive computing: A paradigm for the 21st century. *IEEE Computer*, v. 36, n. 3, p. 25–31, March 2003.

SALTZER, J. H.; SCHROEDER, M. D. The protection of information in computer systems. *Proceedings of the IEEE*, v. 63, n. 9, p. 1278–1308, September 1975.

SATYANARAYANAN, M. Pervasive computing: Vision and challenges. *IEEE Personal Communications*, v. 8, n. 4, p. 10–17, August 2001.

SERJANTOV, A.; DANEZIS, G. Towards an information theoretic metric for anonymity. In: *Proceedings of Privacy Enhancing Technologies (PET2002)*. [S.l.]: Springer-Verlag, 2002. p. 41–53.

SPIVEY, J. M. *The Z Notation : A Reference Manual*. [S.l.]: Prentice Hall, 1989.

STAJANO, F. *Security for Ubiquitous Computing*. 1. ed. [S.l.]: John Wiley & Sons, 2002.

STAJANO, F. Security for whom? the shifting security assumptions of pervasive computing. In: *International Security Symposium*. [S.l.]: Springer-Verlag GmbH, 2002. (Lectures Notes in Computer Science, v. 2609), p. 16–27.

TOTH, G.; HORNAK, Z.; VAJDA, F. Measuring anonymity revisited. In: LIIMATAINEN, S.; VIRTANEN, T. (Ed.). *Proceedings of the Ninth Nordic Workshop on Secure IT Systems*. Espoo, Finland: [s.n.], 2004. p. 85–90.

USPRIVACYACT. Us privacy act of 1974. 1974. Disponível em: <http://www.usdoj.gov/foia/privstat.htm>.

W3C. Platform for privacy preferences project. W.W.W. Consortium. May 2007. Disponível em: <http://www.w3.org/P3P/>.

WARREN, S. D.; BRANDEIS, L. D. The right to privacy. *Harvard Law Review*, v. 4, n. 5, p. 193–220, 1890. Disponível em: <http://www.lawrence.edu/fast/boardmaw/Privacy_brand_warr2.html>.

WEISER, M. The computer for the twenty-first century. *Scientific American*, p. 94–104, September 1991.

WEISER, M. Hot topics: Ubiquitous computing. *IEEE Computer*, v. 26, n. 10, p. 71–72, October 1993.

WEISER, M. Some computer science issues in ubiquitous computing. *Communications of the ACM*, v. 36, n. 7, p. 75–84, July 1993.

WEISER, M. The world is not a desktop. *ACM Interactions*, v. 1, n. 1, p. 7–8, January 1994.

WEISER, M.; GOLD, R.; BROWN, J. S. The origins of ubiquitous computing research at parc in the late 1980s. *IBM Systems Journal*, v. 38, n. 4, p. 693–696, 1999.

WESTIN, A. F. Privacy and freedom. Atheneum, New York NY. 1967.

WESTIN, A. F. Privacy in america: an historical and sociopolitical analysis. In: *Proceedings of the national privacy and public policy symposium*. [S.l.: s.n.], 1995. Hartford, Connecticut.

APÊNDICE A – Especificação do Modelo em Object-Z

Esse apêndice apresenta a especificação do modelo em Object-Z apresentado por (CAMPIOLO, 2005). As classes especificam as características e restrições dos elementos para modelar e simular ambientes de computação ubíqua, além de algumas funções básicas para gerenciar seus atributos e suas associações. Há classes que não apresentam os métodos básicos, como por exemplo, métodos de adição, acesso e remoção, para não tornar a especificação extensa e excessivamente carregada de funções não tão relevantes (CAMPIOLO, 2005).

$$
\begin{array}{|l}
[TEXT, DATA, SHAPE, TIME, DATE, ACTION, STATE] \\
Properties : TEXT \pfun DATA \\
Direction : \{North, South, East, West, Northeast, Northwest, \\
\qquad\qquad Southeast, Southwest\}
\end{array}
$$

ModelBase

$ClassType : \{Person, Object, Entity, Space\}$

$identification : \mathbb{N}$
$class : ClassType$
$description : TEXT$
$position : \mathbb{N} \times \mathbb{N} \times \mathbb{N}$
$mass : \mathbb{R}_1$
$width : \mathbb{N}$
$height : \mathbb{N}$
$shape : \mathbb{P}(\mathbb{R} \times \mathbb{R} \times \mathbb{R})$
$emittedSignal : \mathbb{F}\ Signal$
$location : Location$
$created : TimeStamp$

