UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

UMA METODOLOGIA PARA ASSISTIR PACIENTES EM AMBIENTES *HOMECARE* PERVASIVOS

DISSERTAÇÃO DE MESTRADO

Leandro Oliveira Freitas

Santa Maria, RS, Brasil

2011

UMA METODOLOGIA PARA ASSISTIR PACIENTES EM AMBIENTES *HOMECARE* PERVASIVOS

por

Leandro Oliveira Freitas

Dissertação apresentada ao Programa de Pós-Graduação em Informática da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para a obtenção do grau de

Mestre em Computação

Orientador: Prof. Dr. Giovani Rubert Librelotto (UFSM)

Santa Maria, RS, Brasil

2011

Universidade Federal de Santa Maria
Centro de Tecnologia
Programa de Pós-Graduação em Informática

A Comissão Examinadora, abaixo assinada,
aprova a Dissertação de Mestrado

UMA METODOLOGIA PARA ASSISTIR PACIENTES EM AMBIENTES *HOMECARE* PERVASIVOS

elaborada por
Leandro Oliveira Freitas

como requisito parcial para obtenção do grau de
Mestre em Computação

COMISSÃO EXAMINADORA:

Giovani Rubert Librelotto (UFSM), Dr.
(Presidente/Orientador)

Marco Antônio de Castro Barbosa, Prof. Dr. (UTFPR)

Giliane Bernardi, Profª. Drª. (UFSM)

Santa Maria, 09 de Dezembro de 2011.

DEDICATÓRIA

Dedico este trabalho ao meu pai **Paulo E. de Freitas** que sempre foi o maior incentivador para que eu continuasse na vida acadêmica e ingressasse no curso de mestrado. Além de ter sido meu melhor amigo, sempre foi exemplo de persistência e força de vontade.

AGRADECIMENTOS

Agradeço primeiramente à minha família. Ao meu pai **Paulo E. de Freitas**, minha mãe **Ladir U. de Oliveira** e minha irmã **Alice O. Freitas**, que sempre me apoiaram durante esta caminhada e também pela paciência que tiveram comigo. Sei que muitas vezes, ao chegar em casa depois de um longo dia de trabalho, não fui a melhor companhia.

Agradeço também à minha namorada **Greice da Silva Zanini** pela força e compreensão nas horas em que não pude estar presente. Muito obrigado por todo seu incentivo, pelas palavras de carinho e conforto quando pensava que não teria condições de terminar o trabalho. Enfim, muito obrigado por fazer parte da minha vida e estar sempre ao meu lado.

Ao meu orientador, Prof. Dr. **Giovani R. Librelotto**, agradeço por ter me acolhido em seu projeto desde o início. Por me auxiliar no decorrer do curso e pelas conversas e trocas de ideias que tivemos, as quais foram de fundamental importância para conseguirmos finalizar o trabalho com qualidade.

Por fim, agradeço aos meus **colegas e amigos** pelo apoio, pelos trabalhos realizados em conjunto e também pelos momentos de descontração.

“Se você pensa que pode
ou se pensa que não pode,
de qualquer forma você está certo.”
— HENRY FORD

RESUMO

Dissertação de Mestrado
Programa de Pós-Graduação em Informática
Universidade Federal de Santa Maria

UMA METODOLOGIA PARA ASSISTIR PACIENTES EM AMBIENTES *HOMECARE* PERVASIVOS
AUTOR: LEANDRO OLIVEIRA FREITAS
ORIENTADOR: GIOVANI RUBERT LIBRELOTTO (UFSM)
Local da Defesa e Data: Santa Maria, 09 de Dezembro de 2011.

A cada ano que passa as filas de hospitais públicos e privados crescem devido, entre outros fatores, ao aumento da população mundial e a demora no atendimento de pacientes. Este é um problema sério enfrentado por administradores de hospitais os quais acreditam ser cada vez mais difícil oferecer um serviço de qualidade a quem os procura. Uma das formas de diminuir estas filas é através do desenvolvimento de sistemas de *homecare* que possibilitam que o paciente receba um tratamento clínico diretamente em sua casa. O desenvolvimento de sistemas deste tipo ajudaria a diminuir as filas e, consequentemente, iria melhorar o atendimento daqueles que forem até os hospitais a procura de auxílio. Considerando isto, este trabalho tem como objetivo principal apresentar uma modelagem para uma arquitetura de um sistema pervasivo para ser aplicado em ambientes de *homecare*. Os sistemas pervasivos desenvolvidos a partir desta modelagem visam o aprimoramento dos serviços prestados por profissionais da saúde no tratamento de pacientes que encontram-se em suas casas. A metodologia proposta apresenta uma arquitetura que utiliza conceitos de computação pervasiva possibilitando acesso à informação a qualquer hora e independente do lugar onde estejam, uma vez que um ambiente *homacare* possui um alto grau de dinamicidade dos envolvidos. A representação do conhecimento existente no ambiente *homecare* necessária para a modelagem da arquitetura é feita através de ontologias devido possibilidadde de reuso das informações ali armazenadas, bem como a interoperabilidade de informações entre diferentes dispositivos computacionais. Para validação da metodologia proposta, são apresentados dois casos de estudos, os quais também são utilizados para demonstração do fluxo de funcionamento do sistema pervasivo de *homecare*.

Palavras-chave: Computação Ubíqua, Computação Pervasiva, Ontologias, *Homecare*.

ABSTRACT

Master's Dissertation
Program in Computer Science
Universidade Federal de Santa Maria

A METHODOLOGY TO ASSIST PATIENTS IN PERVASIVE HOMECARE ENVIRONMENTS

AUTHOR: LEANDRO OLIVEIRA FREITAS
ADVISOR: GIOVANI RUBERT LIBRELOTTO (UFSM)
Defense Place and Date: Santa Maria, December 09th, 2011.

Every year the queues in hospitals publics and privates grows due to, among others, the increasing of the world population and the delay in the patients service. This is a serious problem faced by administrators of hospitals which believe that it is increasingly difficult to offer services of quality to those who search for them. One of the ways to decrease these queues is through the development of homecare systems that allow the patient to receive the clinic treatment directly in his house. The development of these kind of systems would help to decrease the queues and consequently, would improve the attendance of those who goes to the hospitals looking for assistance. Considering this, this work has as main purpose to present the modelling of an architecture of a pervasive system to be applied in homecare environments. The pervasive systems developed from this modelling aim to improve the services provided by healthcare professionals in the treatment of patients that are located in their houses. The architecture proposed by the methodology uses concepts of pervasive computing to provide access to information anytime and wherever the user is, once that a homecare environment has a high level of dynamicity of the envolved. The knowledge representation of the homecare environment needed in the modelling of the architecture is made through ontologies due to the possibility of reuse of the information stored, as well as the interoperability of information among different computational devices. To validate the proposed methodology, we present two use cases, which are also used to demonstrate the workflow of the pervasive system of homecare.

Keywords: Ubiquitous Computing, Pervasive Computing, Ontologies, Homecare.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

OWL	*Web Ontology Language*
PEP	*Prontuário Eletrônico de Pacientes*
RDF	*Resource Description Framework*
RDFS	*RDF Schema*
DTD	*Document Type Definition*
SWRL	*Semantic Web Rule Language*
SQWRL	*Semantic Query-Enhanced Web Rule Language*
XML	*Extensible Markup Language*
XSL	*eXtensible Stylesheet Language*
HTML	*HyperText Markup Language*
W3C	*World Wide Web Consortium*
SGML	*Standard Generalized Markup Language*
URI	*Uniform Resource Identifier*
DAML	*DARPA Agent Markup Language*
OIL	*Ontology Inference Layer*
ER	*Entidade-Relacionamento*
UML	*Unified Modeling Language*
SQL	*Structured Query Language*

SUMÁRIO

1 INTRODUÇÃO

Atualmente, um dos principais problemas enfrentados por administradores de hospitais é oferecer a garantia de um serviço de qualidade a seus clientes. A cada ano que passa, as filas em hospitais públicos e privados aumenta por diversos fatores como, por exemplo, alta demanda de paciente, problemas de infra-estrutura, comunicação interna falha e falta de compromisso profissional, o que compromente o tratamento clínico do paciente (ANDRADE et al., 2009).

Uma das formas de minimizar a demora no atendimento é através do desenvolvimento de aplicações computacionais que agilizem o fluxo de tratamento de pacientes. A grande maioria dos hospitais conta com sistemas informatizados para gerenciar o trabalho burocrático e administrativo; ainda assim, alguns deles também possuem aplicações que tornam o trabalho dos clínicos mais eficiente. Porém, estes sistemas foram projetados para serem utilizados em locais fixos, onde o usuário deve sentar-se em frente ao computador, em suas salas, para utilizá-lo (ZARGHAMI et al., 2011). Este tipo de aplicação não pode ser considerado como ideal, uma vez que um hospital é um ambiente que exige um alto grau de mobilidade dos profissionais, exige um trabalho colaborativo entre especialistas de diferentes áreas, além de ser um ambiente propício para ocorrer frequentes interrupções nas atividades que estes profissionais realizam.

Outra alternativa é inverter a forma como o paciente é tratado hoje. Ao invés dele ir até o hospital para receber atendimento, os serviços clínicos vão até a casa do paciente. Em outras palavras, o paciente recebe atendimento médico diretamente em sua casa, através da implantação de aplicações de *homecare*. Um ambiente *homecare* é caracterizado por possibilitar que o paciente receba atendimento médico no conforto de sua casa, seja por necessidade, como em casos onde o paciente é impossibilitado de locomoção, ou até mesmo por vontade própria, uma vez que ele ficará mais tempo junto de seus familiares e amigos.

Porém, por se tratar da casa do próprio paciente, um ambiente *homecare* pode vir a se tornar bastante dinâmico, uma vez que o paciente pode sair e entrar no ambiente quando quiser, além de circular pelos cômodos da casa ou ainda receber visita de parentes e amigos. Contudo, apesar de estar em casa, o paciente deve estar sob monitoramento constante, e qualquer piora repentina de seu estado de saúde, deve ser tratada imediatamente. Desta forma, um sistema com aplicações médicas para *homecare* deve suportar estas mudanças frequentes no contexto do ambiente.

Neste cenário, a computação pervasiva aparece como uma solução para sanar problemas

como estes da mobilidade e dinamicidade que caracterizam um ambiente *homecare*. A computação pervasiva é vista hoje como a terceira onda da computação (SHIBU; SANJEEV, 2010) onde tem-se vários dispositivos computacionais trabalhando em prol de um ou mais usuários. Um ambiente pervasivo deve possibilitar a interação entre estes dispositivos e usuários da forma mais intuitiva possível, ou seja, o usuário foca totalmente sua atenção na tarefa que está realizando, sem sequer perceber que está utilizando computadores para tal. Além disso, a computação pervasiva visa proporcionar acesso à informação a qualquer hora e lugar, desde que o usuário possua recursos computacionais para recebê-las.

Em um ambiente *homecare* pervasivo, esta computação pode proporcionar aos clínicos um acesso rápido e seguro às informações dos pacientes mesmo se estas não estiverem em seu local de trabalho. Desta forma, é possível tomar decisões mais rapidamente sobre que tratamento aplicar a pacientes cujo estado de saúde é mais grave.

Para que isto seja possível, é preciso uma representação detalhada do conhecimento existente nesse domínio. Uma das maneiras mais adequadas para esta representação é através de ontologias. As ontologias podem ser definidas como uma representação formal de uma contextualização. Uma ontologia é composta por entidades e relacionamentos, onde relacionamentos são definidos formalmente e sua semântica é detalhada. Logo, se estes relacionamentos possuírem nomes que identificam seus significados, um humano poderá entendê-la diretamente; assim como um programa pode assumir uma semântica de um relacionamento e atuar através da mesma.

Devido ao fato de que vários dispositivos computacionais devem trabalhar em conjunto, um ambiente pervasivo pode se tornar bastante heterogêneo, ou seja, cada dispositivo pode utilizar uma plataforma de processamento diferente, o que dificulta a comunicação entre os mesmos. As ontologias garantem a interoperabilidade das informações que trafegam no ambiente, de forma que estas serão entendidas pelos dispositivos envolvidos de maneira objetiva, sem qualquer ambiguidade.

Utilizando conceitos de computação pervasiva e ontologias, o presente trabalho possui três objetivos principais, os quais estão interligados entre si. Primeiramente, pretente-se encontrar uma forma de auxiliar os profissionais da saúde em suas tarefas diárias. Com isto, o fluxo de tratamento clínico de pacientes que procuram ajuda será melhorado. Com estes dois objetivos alcançados, o último, que refere-se ao aprimoramento do atendimento a pacientes que recebem cuidados médicos em hospitais ou em casa, também será alcançado.

Para atingir estes objetivos, este trabalho apresenta uma metologia para uma arquitetura para sistemas pervasivos com aplicações médicas em ambientes *homecare*.

A partir desta arquitetura é possível desenvolver sistemas pervasivos para auxiliar profissionais no tratamento de pacientes. Estes sistemas possibilitam uma comunicação constante entre a casa do paciente e a clínica que está oferecendo o serviço. Assim, se um paciente tem uma piora repentina, o sistema comunica imediatamente o profissional de plantão na casa do paciente além de avisar o hospital (ou clínica) caso necessite de apoio mais especializado. Isto é possível pois o ambiente *homecare* está dotado de sensores que fazem um monitoramento constante do paciente enviando informações sobre o estado de saúde do mesmo para o sistema pervasivo.

Outro problema resolvido com aplicações desenvolvidas para ambientes *homecare* é o da super-lotação dos hospitais. Uma vez que exista a possibilidade do paciente receber o tratamento diretamente em sua casa, com a mesma qualidade do hospital, as filas irão diminuir consideravelmente e, consequentemente, os pacientes que forem até os hospitais terão um atendimento de melhor qualidade.

Por último, com este sistema pretende-se agilizar o fluxo de tratamento de pacientes em suas casas, através da sugestão de possíveis próximas ações que os profissionais possam querer executar com relação ao tratamento do paciente.

De forma a apresentar a metodologia proposta, o documento está organizado da seguinte forma. No Capítulo 2 são descritos os conceitos e características da computação pervasiva, bem como a importância de sua aplicação na área da saúde. A seguir, no Capítulo 3 são apresentados conceitos de ambientes *homecare*, bem como a descrição das características destes sistemas, e também os desafios a serem superados para que ambientes pervasivos de *homecare* se consolidem no mercado.

No Capítulo 4 são detalhados conceitos de ontologias, áreas de utilização e classificação, bem como as formas e linguagens de representação e consultas. Uma ontologia que representa o conhecimento de ambientes *homecare* foi desenvolvida e está descrita no Capítulo 5.

A metodologia proposta para ambientes *homecare* é apresentada no Capítulo 6, através da descrição de cada um dos módulos que a compõem, bem como o fluxo de funcionamento da mesma. Com o objetivo de validar a metodologia proposta, no Capítulo 7 é utilizado um caso de estudo já publicado na literatura e outro criado para demonstrar o fluxo de funcionamento do sistema, além da descrição dos resultados obtidos.

A comparação entre o trabalho apresentado e outros existentes na literatura é descrita no Ca-

pítulo 8. Por último, no Capítulo 9 apresenta-se as conclusões do trabalho, através da descrição das contribuições, trabalhos futuros e publicações já realizadas.

2 COMPUTAÇÃO PERVASIVA

A computação pervasiva é vista como o novo paradigma computacional do século XXI e tem como principal objetivo disponibilizar recursos e informações aos usuários a qualquer hora e lugar (SAHA; MUKHERJEE, 2003). Assim, este capítulo apresenta uma revisão sobre diversos conceitos envolvidos em computação pervasiva, assim como a descrição de suas características e algumas aplicações na área da saúde.