$\forall\, x, y : identification \bullet x \neq y$
$\forall\, x, y : identification;\ p1, p2 : position \mid x \neq y \bullet p1 \neq p2$

SetPosition

$\Delta(position)$
$p? : \mathbb{N} \times \mathbb{N} \times \mathbb{N}$

$\exists_1\, s : Space \mid s \in Environment.space \bullet p? \in s.shape$
$position' = p?$

Person

ModelBase

$[POSITIONAL, PHYSICAL, PSYCHOLOGICAL]$

$Personality : \{Extraversion, Agreeableness, Conscientiousness, EmotionalStability, Openness\}$

$profile : Profile$

$positional : \mathbb{P}\, POSITIONAL$
$physical : \mathbb{P}\, PHYSICAL$
$psychological : \mathbb{P}\, PSYCHOLOGICAL$

$speed : \mathbb{R}$
$direction : Direction$
$orientation : Direction$

$gender : \{male, female\}$
$age : \mathbb{N}$

$personality : Personality \rightarrow \mathbb{N}$

$hasObject : \mathbb{F} \downarrow Object$
$hasEntity : \mathbb{F} \downarrow Entity$

$speed \in positional$
$direction \in positional$
$orientation \in positional$
$gender \in physical$
$age \in physical$
$personality \in psychological$
$\operatorname{ran} personality \leq 100$

INIT

$class = Person$

$\Phi SelectEntity$

$\Delta(hasEntity)$
$e?, e' : \downarrow Entity$

$e? \in hasEntity \Rightarrow e' = e?$

$\Phi SelectObject$

$\Delta(hasObject)$
$o?, o' : \downarrow Object$

$o? \in hasObject \Rightarrow o' = o?$

TakeObject

$\Delta(hasObject)$
$o? : \downarrow Object$

$hasObject' = hasObject \cup \{o?\}$

TakeEntity

$\Delta(hasEntity)$
$e? : \downarrow Entity$

$hasEntity' = hasEntity \cup \{e?\}$

PutObject

$\Delta(hasObject)$
$o? : \downarrow Object$

$o? \in hasObject$
$hasObject' = hasObject \setminus \{o?\}$
$o?.SetPosition$

PutEntity

$\Delta(hasEntity)$
$e? : \downarrow Entity$

$e? \in hasEntity$
$hasEntity' = hasEntity \setminus \{e?\}$
$e?.SetPosition$

Move

$\Delta(speed, direction)$
$s? : \mathbb{R}$
$d? : Direction$

$d? \neq direction \Rightarrow direction' = d?$
$s? \neq speed \Rightarrow speed' = s?$
$speed' \neq 0 \Rightarrow SetPosition$

Object

ModelBase

$$
\begin{array}{l}
initialState : \mathbb{P}\, STATE \\
possibleState : \mathbb{P}\, STATE \\
changeState : ACTION \pfun (STATE \rel STATE) \\
associations : \mathbb{F} \downarrow ModelBase \\
\hline
initialState \subseteq possibleState
\end{array}
$$

INIT

$$
class = Object
$$

AddAssociation

$$
\begin{array}{l}
\Delta(associations) \\
m? : \downarrow ModelBase \\
\hline
associations' = associations \cup \{m?\}
\end{array}
$$

RemoveAssociation

$$
\begin{array}{l}
\Delta(associations) \\
m? : \downarrow ModelBase \\
\hline
associations' = associations \setminus \{m?\}
\end{array}
$$

ReceiveAction

$$
\begin{array}{l}
\Delta(initialState) \\
a? : ACTION \\
\hline
a? \in \mathrm{dom}\, changeState \Rightarrow \\
\quad initialState' = (\ initialState \setminus \\
\quad \{first(changeState(a?))\}\ ) \\
\quad \cup \{second(changeState(a?))\}
\end{array}
$$

Entity

ModelBase

$[Process, Protocol]$

$$
\begin{array}{l}
enabled : \mathbb{B} \\
\\
connections : \mathrm{seq}\downarrow Entity \\
communication : \mathrm{seq}\, Protocol \\
channel : \mathrm{seq}\, CommunicationChannel \\
\\
offeredServices : \mathbb{P}\, Service \\
\hline
\#connections = \#communication = \#channel \\
\exists f : \mathrm{ran}\, connections \times \mathrm{ran}\, communication \times \mathrm{ran}\, channel \\
\quad \mid \forall i : 1..\#f \bullet \\
\quad f(i) = (connections(i), communication(i), channel(i))
\end{array}
$$