Em um ambiente pervasivo, o computador deve estar disposto de forma transparente, ou seja, o usuário utiliza seus serviços sem perceber que está utilizando um computador. Além disso, estes dispositivos têm a capacidade de obter informações do ambiente e utilizá-las para construir modelos computacionais dinamicamente, ou seja, controlar, configurar e ajustar a aplicação para melhor atender as necessidades do usuário. O ambiente deve ser capaz de detectar outros dispositivos que possam vir a fazer parte dele. A partir desta interação, computadores passam a agir de forma inteligente em um ambiente povoado por sensores e serviços computacionais (ARAUJO, 2003).

A computação pervasiva pode ser descrita como ambientes inteligentes cheios de dispositivos computacionais que interagem com usuários de forma tão natural a ponto de se tornarem parte deste ambiente. Porém, quando o visionário Mark Weiser (WEISER, 1999) fez este relato, os recursos de hardware necessários para sua idealização eram muito restritos e tinham um custo muito alto.

Weiser, em seu artigo intitulado "*The origins of ubiquitous computing research at PARC in the late 1980*", descreveu a Computação Ubíqua como uma computação onipresente, onde recursos dispostos em um ambiente inteligente se auto-ajustariam de forma a melhor atender as necessidades dos usuários. Embora a idéia original de Weiser quanto a "Computação Ubíqua" ainda esteja distante de uma prática cotidiana alicerçada por produtos de mercado (SATYANARAYANAN, 2001), sua proposta vem se tornando realidade com a disponibilização de tecnologias como *PDAs*, *Smartphones* e a consolidação de padrões para redes sem fio como o *Bluetooth* e o IEEE 802.11 (GASSEN et al., 2008).

Atualmente, os recursos de hardware disponíveis no mercado já possuem um tamanho consideravelmente menor a ponto de se tornarem portáveis, com um poder de processamento maior e baixo custo. O avanço desta tecnologia de comunicação sem fio começa a tornar possível a disponibilização da informação a qualquer hora e em qualquer lugar.

Os recursos dispostos em um ambiente pervasivo devem ser sensíveis ao contexto, realizando freqüentes varreduras neste ambiente à procura de possíveis mudanças que possam ocorrer e adaptando-se à estas mudanças, trabalhando de forma dinâmica. A adaptação é fundamental, pois um ambiente inteligente pode vir a se tornar bastante heterogêneo, possuindo uma grande variedade de dispositivos com diferentes plataformas de processamento.

A computação pervasiva propõe um modelo computacional que integra de forma transparente os dispositivos de *hardware* e *software* existentes hoje, a fim de proporcionar aplicações que facilitam o desenvolvimento das tarefas do usuário quando ele se encontra em um ambiente inteligente.

2.1 Computação sensível ao contexto

Um dos aspectos fundamentais da computação pervasiva é a sensibilidade do contexto (*context-aware*), onde dispositivos computacionais de um determinado ambiente devem ter a capacidade de detectar as mudanças que possam ocorrer neste ambiente, adaptando-se a elas e assim procurar atender o usuário da melhor forma possível.

A computação sensível ao contexto tem como objetivo permitir que aplicações sejam capazes de acessar e capturar informações relacionadas aos processos que realizam em um determinado ambiente, a fim de aperfeiçoar seu desempenho. Em outras palavras, pode-se dizer que ela está ligada à capacidade que os sistemas computacionais tem de obter informações de um ambiente e armazená-las junto com as que já possui para melhorar seu desempenho na realização de tarefas complexas. Assim, além de trabalhar com entradas explícitas, a computação sensível ao contexto também lida com entradas implícitas, analisando a infra-estrutura do ambiente, preferências do usuário, atividade do usuário, número e tipos de dispositivos, carga computacional, entre outros (CHEN; KOTZ, 2000).

Diversos trabalhos relacionados à sensibilidade ao contexto foram realizados durante a década de 1990 (ABOWD et al., 1997), tendo como enfoque principal a localização do usuário. O *Active Badge Location System* (WANT; FALCAO; GIBBONS, 1992) é considerado o primeiro sistema sensível ao contexto. Seu sistema era baseado na tecnologia de infravermelho e conseguia determinar a localização do usuário para que uma ligação fosse encaminhada para o telefone mais próximo a ele.

Um dos aspectos mais importantes quando se fala em sistemas móveis, é a localização de dispositivos e usuários no ambiente, pois ela influi significativamente no contexto disponibili-

zado para os mecanismos de adaptação. O contexto representa uma abstração peculiar da computação pervasiva, e inclui informações sobre recursos, serviços e outros componentes do meio físico de execução. Assim, devem ser obtidas outras informações de contexto que extrapolam a localização onde o componente móvel da aplicação se encontra (YAMIN, 2004).

Com o surgimento da nova geração de dispositivos móveis (PDAs, *Smartphones*), a computação sensível ao contexto ganhou uma atenção especial. Porém, o desenvolvimento de ferramentas que dão suporte a ela traz à tona diversos desafios como a obtenção, modelagem, armazenamento, distribuição e monitoramento do contexto.

2.1.1 Definição de contexto

Entende-se por contexto tudo o que faz parte de um determinado ambiente, incluindo suas mudanças. O contexto pode ser descrito como o local, a identidade de pessoas neste local, os objetos e as mudanças que podem ocorrer nele (SCHILIT; THEIMER, 1994).

De acordo com (YAMIN et al., 2003), o contexto pode ser definido como qualquer informação gerada pela infra-estrutura computacional do ambiente que pode ser relevante para uma aplicação, na qual uma alteração pode disparar um processo para a adaptação desta aplicação. Portanto, o contexto deve ter a capacidade de filtrar informações do ambiente, descartando as que não interessam e, com isto, a aplicação consegue definir as entidades que caracterizam uma situação e estas passam a fazer parte do seu contexto.

Na computação móvel, o contexto é formado por dois aspectos (CHEN; KOTZ, 2000). O primeiro se refere às características do ambiente que determinam o comportamento das aplicações. O outro aspecto diz que o contexto é um conjunto de estados e configurações do ambiente que determina um comportamento da aplicação.

2.1.2 Arquiteturas de aplicações sensíveis ao contexto

Existem diversas formas de implementação de aplicações sensíveis ao contexto. A abordagem depende dos requisitos e condições existentes, como a localização dos sensores, o número de usuários, a disponibilidade dos recursos a serem utilizados e a facilidade para a extensão do sistema. Considerando isto, três tipos arquiteturas podem ser citados (CHEN, 2004):

- Acesso direto ao sensor: esta abordagem vem sendo pouco utilizada, já que dificulta a capacidade de expansão do sistema. Isto acontece, pois o software cliente pega a informação diretamente do sensor, ou seja, não existe uma camada específica para o processamento

dos dados. Além disto, também não é adequada para sistemas distribuídos, por possuir acesso direto, não há um componente para gerenciar acessos concorrentes ao sensor;

- Baseado em *Middleware*: esta abordagem traz uma arquitetura em camadas para aplicações sensíveis ao contexto que visam esconder detalhes de baixo nível relativos à sensibilidade. Em comparação com a anterior, a abordagem baseada em *Middleware* facilita a capacidade de expansão do sistema, pois não há necessidade de mudanças no código fonte do cliente, bem como há uma simplificação para a reutilização do código dependente de hardware, devido a seu rígido encapsulamento;

- Servidor de contexto: é uma abordagem especializada da arquitetura baseada em *Middleware*, pois introduz um componente para gerenciar o acesso remoto. Um servidor de contexto recebe as informações dos sensores, facilitando o acesso concorrente. Além do reuso dos sensores, a utilização de um servidor de contexto tem a vantagem de retirar dos clientes operações que necessitam uso intensivo de recursos computacionais. Isto é importante, pois a maioria dos dispositivos utilizados neste tipo de aplicação possui um poder computacional limitado. Por outro lado, ao projetar um sistema sensível ao contexto baseado em arquitetura de cliente-servidor, é preciso levar em consideração o uso de protocolos apropriados, analisar o desempenho de rede e avaliar parâmetros de qualidade de serviço.

É necessário fazer a separação entre a obtenção e o uso de um contexto para melhorar a capacidade de expansão e reutilização dos sistemas. Assim, é possível incluir camadas em uma arquitetura abstrata tornando possível captar e usar informações do contexto através da adição de funcionalidades de interpretação e raciocínio.