INIT

$$
\begin{array}{l}
class = Entity
\end{array}
$$

TurnOn

$$
\begin{array}{l}
\Delta(enabled) \\
\hline
enabled' = \mathrm{true}
\end{array}
$$

TurnOff

$$
\begin{array}{l}
\Delta(enabled) \\
\hline
enabled' = \mathrm{false}
\end{array}
$$

AddConnection

$$
\begin{array}{l}
\Delta(connections, communication, channel) \\
e? : \downarrow Entity \\
p? : Protocol \\
c? : CommunicationChannel \\
\hline
connections' = connections^\frown < e? > \\
communication' = communication^\frown < p? > \\
channel' = channel^\frown < c? >
\end{array}
$$

RemoveConnection

$$
\begin{array}{l}
\Delta(connections, communication, channel) \\
index? : \mathbb{N} \\
\hline
connections' = squash(\{index?\} \lhd connections) \\
communication' = squash(\{index?\} \lhd communication) \\
channel' = squash(\{index?\} \lhd channel)
\end{array}
$$

SendNotify

$t! : \downarrow Trigger$

$connections \neq \langle\rangle$

ReceiveNotify

$t? : \downarrow Trigger$

AddOfferedServices

$offeredService? : Service$

$offeredServices' = offeredServices \cup \{offeredService?\}$

RemoveOfferedServices

$offeredService? : Service$

$offeredServices' = offeredServices \setminus \{offeredService?\}$

Sensor

Entity

$type : \mathbb{N}$
$perceptionShape : SHAPE$
$perceptionDistance : \mathbb{N}$
$knownSignal : SignalType$
$internalState : \mathbb{P}\, STATE$
$stateTransition : Signal \pfun (STATE \rel STATE)$

$\#knownSignal = 1$
$\mathrm{dom}\ \mathrm{ran}\, stateTransition \subset internalState$
$\mathrm{ran}\ \mathrm{ran}\, stateTransiction \subset internalState$

PerceiveSignal

$\Delta(internalState)$
$s? : Signal$

$knownSignal = s?.knownSignal \Rightarrow$
$\quad internalState' = (\ internalState \setminus$
$\quad \{first(stateTransition(s?))\}\ )$
$\quad \cup \{second(stateTransition(s?))\}$

SendNotify

$e? : Event$
$c? : Command$

$\forall\, d : Device \mid d \in \mathrm{ran}\, connections \bullet t! = e?$
$\forall\, a : Actuator \mid a \in \mathrm{ran}\, connections \bullet t! = c?$

Actuator

Entity

$$
\begin{array}{l}
object : \mathbb{P} \downarrow Object \\
action : \mathbb{P}\, ACTION \\
actuation : Command \pfun (Object \rel ACTION) \\
\hline
\mathrm{dom}\ \mathrm{ran}\, actuation \subseteq object \\
\mathrm{ran}\ \mathrm{ran}\, actuation \subseteq action
\end{array}
$$

ReceiveNotify

$$
\begin{array}{l}
c! : Command \\
\hline
c! = t?
\end{array}
$$

SendAction

$$
\begin{array}{l}
c? : Command \\
a! : ACTION \\
o! : Object \\
\hline
o! = first(actuation(c?)) \\
a! = second(actuation(c?))
\end{array}
$$

Trigger

$$
\begin{array}{l}
identification : \mathbb{N} \\
contents : \mathbb{P}\, Properties \\
\hline
\forall\, x, y : identification \bullet x \neq y
\end{array}
$$

Disparate

[abstract operation to run a event]

Stop

[abstract operation to cancel a event]

SignalType

$$
\begin{array}{l}
identification : \mathbb{N} \\
description : TEXT
\end{array}
$$

EventType

$$
\begin{array}{l}
identification : \mathbb{N} \\
description : TEXT
\end{array}
$$

Signal

Trigger

[*SignalConstraints*]

$type : SignalType$

$source : \downarrow ModelBase$

$target : \mathbb{P}\, Sensor$

$propagation : SHAPE$

$range : \mathbb{N}$

$constraints : SignalConstraints$

Event

Trigger

$type : EventType$

$source : \downarrow Device$

$target : \mathbb{P} \downarrow Device$

Command

Trigger

$source : \downarrow Entity$

$target : \mathbb{P}\, Actuator$

$sintaxe : DATA$

$parameters : \mathbb{P}\, DATA$

$sintaxe \in \operatorname{ran} contents$

$parameters \subset \operatorname{ran} contents$

Device

Entity

$[InputDevice, OutputDevice]$
$pid, mid : \mathbb{N}$

$model : TEXT$

$input : \mathbb{P}\, InputDevice$
$output : \mathbb{P}\, OutputDevice$
$entry : InputDevice \pfun \mathbb{P}\, DATA$
$exit : OutputDevice \pfun \mathbb{P}\, DATA$

$process : pid \pfun Process$
$aliveProcess : \mathbb{P}(Process)$

$state : \mathbb{P}\, STATE$
$stateTransition : Event \pfun (STATE \rel STATE)$

$memory : \mathbb{P}\, DATA$
$addressMemory : mid \pfun DATA$

$event : \mathbb{P}\, Event$
$eventMap : Event \pfun \mathbb{P}\, Device$

$input = \mathrm{dom}\, entry$
$output = \mathrm{dom}\, exit$
$aliveProcess = \mathrm{ran}\, process$
$\mathrm{dom}\ \ \mathrm{ran}\, stateTransition \subseteq state$
$\mathrm{ran}\ \ \mathrm{ran}\, stateTransition \subseteq state$
$memory = \mathrm{ran}\, addressMemory$
$event = \mathrm{dom}\, eventMap$

Register

$\Delta(eventMap)$
$e? : Event$
$d? : Device$

$e? \in Event$
$eventMap'(e?) = eventMap(e?) \cup \{d?\}$

Unregister

$\Delta(eventMap)$
$e? : Event$
$d? : Device$

$e? \in Event$
$eventMap'(e?) = eventMap(e?) \setminus \{d?\}$

$\Phi SelectInput$

$\Delta(input)$

$i?, i' : InputDevice$

$i? \in InputDevice \Rightarrow i' = i?$

$\Phi SelectOutput$

$\Delta(output)$

$o?, o' : OutputDevice$

$o? \in OutputDevice \Rightarrow o' = o?$

SendNotify

$e! : Event$

$e! \in event$

ReceiveNotify

$e? : Event$

$e? = t?$

[event implications]

EmbeddedDevice

Device

$embeddedIn : Object$

PortableDevice

Device

$owner : Person$

INIT

$owner = \varnothing$

FixedDevice

Device

$space : Space$

Location

$$symLocation : SymbolicLocation$$
$$relLocation : \text{seq}\, RelativeLocation$$

SymbolicLocation

$$place : Space$$
$$point : \mathbb{N} \times \mathbb{N} \times \mathbb{N}$$

RelativeLocation

$$toObject : \downarrow ModelBase$$
$$orientation : Direction$$
$$distance : \mathbb{N}$$

Wall

Object

$$visible : \mathbb{B}$$
$$opaque : \mathbb{B}$$

$$\#shape = 2$$

Window

Object

$$stateOpened : \mathbb{B}$$
$$opaque : \mathbb{B}$$

$$stateOpened \in possibleState$$

Door

Object

$$alwaysOpened : \mathbb{B}$$
$$stateOpened : \mathbb{B}$$
$$opaque : \mathbb{B}$$

$$stateOpened \in possibleState$$
$$\#shape = 2$$

Space

ModelBase

$$
\begin{array}{l}
name : TEXT \\
delimiters : \mathbb{F} \downarrow Object \\
persons : \mathbb{F}\ Person \\
objects : \mathbb{F}\ Object \\
entities : \mathbb{F} \downarrow Entity
\end{array}
$$

INIT

$$
\begin{array}{l}
class = Space \\
\forall\, d : \downarrow Object \mid d \in delimiters \bullet d.INIT \\
\forall\, p : Persons \mid p \in persons \bullet p.INIT \\
\forall\, o : Objects \mid o \in objects \bullet o.INIT \\
\forall\, e : Entity \mid e \in entities \bullet e.INIT
\end{array}
$$

$\Phi SelectPerson$

$$
\begin{array}{l}
\Delta(persons) \\
p?, p' : Person \\
\hline
p? \in persons \Rightarrow p' = p?
\end{array}
$$

$\Phi SelectObject$

$$
\begin{array}{l}
\Delta(objects) \\
o?, o' : Object \\
\hline
o? \in objects \Rightarrow o' = o?
\end{array}
$$

$\Phi SelectEntity$

$$
\begin{array}{l}
\Delta(entities) \\
e?, e' : \downarrow Entity \\
\hline
e? \in entities \Rightarrow e' = e?
\end{array}
$$