2.2 Computação pervasiva aplicada à saúde

Em (BARDRAM, 2004), foi discutida a importância de integrar a sensibilidade de contexto aos sistemas de Prontuário Eletrônico de Pacientes (PEP) convencionais, fazendo com que os mesmos consigam se adaptar às possíveis mudanças que possam ocorrer no ambiente. Os autores apresentam um protótipo de uma cama de hospital ciente às mudanças no contexto. Ela possui uma tela que, na maior parte do tempo, é utilizada como um aparelho de TV normal para entreter o paciente. Porém, quando um médico ou enfermeiro entra no recinto a cama, através de seus sensores, detecta esta mudança no ambiente e passa a mostrar na tela informações rele-

vantes ao profissional no tratamento do paciente. Estas aplicações passaram por avaliações em uma série de workshops realizados com médicos e pacientes.

Outro trabalho interessante é (BOSSEN; JØRGENSEN, 2004), onde os autores descrevem um protótipo de contexto-descritivo de processos realizados durante a administração de medicamentos, além de seu suporte por um novo sistema pervasivo. Um dos objetivos do projeto é tornar mais fácil o acesso as informações presentes no sistema PEP. Para isto, essas informações devem serem acessadas a partir de qualquer dispositivo móvel. Foi então proposto o desenvolvimento de um sistema pervasivo de cuidados médicos (PHCS).

O artigo aborda o processo de administração de medicamentos (dosagem e distribuição), que envolve diferentes ambientes, partindo da sala de medicamentos onde os remédios são selecionados até o quarto do paciente onde o medicamento é administrado; e seu suporte pelo sistema PHCS. Com a identificação eletrônica (RFID) de pacientes, profissionais e até mesmo bandejas de medicamentos, pretende-se agilizar este processo. Ao entrar na sala de medicamentos, o PHCS identifica o profissional e possibilita através de uma interface que o profissional (um enfermeiro ou técnico em enfermagem) possa acessar diretamente a lista de pacientes que estão sob a sua supervisão ou fazer logon no sistema. Essas sugestões são baseadas no contexto da sala de medicamentos.

Os projetos relacionados acima se assemelham ao apresentado neste trabalho de dissertação pelo fato de também aplicar as tecnologias referentes a computação pervasiva à saúde. Porém, o que os diferencia é o fato de que o foco deste trabalho é *homecare*, ou seja, levar a computação pervasiva à pacientes que recebem tratamento em casa, bem como a seus médicos, e não aplicá-la à hospitais.

2.3 Considerações do capítulo

Este Capítulo apresentou detalhadamente características relacionadas a computação pervasiva, bem como as arquiteturas que aplicações sensíveis ao contexto devem possuir e também a importância da utilização da computação pervasiva e ubíqua na área da saúde, principalmente em hospitais e ambientes de *homecare*. A partir das discussões levantadas aqui, pode-se concluir que, para o desenovolvimento de aplicações voltadas para *homecare*, é necessário que este tipo de ambiente possua características da computação pervasiva, permitindo assim, o tratamento do paciente sem interferir em suas tarefas rotineiras. A seguir será detalhada a idéia de ambientes *homecare* pervasivos.

3 *HOMECARE*

O número de pessoas doentes e idosos que preferem receber cuidados médicos em casa vem aumentando consideravelmente nos últimos anos (MCGEE-LENNON, 2008). A modificação neste cenário traz dois grandes benefícios. O primeiro deles é o benefício social, visto que os pacientes podem permanecer mais tempo junto a seus familiares e amigos. Também há o beneficio econômico, uma vez que está cada vez mais difícil para os hospitais fornecerem um serviço de qualidade com baixo custo a seus clientes, devido ao aumento considerável no número de idosos (ISERN et al., 2009).

As tecnologias existentes hoje possibilitam que pacientes recebam tratamento à domicílio, porém é preciso desenvolver aplicações confiáveis para que este tipo de serviço seja aceito pela sociedade. Estas tecnologias podem ser utilizadas para monitorar pacientes que estão incapazes de locomoção e que possam precisar de uma intervenção externa (MISKELLY, 2004). Os avanços no desenvolvimento de tecnologias de rede facilitam a troca de informações entre paciente e equipe médica, mesmo quando eles não se encontram não mesmo ambiente de tratamento. Este tipo de aplicação, por natureza, envolve diferentes grupos de pessoas, desde usuários a investidores, onde todos têm interesse e são potencialmente capazes de interferir em como o sistema deve se comportar sob diferentes situações.

3.1 Serviços de *homecare*

O termo *homecare* pode ser definido como um conjunto de serviços conectados que possibilitam o fornecimento de cuidados médicos à pacientes em suas casas. Estes cuidados podem ser de ordem social, de saúde ou ambos. Um sistema de *homecare* pode oferecer uma vasta gama de serviços, variando de aplicações simples como, por exemplo, alarmes eletro-mecânicos independentes instalados na casa do paciente e que o avisam quando uma banheira está transbordando ou uma porta ficou entreaberta, até sistemas integrados à infraestrutura da casa que realizam um monitoramento constante do estado de saúde do paciente e que tem a capacidade de realizar análises sofisticadas, trocar informações customizadas entre médico e paciente e suportar uma comunicação remota entre eles (MCGEE-LENNON, 2008).

3.2 Redes de *homecare*

Existem muitas entidades envolvidas em um sistema de *homecare*, as quais podem ser divididos da seguinte forma (MCGEE-LENNON, 2008):

- O(s) paciente(s): é o centro de um sistema de *homecare*. É possível que um único paciente esteja sob cuidados médicos para tratamento de mais de um problema. Também é possível que em um mesmo ambiente de *homecare* exista mais de um paciente que precisem de cuidados médicos como, por exemplo, um casal de idosos onde o marido tem problemas cardíacos e a esposa tem câncer;

- Os cuidadores: são definidos como as pessoas que fornecem a primeira assistência ao paciente, sejam médicos, enfermeiras ou familiares. Muitas vezes essa assistência é feita por um membro da família e isso pode se tornar bastante complexo quando o cuidador também precisa de cuidados, ou seja, quando um paciente depende da ajuda de outro paciente. Em situações como essa onde podem existir diferentes usuários finais, é necessário que o sistema *homecare* se comporte de maneiras distintas de acordo com as diferentes situações, ou seja, levando em consideração o contexto atual do ambiente. Podem existir casos onde os cuidadores queiram acessar remotamente informações sobre o paciente como, por exemplo, quando o médico está em seu consultório e precisa de uma informação atualizada, ou então se os cuidadores são familiares e estes não moram junto com o paciente, mas querem saber sobre seu estado de saúde;

- Visitantes: são os profissionais chamados para dar uma assistência. Neste grupo são incluídos, por exemplo, enfermeiras e paramédicos. Muitas vezes a diferença ente os visitantes e cuidadores não é bem definida. Se um profissional tem uma rotina de visitas ao paciente, de forma planejada, então ele é classificado como cuidador. Porém, se a visita ocorre de forma inesperada como em um atendimento a um paciente que está tendo um infarto, o profissional é classificado como visitante. Pessoas que entram no ambiente de *homecare* simplesmente para visitar o paciente também se encaixam neste grupo, desde que a visita não seja regular;

- Usuários remotos: neste grupo estão incluídos membros da equipe médica que acessam as informações do paciente mesmo sem se deslocar até a casa dele. Por muitas vezes, um profissional pode estar incluído neste grupo e também no grupo de visitantes. Além

destes, também estão incluídos aqui os familiares que não moram com o paciente, mas que querem estar cientes sobre seu estado de saúde e para isso acessam as informações remotamente;

- Fornecedores de tecnologia: são profissionais que projetam, desenvolvem e distribuem a tecnologia individual para o sistema de *homecare* e que podem impactar de alguma forma em como o sistema deve funcionar;

- Investidores institucionais: são pessoas que não são usuários diretos do sistema, mas que podem influenciar de alguma maneira em sua forma ou conteúdo. O ministro da saúde se encaixa neste grupo quando decide criar uma iniciativa para promover sistemas de *homecare* para um determinado propósito como, por exemplo, para tratar enfisema pulmonar;

- Demais investidores: pessoas que não usarão diretamente o sistema, mas que tem algum tipo de interesse nele. Neste grupo estão os membros da família do paciente que não moram com o paciente, mas que tem interesse em seu estado de saúde, ou então que têm alguma obrigação financeira ou moral com relação a ele.