$\Phi SelectDelimiter$

$$
\begin{array}{l}
\Delta(delimiters) \\
d?, d' : \downarrow Object \\
\hline
d? \in delimiters \Rightarrow d' = d?
\end{array}
$$

Environment

$name : TEXT$
$space : \mathbb{F}\, Space$

$timeController : TimeController$
$situationalController : SituationalController$
$locationController : LocationController$
$energyController : EnergyController$
$communicationController : CommunicationController$

$\#space > 0$

INIT

$\forall\, s : Space \mid s \in space \bullet s.INIT$

$locationController.INIT$
$energyController.INIT$
$communicationController.INIT$
$situationalController.INIT$
$timeController.INIT$

$\Phi SelectSpace$

$\Delta(space)$
$s?, s' : Space$

$s? \in space \Rightarrow s' = s?$

CommunicationChannel

$CommunicationConstraints : \{Interference, Delay, Obstruction,$
$Noise, DataLost, Synchronization, DeniedAccess, ...\}$

$physicalChannel : \{infrared, radio, opticalfiber, satellite, ...\}$
$source : \mathbb{P} \downarrow Entity$
$target : \mathbb{P} \downarrow Entity$
$propagation : SHAPE$
$range : \mathbb{N}$
$transmissionRate : \mathbb{R}_1$
$frequency : \mathbb{R}_1$

$constraints : \mathbb{P}\, CommunicationConstraints$

TimeStamp

$$date : DATE$$
$$time : TIME$$

SetTime

$$\Delta(date, time)$$
$$time? : TIME$$
$$date? : DATE$$

$$time' = time?$$
$$date' = date?$$

TimeController

$$clock : TimeStamp$$
$$scheduler : TimeStamp \leftrightarrow \downarrow Trigger$$
$$duration : TimeStamp \leftrightarrow \downarrow Trigger$$
$$frequency : TimeStamp \leftrightarrow \downarrow Trigger$$

$$\forall\, t : TimeStamp \mid (t \in \operatorname{dom} scheduler \land t = clock) \bullet$$
$$\quad scheduler(t).Disparate$$
$$\forall\, event : \downarrow Trigger \mid event \in \operatorname{ran} duration \bullet$$
$$\quad \exists\, t : TimeStamp \mid$$
$$\quad (t \in \operatorname{dom} scheduler \land scheduler(t) = event) \bullet$$
$$\quad clock = (t + duration^{\sim}(event)) \Rightarrow event.Stop$$
$$\forall\, event : \downarrow Trigger \mid event \in \operatorname{ran} frequency \bullet$$
$$\quad \exists_1\, t : TimeStamp \mid$$
$$\quad (t \in \operatorname{dom} scheduler \land scheduler(t) = event) \bullet$$
$$\quad \forall\, i : \mathbb{N}_1 \bullet$$
$$\quad clock = (t + i * frequency^{\sim}(event)) \Rightarrow event.Disparate$$

INIT

[add events]

$$clock.SetTime$$

SituationalController

$$situation : \mathbb{F}\, STATE \nrightarrow \downarrow Trigger$$

$$y : \mathbb{F}\, STATE \mid y \in \operatorname{dom} situation \bullet$$
$$(\forall\, x : STATE \mid (x \in y) \land (x = \text{true}) \bullet situation(y).Disparate)$$

INIT

[add situations]

LocationController

$$registeredElement : \mathbb{P} \downarrow ModelBase$$
$$location : \downarrow ModelBase \nrightarrow Location$$

$$registeredElement = \operatorname{dom} location$$

INIT

[add elements and locations]

EnergyController

$$device : \mathbb{P} \downarrow Entity$$
$$energy : \downarrow Entity \nrightarrow \mathbb{R}$$

$$device = \operatorname{dom} energy$$
$$\forall x : \downarrow Entity \mid x \in device \land energy(x) = 0 \bullet x.TurnOff$$

INIT

[add monitored devices]

CommunicationController

$$activeEntities : \mathbb{P} \downarrow Entity$$
$$activeChannels : \mathbb{P}\ CommunicationChannel$$
$$connections : CommunicationChannel \times \downarrow Entity$$
$$area : \mathbb{P}\ SHAPE$$
$$restrictArea : SHAPE \nrightarrow CommunicationChannel.Constraints$$

$$activeEntities = \operatorname{ran} connections$$
$$activeChannels = \operatorname{dom} connections$$
$$area = \operatorname{dom} restrictArea$$

INIT

$$activeEntities = \varnothing$$
$$activeChannels = \varnothing$$
$$connections = \varnothing$$

[add restrict areas]

APÊNDICE B – Classes de Privacidade

Nesse apêndice são apresentadas as três classes de privacidade que foram acrescentadas ao modelo apresentado no apêndice A. Assim como em (CAMPIOLO, 2005), há classes que não apresentam os métodos básicos, como por exemplo, métodos de adição, acesso e remoção.

Service

ServiceState

$identification : \mathbb{N}$
$description : TEXT$
$created : TimeStamp$

$\forall\, x, y : identification \bullet x \neq y$

CreateService

$\Delta(identification, description, created)$
$identification? : \mathbb{N}$
$description? : TEXT$
$created? : TimeStamp$

$identification' = identification?$
$description' = description?$
$created' = created?$

PrivacyPolicy

$MODE ::= allow \mid deny \mid ask$

PolicyState

$identification : \mathbb{N}$
$service : Service$
$serviceProvider : \mathbb{P} \downarrow Entity$
$defaultMode : MODE$
$unknownMode : MODE$
$temporalConstraint : \text{seq}(TIME \times TIME)$

SetService

$\Delta(service)$
$service? : Service$

$service' = service?$

SetProvider

$\Delta(serviceProvider)$
$provider? : \downarrow Entity$
$mode? : MODE$

$serviceProvider' = serviceProvider \cup \{provider?\}$

AlreadyKnownProvider

$\Xi(mode)$
$provider? : \mathbb{P} \downarrow Entity$
$mode! : MODE$

$provider \in serviceProvider$
$mode! = defaultMode$

AllowModeProvider

$\Xi(mode)$
$mode! : MODE$

$unknownMode = allow$
$mode! = allow$

DenyModeProvider

$\Xi(mode)$
$mode! : MODE$

$unknownMode = deny$
$mode! = deny$

AskUser

$\Xi(mode)$
$provider? : \mathbb{P} \downarrow Entity$
$mode! : MODE$

$provider \notin serviceProvider$
$mode! = allow \lor mode! = deny$

AskModeProvider

$\Xi(mode)$

$mode! : MODE$

$unknownMode = ask$

$mode! = askUser$

$AskModeProvider \mathrel{\widehat=} AlreadyKnownProvider \lor$
$AllowModeProvider \lor DenyModeProvider \lor$
$AskModeProvider$

Profile

ProfileState

$identification : \mathbb{N}$
$pseudonym : TEXT$
$personalProperties : \mathbb{P}(TEXT \times \mathbb{F}\ TEXT)$
$servicePolicies : \mathbb{P}\ PrivacyPolicy$
$mixingRate : \mathbb{N}$

$\forall x, y : pseudonym \bullet x \neq y$

SetPseudonym

$\Delta(pseudonym)$
$pseudonym? : TEXT$

$mixingRate > 0$
$pseudonym' = pseudonym?$

GetPseudonym

$\Xi(pseudonym)$
$pseudonym! : TEXT$

$pseudonym! = pseudonym$

SetPersonalProperties

$\Delta(Profile)$

GetPersonalProperties

$\Xi(Profile)$

SetServicePolicies

$\Delta(servicePolicies)$
$servicePolicy? : PrivacyPolicy$

$servicePolicies' = servicePolicies \cup \{servicePolicy?\}$

GetServicePolicies

$\Xi(servicePolicies)$
$servicePolicies! : \mathbb{P}\ PrivacyPolicy$

$servicePolicies! = servicePolicies$

APÊNDICE C – Aplicação da Especificação

Neste apêndice, a especificação do modelo apresentada no apêndice A e no apêndice B é aplicada ao cenário de um shopping center, descrito no capítulo 5. Assim como em (CAMPIOLO, 2005), algumas notações foram adotadas:

- Em determinados pontos a linguagem textual é empregada para descrever propriedades complexas, como por exemplo, a propriedade **shape**.
- Propriedades compostas por muitos elementos são resumidas com a notação reticência (...).
- Alguns objetos são apresentados resumidos, explicitando somente as propriedades relevantes para discussão.
- Alguns objetos não foram especificados (ex.: eventos), apesar da existência de referência.