3.3 Sistemas *homecare*

Atualmente já existem diversas tecnologias disponíveis no mercado capazes de suportar aplicações voltadas para quem precisa ou prefere receber cuidados médicos em suas próprias casas. De acordo com (MCGEE-LENNON, 2008), os sistemas voltados para *homecare* devem possuir algumas características básicas, as quais são descritas a seguir.

Sensores devem ser capazes de fornecer informações sobre o estado atual do paciente. O sistema deve levar em conta o contexto atual do ambiente em que o paciente se encontra como, por exemplo, sua localização na casa ou então que atividades está realizando em determinado momento. Estes sensores também devem ser capazes de monitorar o estado de saúde do paciente, fornecendo informações de ordem fisiológica como temperatura, pressão sanguínea e batimentos cardíacos.

Em um sistema de *homecare*, qualquer pessoa que possa vir a utilizar o sistema de alguma forma deve ser considerado um usuário em potencial. Desta forma, um sistema deve ser multi-usuário. Neste grupo encontram-se os pacientes e os profissionais responsáveis por seus cuidados médicos. Os pacientes são vistos como fonte de informações e podem utilizar serviços do sistema para gerenciar informações personalizadas de sintomas e criar alertas de condições

médicas. Por sua vez, os profissionais não usarão o sistema diretamente, como o paciente, mas de alguma maneira podem utilizar o sistema como forma de auxílio em seu trabalho.

Um ambiente de *homecare* deve possuir um conjunto de sensores que captam informações de contexto e com elas criam um cenário atual do ambiente. Este cenário construído pode conter informações sobre algum tipo de situação anormal do paciente ou que precise de intervenção externa. Desta forma, um sistema de *homecare* deve ser distribuído possibilitando acesso remoto de informações do ambiente e dessa forma criar um elo de ligação entre o que está acontecendo em casa com o paciente e o mundo externo. Isto permite que membros da equipe médica consigam responder a alertas de emergência com mais rapidez, uma vez que não será necessário esperar pela visita de rotina do médico para que ele detecte que existe algo errado com o paciente. Assim, através da utilização de tecnologias de redes de computadores e gerenciamento de *software*, é possível detectar situações adversas no ambiente e informar os envolvidos, sejam eles familiares e amigos ou então médicos, para que as providencias sejam tomadas mais rapidamente de forma a melhor atender o paciente (MISKELLY, 2004).

Em muitos casos o paciente que recebe tratamento em casa possui algum tipo de limitação, seja de ordem física ou psicológica. Dessa forma, um sistema de *homecare* deve ter suporte a diferentes formas de interação com o usuário. Deve possibilitar, por exemplo, o uso da fala para que pacientes com movimentos do corpo comprometidos consigam interagir com o sistema; a utilização de gestos como forma de entrada de dados também deve ser levada em consideração e; formas de interação táteis para usuários deficientes visuais. Além disso, a equipe médica e familiares devem conseguir receber informações em formato gráfico em seus *smartphones* ou então visualizá-los em telas digitais. É importante que os usuários possam escolher entre as diferentes formas de interação com o sistema dependendo do contexto atual do ambiente.

Devido ao fato de ter grande parte do seu funcionamento baseado no contexto do ambiente, um sistema de *homecare* deve ter aplicações dinâmicas. Além da possibilidade de o estado de saúde do paciente poder mudar repentinamente, os outros usuários podem possuir diferentes desejos, necessidades e responsabilidades com o passar do tempo (GARDE; KNAUP, 2006). Por exemplo, diferentes usuários podem preferir analisar os mesmos resultados de exames de formas diferentes, um através de texto e outro através de imagens. Um sistema desenvolvido com a possibilidade de atender as necessidades de cada usuário de forma individual resulta em uma melhor qualidade do serviço e, conseqüentemente, maior aceitação do que um sistema com um único tipo de resposta para os usuários. Geralmente as pessoas responsáveis pelos cuidados

do paciente, sejam familiares ou médicos, acreditam ter um conhecimento único das necessidades e preferências do paciente, dessa forma deveriam ter o poder de gerenciar a forma como os serviços são executados, além de realizar mudanças de acordo com as novas necessidades do paciente.

Analisando as características acima pode-se afirmar que ao utilizar padrões de execução de tarefas cria-se uma garantia de uma maior qualidade dos serviços de cuidados médicos do paciente. No entanto, também é sabido que há muitos problemas que dificultam a utilização destes padrões por profissionais, tais como uma falta de conhecimento sobre tais padrões ou então dificuldades inerentes à mudança de hábitos no comportamento diário destes profissionais (CABANA et al., 1999). Desta forma, o desenvolvimento de aplicações pervasivas para *homecare* deve também utilizar estes padrões de execução de tarefas como garantia de oferecer um serviço de qualidade aos usuários.

Considerando estas características e requisitos que um sistema de *homecare* deve possuir ao ser implementado, é natural que alguns obstáculos aparecem dificultando seu desenvolvimento. Dessa forma, a seção seguinte apresenta alguns desafios a serem superados para que os sistemas de *homecare* sejam consolidados.

3.4 Desafios de *homecare*

No processo de implementação de um sistema de *homecare* podem aparecer diversos conflitos gerados pelas diferentes preferências e necessidades do usuário, bem como com relação aos requisitos e configurações por parte da equipe de desenvolvimento e seus investidores. Alguns conflitos podem aparecer quando um usuário não consegue entender corretamente as necessidades e intenções dos outros usuários ou se ele interage de forma errada com o sistema. Para evitar que estes problemas causem um dano maior ao funcionamento do sistema, eles devem ser identificados e descritos de forma que suas estruturas e características revelem uma possível resolução. A seguir são descritos alguns dos conflitos que podem ser encontrados em um ambiente de *homecare*, de acordo com trabalho realizado por (MCGEE-LENNON, 2008).

É bastante comum, principalmente quando se trata de pacientes idosos, que eles sofram de mais de uma doença sendo que estas ainda podem estar interligadas. Desta forma, um sistema de *homecare* deve ter a capacidade de lidar com situações onde as informações colhidas pelos sensores podem conflitar entre si, e assim, muitas vezes nebular qual seria a aplicação correta a ser disparada. Assim, o sistema precisa analisar corretamente as informações recebidas para as-

sim executar o serviço correto. Uma informação interpretada erroneamente pode comprometer seriamente a saúde do paciente.

Muitos problemas de ordem política e social podem aparecer durante a especificação de um sistema de *homecare*, principalmente se existem investidores diferentes com suas prioridades e necessidades específicas. Um exemplo disso acontece quando o sistema deve decidir quem tem prioridade mais alta em situações onde mais de um usuário precisa utilizar os mesmos recursos do sistema.

Outro desafio a ser superado por um sistema desenvolvido para um ambiente de *homecare* está relacionado com a forma em que informação deve ser apresentada aos usuários. Por exemplo, se o paciente encontra-se imóvel sobre a cama, ele terá que as receber informações sobre seu estado de saúde na forma de áudio ou então visualizá-las em uma TV que antes era usada para entreter os usuários do ambiente. Estas formas de exibição podem irritar as outras pessoas que estão no ambiente, sejam elas membros da equipe médica, familiares ou amigos. Ou ainda, causar algum desconforto ao paciente, já que suas informações estarão sendo compartilhadas por outras pessoas, que a princípio não deveriam ter acesso àquela informação, como é o caso de visitantes.

Em muitos casos, um sistema de *homecare* deve suportar serviços que não condizem com o que os pacientes acreditam ser necessário levando em conta suas condições de saúde ou então com o que os responsáveis pelos seus cuidados pensam ser fundamentais. Dessa forma, cabe aos investidores e desenvolvedores encontrarem a forma de funcionamento menos agressiva aos olhos dos usuários, sem prejudicar o desempenho do sistema.