Person : *alice*

$identification = 1$
$profile = profile\_A$
$class = Person$
$description =$ client
$position = (65, 85, 0)$
$mass = 57Kg$
$width = 47cm$
$height = 176cm$
$shape = cylinder$
$emittedSignal = \{sig1\}$
$positional = \{position, speed, direction, orientation\}$
$physical = \{gender, age\}$
$psychological = \{personality\}$
$speed = 2m/s$
$direction = West$
$orientation = West$
$gender = female$
$age = 23$
$personality = \{(1, 95), (2, 80), (3, 80), (4, 70), (5, 70)\}$
$hasObject = \{\}$
$hasEntity = \{handheld\}$

Sensor : *senIO*

$identification = 2$
$class = Entity$
$description =$ presence detector sensor
$position = (100, 40, 16)$
$enabled = true$
$connections = \langle server \rangle$
$communication = \langle prot1 \rangle$
$channel = \langle com1 \rangle$
$type = 5$
$perceptionShape = circle$
$perceptionDistance = 2m$
$knownSignal = presence$
$internalState = \{detect[false]\}$
$stateTransition = \{sig1 \mapsto (detect[false], detect[true])\}$

Sensor : *sen_bookstore_L*

$identification = 3$
$class = Entity$
$description =$ presence detector sensor
$position = (68, 60, 16)$
$enabled = true$
$connections = \langle server, handheld \rangle$
$communication = \langle prot2, prot3 \rangle$
$channel = \langle com1, com2 \rangle$
$offeredServices = \{marketing\_service, ...\}$
$type = 5$
$perceptionShape = circle$
$perceptionDistance = 2m$
$knownSignal = presence$
$internalState = \{detect[false]\}$
$stateTransition = \{sig1 \mapsto (detect[false], detect[true])\}$

Sensor : *sen_bookstore_M*

$identification = 4$
$class = Entity$
$description =$ presence detector sensor
$position = (75, 25, 16)$
$enabled = true$
$connections = \langle server \rangle$
$communication = \langle prot2 \rangle$
$channel = \langle com1 \rangle$
$offeredServices = \{marketing\_service, ...\}$
$type = 5$
$perceptionShape = circle$
$perceptionDistance = 2m$
$knownSignal = presence$
$internalState = \{detect[false]\}$
$stateTransition = \{sig1 \mapsto (detect[false], detect[true])\}$

SignalType : *presence*

$identification = 5$
$description =$ presence signal of a person

Signal : *sig*1

$identification = 6$
$contents = \{(presence, true)\}$
$type = presence$
$source = \{\}$
$target = \{senIO, sen\_bookstore\_L, sen\_bookstore\_M\}$
$propagation = circle$
$range = 3m$
$constraints = \{\}$

PortableDevice : *handheld*

$identification = 7$
$class = Entity$
$description =$ handheld
$position = (65, 75, 10)$
$mass = 0.10Kg$
$width = 10cm$
$height = 6cm$
$enabled = true$
$connections = \langle sen\_bookstore\_L \rangle$
$communication = \langle prot3 \rangle$
$channel = \langle com2 \rangle$
$model =$ Palm Zire 21
$input = \{datebook\_button, addressbook\_button, power\_button,$
$up\_button, down\_button, display\}$
$output = \{display\}$
...
$owner = alice$

FixedDevice : *server*

$identification = 8$
$class = Entity$
$description =$ computer to register privacy policy data
$position = (45, 85, 10)$
$mass = 5Kg$
$width = 30cm$
$height = 40cm$
$shape = cube$
$enabled = true$
$connections = \langle senIO, sen\_bookstore\_L, sen\_bookstore\_M \rangle$
$communication = \langle prot1, prot2 \rangle$
$channel = \langle com1 \rangle$
$model =$ Dell Dimension
$input = \{keyboard, network\}$
$output = \{monitor\}$
$entry = ...$
$exit = ...$
$process = \{(1, register\_policy1), (2, identify\_person), ...\}$
$aliveProcess = \{register\_policy1\}$
$state = \{\}$
$stateTransition = \{\}$
$memory = \{person\_info, privacy\_policy\_info...\}$
$addressMemory = \{(1A, person\_info), (6E, privacy\_policy\_info)\}$
$event = \{\}$
$eventMap = \{\}$
$space = serverRoom$

Object : *shelf*

$identification = 9$
$class = Object$
$description =$ furniture
$position = (60, 70, 0)$
$mass = 9Kg$
$width = 120cm$
$height = 200cm$
$shape = parallelogram$
$emittedSignal = \{\}$