Um dos requisitos fundamentais para a implementação de um sistema de *homecare* é a questão da privacidade e segurança. Levando em consideração que o paciente tem total direito sobre as informações de sua ficha médica, é importante encontrar uma forma de manter estas informações em sigilo, permitindo sua visualização apenas por pessoas autorizadas. Assim, além dos pacientes, neste grupo encontram-se os membros da equipe médica e os responsáveis pelos cuidados do paciente e outras pessoas que tenham sido autorizadas pelo mesmo. Se mais de um usuário quer acessar ou controlar informações do paciente a qualquer momento, então é necessário que se tenha regras sobre a utilização destas informações.

Também deve ser levado em consideração que as necessidades do usuário podem mudar de acordo com o contexto atual do ambiente (FICKAS, 2005). Estas mudanças no contexto podem abranger modificações nas condições médicas do paciente, novos dispositivos inseridos

no ambiente, circunstâncias familiares ou apenas no modo como os pacientes se comportam.

Levando em consideração os problemas apresentados, muitas vezes podem aparecer dúvidas sobre de quem é a responsabilidade sobre os problemas relacionados a integridade dos dados ou então sobre o funcionamento correto dos dispositivos do sistema. Este tipo de problema aparece principalmente quando existem diversos investidores envolvidos no desenvolvimento e configuração do sistema, sejam eles institucionais ou particulares (MCGEE-LENNON, 2008).

3.5 Considerações do capítulo

A partir das questões levantadas neste Capítulo como, características de serviços e sistemas, possíveis entidades envolvidas e os desafios ainda a serem superados, pode-se concluir que para a construção de um ambiente *homecare* com características pervasivas é preciso um estudo detalhado sobre o domínio no qual os envolvidos estarão inseridos para que seja possível desenvolver aplicações que realmente possam auxiliar no tratamento de pacientes diretamente em suas casas, sem que isto comprometa seu estado de saúde.

A seguir serão discutidas questões relacionadas a ontologias, tais como formas utilização, classificação, representação e consultas.

4 ONTOLOGIAS

Uma ontologia pode ser definida como uma especificação formal e explícita de uma conceituação compartilhada (GRUBER, 1993), onde: (a) *conceituação* se refere ao modelo abstrato do mundo real; (b) *explícita* significa que os conceitos e seus requisitos são definidos explicitamente; (c) *formal* indica que a ontologia é processável por máquina, permite raciocínio automático e possui semântica lógica formal; (d) *compartilhada* significa que uma ontologia captura o conhecimento apresentado não apenas por um único indivíduo, mas por um grupo. De outra forma, pode-se dizer que ontologia é um modelo de dados que representa o conhecimento de um domínio através dos conceitos e relacionamentos entre eles.

Outra definição propõe o compartilhamento e reuso de ontologias. Esta definição fala de uma proposta de uso de ontologias para modelar problemas e domínios, onde elas fornecem uma biblioteca para fácil reutilização de classes e objetos para a modelagem. O objetivo desta proposta é o desenvolvimento de uma biblioteca de ontologias que poderia ser reutilizada e adaptada para diferentes classes de problemas e ambientes (DEPARTMENT; GRUNINGER, 1996).

A diferença entre ontologias e outros modelos de dados é que sua principal preocupação está voltada aos conceitos e seus relacionamentos, no qual a semântica destes relacionamentos é aplicada uniformemente (LIBRELOTTO et al., 2011).

Uma ontologia não se restringe a definição de um vocabulário sobre um determinado domínio, mas também possui relacionamentos e restrições entre os conceitos definidos no vocabulário. Os relacionamentos de uma ontologia são definidos formalmente e a semântica é feita de forma detalhada. Se os relacionamentos possuírem nomes apropriados, um humano ao visualizar a ontologia entenderá seu significado diretamente; assim como um programa pode assumir a semântica de um relacionamento e atuar sistematicamente através da mesma.

Um exemplo básico de relacionamento é o hierárquico “é um” que define que um conceito é de um determinado tipo específico. Essas especificações definidas com relacionamentos hierárquicos são denominadas taxonomias. Podem também existir relacionamentos do tipo “tem interesse em” entre os conceitos pessoa e interesse, sem que se trate de um relacionamento hierárquico (KOH; MUI, 2001).

As restrições em uma ontologia são denominadas axiomas e servem para criar uma restrição para um conceito baseando-se em um relacionamento. Por exemplo, em uma ontologia que

possua pessoas, existirá um relacionamento chamado “tem nome” que liga os conceitos nome e pessoa, e este relacionamento terá uma restrição que diz que “uma pessoa tem exatamente um nome”.

Uma ontologia pode também possuir regras de inferência que são capazes de chegar a conclusões baseando-se em fatos existentes nela. Por exemplo, uma ontologia em que “parente” é um relacionamento mais genérico que “pai”. Se Jorge é pai de Ana, o sistema deverá concluir automaticamente que Jorge também é parente de Ana. Portanto, se um usuário realizar uma pesquisa para saber quem são os parentes de Ana, Jorge estará no resultado, mesmo que este fato não tenha sido declarado.

Ontologias podem compartilhar certas semelhanças em sua estrutura, independente da linguagem em que são expressas. Entre estas semelhanças, é possível citar três elementos que são tidos como base para a construção de uma ontologia: indivíduos, classes e propriedades (FREITAS et al., 2008).

Os indivíduos, também chamados de instâncias, são considerados elementos-chave para a construção de uma ontologia. As classes são vistas como um conjunto que agrega indivíduos semelhantes e podem ser divididas em subclasses. Por sua vez, as propriedades têm a função de definir atributos, como nome, idade, ou qualquer outro tipo de dado, e também pode ser usada para definir relações entre indivíduos de classes distintas.

O primeiro uso de ontologias na área da informática foi em sistemas de inteligência artificial e se refere a artefatos de engenharia, constituídos por um vocabulário específico usado para descrever certa realidade (GUARINO, 1998). Após isto, seu uso foi proposto como base nos sistemas de informação baseados em ontologias.

O processo de construção de uma ontologia é bastante complexo visto que para sua definição é necessário realizar um estudo bastante detalhado de um determinado domínio para que não haja nenhum tipo de ambigüidade ou contestações quanto sua validade.

Resumidamente, pode-se afirmar que as ontologias atuam como modelos semânticos conceituais representando um conhecimento comum em um modelo bem definido, consistente, complexo, extensível, reutilizável e modular (LIBRELOTTO et al., 2008).

4.1 Utilização de ontologias

Pode-se dizer que ontologias são usadas para prover a comunicação entre pessoas ou sistemas que fazem parte de um domínio de conhecimento, mas nem sempre compartilham da

mesma conceituação a respeito dos componentes deste domínio. A falta de entendimento compartilhado pode trazer problemas na interoperabilidade e possibilidade de reuso e compartilhamento de conhecimento, que é de grande importância tendo-se em vista a grande variedade de métodos, paradigmas, linguagens e ferramentas existentes na área de computação (GUARINO, 1998).

A interoperabilidade entre recursos é possibilitada pelo uso de ontologias no desenvolvimento de modelagens que expressam o conhecimento possuído, formando uma camada de comunicação única aos usuários. Já o reuso e o compartilhamento tornam-se possíveis porque quando se usa ontologias para representar um determinado domínio, este torna-se padronizado, pois foi especificado em uma linguagem formal. Assim a leitura e interpretação da ontologia por outros domínios se torna mais fácil.

Segundo (DEPARTMENT; GRUNINGER, 1996), outro forte motivo para a utilização de ontologias se dá devido a necessidade de confiabilidade com relação aos conceitos do vocabulário ou linguagem que está sendo utilizado no domínio, pois a representação formal que se adquire com seu uso pode tornar possível a automação da checagem de consistência, o que implicará em ambientes mais confiáveis.

Além disso, é possível citar outras vantagens no uso de ontologias na área da computação (LOPES, 2006):

- conhecimento representado através de um vocabulário, que possui uma conceituação que o sustenta e evita que sejam feitas interpretações ambíguas;

- compartilhamento e reuso da ontologia que irá modelar de forma adequada um domínio por pessoas que desenvolvam aplicações desse domínio;

- por ser escrita em uma linguagem formal como, por exemplo, a linguagem OWL (MCGUINNESS; HARMELEN, 2004), terá a descrição exata do conhecimento em um domínio, evitando o "*gap* semântico" que existe na linguagem natural, onde a mesma palavra pode ter significados diferentes de acordo com o contexto em que está empregada. Por exemplo, a palavra memória pode se referir a um dispositivo de armazenamento de dados em um computador ou à memória humana (capacidade de natureza psicológica de adquirir, armazenar e evocar informações). Portanto, se existir uma conceituação comum e as pessoas envolvidas concordarem em uma ontologia sobre o domínio "computadores", possivelmente não haverá mal entendido;

- é possível fazer o mapeamento da ontologia sem alterar sua conceituação, ou seja, uma conceituação pode ser expressa em várias línguas;

- também é possível estender o uso de uma ontologia genérica para adaptá-la ao um domínio específico.

4.2 Classificação das ontologias

Na literatura, são encontrados diversos tipos de classificações para as ontologias na área da computação. Entre estas classificações podem-se citar quatro (STUDER; BENJAMINS; FENSEL, 1998).

As *ontologias de domínio* são usadas para representar o conhecimento de um domínio específico (como mecânica, medicina, biologia, entre outros). Expressam o vocabulário relativo a um domínio particular, descrevendo situações reais sobre o mesmo.

As *ontologias genéricas* são consideradas similares às ontologias de domínio, porém definem conceitos mais genéricos ou que podem ser usados em diferentes domínios, como: estado, espaço, tempo, processo, evento, importância, etc. Esses conceitos são independentes de um domínio particular. Os conceitos de uma ontologia de domínio são definidos como especializações de conceitos de ontologias genéricas.

Por sua vez, as *ontologias de aplicação* possuem os conceitos necessários para modelagem do conhecimento requerido por uma determinada aplicação. Estes conceitos correspondem freqüentemente aos papéis desempenhados por entidades do domínio enquanto executam uma atividade.

Por último, as *ontologias de representação* não estão ligadas a nenhum domínio específico e determinam entidades representacionais sem especificar o que será representado. As ontologias de representação fornecem primitivas que serão usadas pelas ontologias de domínio e genérica na hora de descrever seus domínios.

Outra classificação de ontologias definidas por (GUARINO, 1998), chamadas *ontologias de tarefas*, são usadas para descrever tarefas ou atividades genéricas através da especialização dos termos introduzidos pelas ontologias genéricas.

Entre as linguagens mais utilizadas para representação de ontologias destacam-se a OWL, RDF e RDFS, provenientes do XML, descritas a seguir.

4.3 XML

A linguagem XML (BRAY et al., 2008) é uma linguagem de marcação de dados e apresenta um formato para descrição de dados estruturados, facilitando assim, declarações mais precisas do conteúdo e trazendo resultados de busca mais relevantes através de múltiplas plataformas.

Enquanto a linguagem HTML possui *tags* apenas para definir formatação de parágrafos e caracteres, com XML é possível criar um número infinito de tags, pois possui um sistema para criar tags para dados estruturados.

Um elemento XML pode possuir tags como, nome do cliente, titulo de um livro, titulo de um CD, ou qualquer outro tipo de elemento de dado. As *tags* XML, quando encontradas, podem ser visualizadas através de um *browser* qualquer como, por exemplo, o Internet Explorer ou Mozilla Firefox, independente das aplicações onde os quais foram encontrados. Este dado encontrado, também pode ser transferido para outra aplicação para ser processada e visualizada no futuro.

O XML é um subtipo da Linguagem Padronizada de Marcação Genérica - SGML (*Standard Generalized Markup Language*) (CONNOLLY, 1995) capaz de descrever diversos tipos de dados, assegurando que os dados estruturados serão uniformes e independentes de aplicações e fornecedores.

A linguagem XML foi desenvolvida por um grupo de pesquisa do W3C e dentre seus objetivos pode-se citar (BRAY et al., 2008):

- Ser utilizado diretamente via *Web*;
- Suportar uma ampla variedade de aplicações;
- Ser compatível com SGML;
- Fácil integração com *softwares* que visam processar documentos XML;
- Documentos XML devem ser legíveis por humanos e razoavelmente claros;
- Seu projeto deve poder ser rapidamente preparado;
- Seu projeto também deve ser formal e conciso;
- Documentos XML devem ser de fácil criação.

Uma das principais características do XML é que após um dado ter sido recebido pelo cliente, ele pode ser editado e visualizado sem fazer chamadas ao servidor, reduzindo assim, a sobrecarga devido a menos requisições de banda para as comunicações entre cliente e servidor.

O XML é considerado de grande importância na Internet e em grandes intranets porque provê a capacidade de inter-operação dos computadores por ter um padrão flexível, aberto e independente de dispositivo. As aplicações podem ser construídas e atualizadas mais rapidamente e também permitem múltiplas formas de visualização dos dados estruturados (DUARTE; JUNIOR, 2000).

4.4 XSL

XSL é uma linguagem para estilos. Dada uma classe de documentos XML estruturados de forma arbitrária ou arquivos de dados, os projetistas utilizam XSL para expressar suas intenções sobre como conteúdo estruturado que deverá ser apresentado, isto é, como o conteúdo deve ser estilizado, definidos e paginadas em algum meio de apresentação, como uma janela em um navegador da Web ou um conjunto de páginas físicas em um catálogo, relatório, panfleto, ou livro, por exemplo (GLODBERG, 2008).

Páginas HTML utilizam *tags* as quais tem significados que podem ser entendidos diretamente. A linguagem XML permite a utilização de qualquer *tag* desejada e o significado delas não são automaticamente entendidos pelos *browsers*. Por exemplo, a *tag* <table> pode significar uma tabela HTML ou um móvel de mobília. Desta forma, devido a natureza do XML não existe uma forma padrão de exibição de documentos XML.

Para exibir documentos XML, é preciso um formalismo para descrever como este documento deve ser formatado. Além disso, também é necessário uma ferramenta para interpretar esta descrição e aplicar a melhor forma de visualização. O mecanismo mais utilizado para realizar este trabalho é o XSL (*eXtensible Stylesheet Language*) (GROSSO; WALSH, 2000).

Uma descrição XSL possui duas partes. A primeira refere-se a um método de transformação de documentos XML e, a segunda, refere-se a um método de formatação de documentos XML. A linguagem XSL pode ser usada para definir como um documento XML deve ser visualizado através da transformação deste documento para um formato que possa ser lido por um *browser* como por exemplo, transformá-lo para HTML.

Além disso, XSL também pode ser utilizada para adicionar novos elementos à um arquivo de saída, ou ainda para remover elementos. Com esta linguagem é possível re-arranjar e agrupar

elementos, além de testá-los e tomar decisões sobre quais elementos devem ser visualizados.

É devido a esta característica que XSL será utilizado projeto. Com ela será possível acessar um documento OWL, e selecionar determinados elementos, criando uma nova ontologia OWL. Estes elementos são relativos às entidades presentes no ambiente *homecare* em um momento específico, caracterizando um *contexto atual*.

4.5 Linguagens para especificação de ontologias

Uma ontologia pode ser descrita de diversas maneiras, até mesmo em formato de texto. Porém, existem linguagens específicas para a criação de ontologias, as quais possuem expressividades diferentes e sua utilização facilita o desenvolvimento das mesmas. Estas linguagens são descritas a seguir.

4.5.1 RDF

A linguagem RDF é uma recomendação da W3C para a representação de informações (LASSILA; SWICK, 1999). O RDF é uma fundação para o processamento de metadados que fornece uma interoperabilidade entre aplicações que troca informações compreensíveis por máquinas via *Web*. O objetivo principal dos arquivos RDF é a criação de modelos simples de dados, utilizando um vocabulário baseado em URIs, além de suportar o uso de XML.

O RDF é formado por três componentes básicos: um recurso, uma propriedade e uma indicação, que constituem uma "tripla". O recurso é tudo o que está sendo descrito através de expressões RDF. A propriedade é uma característica específica que será usada para a descrição de um recurso. Da combinação de um recurso, uma propriedade e um valor, surge uma indicação, que deve ser um dado atômico.