Location : *alice_location*

$symLocation = alice\_symbolic$
$relLocation = alice\_relative$

SymbolicLocation : *alice_symbolic*

$place = bookstore\_L$
$point = (65, 85, 0)$

RelativeLocation : *alice_relative*

$toObject = shelf$
$orientation = West$
$distance = 25$

LocationController : *location_controller*

$registeredElement = \{alice\}$
$location = \{(alice, alice\_location)\}$

EnergyController : *energy_controller*

$device = \{handheld\}$
$energy = \{(handheld, 95)\}$

TimeController : *time_controller*

$clock = 2h23m30s40 - 02/06/2007$
$scheduler = \{(2h24m05s - 02/06/2007, e[1]),$
$(2h24m10s - 02/06/2007, e[2]),$
$(2h50m00s - 02/06/2007, f[1]),$
$(3h00m00s - 02/06/2007, d[1]), ...\}$
$duration = \{(10m - 0h0m0s, d[1]), ...\}$
$frequency = \{(50m - 0h0m0s, f[1]), ...\}$

SituationalController : *situational_controller*

$situation = \{(\{shelf.stateOpened[true]\}, s[1]), ...\}$

CommunicationController : *communication_controller*

$activeEntities = \{handheld\}$
$activeChannels = \{com1, com2\}$
$connections = \{(com1, senIO), (com1, server), (com2, sen\_bookstore\_L),$
$(com2, handheld), ...\}$
$area : \{SquareArea[(50, 80)(60, 90)]\}$
$restrictArea : \{(SquareArea[(50, 80)(60, 90)], Interference)\}$

CommunicationChannel : *com*1

$physicalChannel = \text{cabe}$
$source = \{senIO, sen\_bookstore\_L, sen\_bookstore\_M\}$
$target = \{server\}$
$propagation =$
$range =$
$transmissionRate = 10Mbps$
$frequency = 350MHz$
$constraints = \{\}$

CommunicationChannel : *com*2

$physicalChannel = wireless$
$source = \{sen\_bookstore\_L\}$
$target = \{handheld\}$
$propagation = line$
$range = 5m$
$transmissionRate = 100kbps$
$frequency =$
$constraints = \{object\ obstruction, signal\ lost\}$

Door : *door*

$identification = 10$
$class = Object$
$description = \text{main door}$
$position = (100, 50, 0)$
$width = 10$
$height = 2$
$shape = rectangle$
$initialState = \{stateOpened.false\}$
$possibleState = \{stateOpened.false, stateOpened.true\}$
$changeState = \{(open, (stateOpened.false, stateOpened.true)),$
$(close, (stateOpened.true, stateOpened.false))\}$
$opaque = true$
$alwaysOpened = false$
$stateOpened = false$

Space : *server_room*

identification = 11
class = *Space*
description = server room
position = (40, 80, 0)
width = 10
height = 3
shape = *rectangle*
name = *server_room*
entities = {*server*}

Space : *bookstore_L*

identification = 12
class = *Space*
description = Bookstore L
position = (68, 60, 0)
width = 20
height = 3
shape = *rectangle*
name = *bookstore_L*
persons = {*alice*, ...}
objects = {*shelf*, ...}
entities = {*handheld*, *sen_bookstore_L*, ...}

Space : *bookstore_M*

identification = 13
class = *Space*
description = Bookstore M
position = (75, 25, 0)
width = 25
height = 3
shape = *rectangle*
name = *bookstore_M*
entities = {*sen_bookstore_M*, ...}

Environment : *shopping*

$name = shopping$
$space = \{server\_room, bookstore\_L, bookstore\_M\}$
$timeController = time\_controller$
$situationalController = situational\_controller$
$locationController = location\_controller$
$energyController = energy\_controller$
$communicationController = communication\_controller$

Profile : *profile_A*

$identification = 14$
$pseudonym = 4fasd452$
$personalProperties = \{sports, \{gymnastics\}\}, \{films, \{action\_movies\}\},$
$\{books, \{self - help\}\}$
$servicePolicies = marketing\_policy$
$mixingRate = 1$

Profile : *profile_B*

$identification = 15$
$pseudonym = alice$
$personalProperties = \{sports, \{gymnastics\}\}, \{films, \{action\_movies\}\},$
$\{books, \{self - help, traveling\}\}$
$servicePolicies = marketing\_policy$
$mixingRate = 0$

PrivacyPolicy : *marketing_policy*

$identification = 16$
$service = marketing\_service$
$serviceProvider = sen\_bookstore\_L, sen\_bookstore\_M$
$defaultMode = allow$
$unknownMode = deny$
$temporalConstraint : \langle(8h00m00s, 18h00m00s)\rangle$

Service : *marketing_service*

$identification = 17$
$description = MarketingService$
$created = 0h24m05s - 01/01/2007$