O modelo de metadados RDF baseia-se na idéia de fazer depoimentos sobre recursos através de expressões do tipo sujeito-predicado-objeto, o qual na terminologia RDF também se denomina "tripla". Esta tripla é capaz de formar uma rede de informação sobre os dados relacionados (MILLER et al., 2002).

O RDF é capaz de definir um mecanismo para descrever recursos que não fazem suposições sobre um determinado domínio de aplicação e, também não define (a princípio) a semântica deste domínio. A definição deste mecanismo deve ser feita de forma imparcial, ou seja, não pode ser específica para um determinado domínio, mas sim, deve ser aplicável para descrever informações sobre qualquer domínio de aplicação.

A linguagem RDF é orientada a objetos. Uma coleção de classes no RDF é chamado de esquema (*schema*), e são organizadas hierarquicamente, fornecendo uma extensibilidade através das subclasses definidas. Desta forma, para criar uma nova coleção de classes com apenas algumas modificações, não há necessidade de reconstruir todo o *schema*, mas sim, somente adicionar as modificações necessárias ao schema base.

4.5.2 RDF(S)

A linguagem RDF Schema - RDF(S), foi desenvolvida com o objetivo de representar metadados sobre recursos. O RDFS é um sistema de classes extensível e genérico que pode ser usado para descrever schemas de um determinado domínio de aplicação (BRICKLEY; GUHA; MCBRIDE, 2004).

Pode-se afirmar que, na semântica RDFS, um recurso organizado em classes pode ser instância de uma ou mais classes. As propriedades são utilizadas para indicar de qual classe um recurso faz parte. As classes no RDFS também podem ser divididas em subclasses, herdando as propriedades de sua superclasse. As propriedades têm por objetivo caracterizar uma instância ou relacioná-la com outra instância. Além disso, o RDFS permite criar restrições para as propriedades como, por exemplo, os elementos *rdfs:domain* e *rdfs:range* que são usados para dizer a quem esta propriedade se aplica.

A linguagem RDFS fornece informações a respeito da interpretação dos enunciados apresentados em um modelo de dados RDF, o que a difere do propósito das DTDs (BRAY et al., 2008) da linguagem XML, as quais fornecem restrições específicas a estrutura de um documento XML (LOPES, 2006).

4.5.3 OWL

A Linguagem OWL (MCGUINNESS; HARMELEN, 2004) tem sua origem no DAML+OIL (CONNOLLY et al., 2001). Mesmo sendo baseada em RDF e RDFS, e utilizando a sintaxe XML, a OWL tem a característica de ter mais facilidade para expressar significados e semântica que as demais. Além disto, ela foi projetada para ser utilizada em aplicações que precisam processar o conteúdo das informações ao invés de apenas visualizá-las.

A linguagem OWL, que é uma recomendação da W3C para representação de ontologias, tem base na lógica de descrição e pode ser dividida em três sub-linguagens: OWL Lite, OWL DL e OWL Full, cada uma correspondendo a uma lógica de descrição específica.

A OWL Lite possui apenas algumas características da OWL e é voltada para usuários que

necessitam de uma classificação hierárquica com simples restrições, já que possui uma complexidade formal menor que as outras duas sub-linguagens.

Por sua vez, a OWL DL é assim denominada, pois tem correspondência com a lógica descritiva e é utilizada por usuários que precisam de máxima expressividade, onde as conclusões têm garantia de serem processadas e as computações terminam em um tempo finito.

A OWL Full é usada por usuários que procuram máxima expressividade e total liberdade de sintaxe do RDF. Mesmo sendo mais expressiva que as demais sub-linguagens, a OWL Full possui a desvantagem de ter um maior custo de decisão pois dá mais liberdade para o programador desenvolver o código, o que pode torná-lo bastante complexo ao ser processado pelo computador. Ao contrário da OWL DL, a OWL Full permite unir OWL com RDFS e não requer disjunção de classes, propriedades, indivíduos e valores de dados, ou seja, uma ontologia criada em OWL Full pode possuir uma classe que é ao mesmo tempo classe e indivíduo.

Cada uma das sub-linguagens é uma extensão de sua predecessora. Isto significa que, por exemplo, uma ontologia válida em OWL Lite também será válida em OWL DL, mas nem toda ontologia válida em OWL DL será válida em OWL Lite.

4.5.4 OWL 2

Com um crescimento considerável de usuários que passaram a utilizar a linguagem OWL para o desenvolvimento de ontologias, alguns problemas e limitações passaram a ser evidenciados. Apesar de nenhum destes problemas serem considerados graves se analisados separadamente, quando analisados em conjunto mostravam a necessidade de uma revisão da linguagem (GRAU et al., 2008). Assim, no ano de 2007, foi criado um grupo de pesquisa para resolver estas questões e trabalhar em uma nova versão da linguagem, definida, em 2008, como OWL 2 (GROUP, 2009).

Além de resolver problemas de limitações encontrados na primeira versão da linguagem, a OWL 2 apresenta também algumas modificações em sua sintaxe com o objetivo de abranger novas funcionalidades e também um suporte estendido a novos tipos de valores de dados, o que proporciona uma maior expressividade no desenvolvimento de ontologias.

A OWL 2 conta com 3 perfis, ou sub-linguagens que oferecem vantagens importantes em cenários de aplicações específicas. Cada perfil é definido como uma restrição sintática da especificação estrutural da OWL 2, isto é, como um subconjunto de elementos estruturais que podem ser usados em uma ontologia. O perfil OWL 2 EL captura o poder expressivo usado

em ontologias de grande escala. Voltado para ontologias grandes, porém simples, e com bom desempenho no tempo de inferência sobre ela. Por sua vez, o OWL 2 QL captura a expressividade usada normalmente em ontologias simples como thesaurus e a maior parte do poder expressivo de esquemas ER/ UML. Este perfil é mais adequado para aqueles que necessitam uma fácil comunicação com bancos de dados relacionais e onde o raciocínio sobre grandes conjuntos de dados é o mais importante. Por último, o OWL 2 RL foi projetado para acomodar aplicações OWL 2 que necessitam usar toda a expressividade da linguagem com mais eficiência e em aplicações RDF(S) que precisam adicionar a expressividade da OWL 2.

Os três perfis possuem certas restrições sintáticas sobre determinados construtores e axiomas. As versões QL e RL são adequados para aplicações onde ontologias relativamente leves são usadas com grandes conjuntos de dados. A escolha entre eles vai depender dos tipos de dados a serem processados. Quando for necessário acessar dados diretamente através de consultas SQL, o perfil OWL 2 QL é o mais indicado. Por sua vez, o OWL 2 RL é voltado para quem precisa operar sobre dados na forma de triplas RDF.

4.6 Linguagens para realização de inferência sobre ontologias

Para aumentar o poder de expressão e realizar consultas sobre a ontologia, faz-se necessário o uso de de linguagens que possibilitem realizar inferências sobre a mesma. Dessa forma, este capítulo tem por objetivo discutir as linguagens mais difundidas para manipulação de ontologias descritas em OWL e que serão utilizadas neste trabalho.

4.6.1 SWRL

Existem situações onde uma ontologia descrita em OWL não consegue expressar o conhecimento existente em um domínio apenas com seus construtores. Um exemplo disso aparece na modelagem de uma ontologia para um domínio de uma família, ou seja, onde deseja-se modelar as entidades e relações de um conjunto de pessoas que pertencem a mesma família. Neste caso não é possível definir, de forma genérica, que uma pessoa possui um tio. Apenas é possível descrever de forma limitada onde definimos que um indivíduo específico (João) possui um tio (José), ou seja, é possível apenas criar um relacionamento entre indivíduos específicos. Porém, quando há a necessidade de criar um conhecimento genérico sobre a relação “tio”, que seja válida para os indivíduos da classe Pessoa, a linguagem OWL se torna insuficiente. Em casos como este é necessário utilizar as regras da linguagem SWRL - *Semantic Web Rule Language*

(HORROCKS et al., 2004).

A linguagem SWRL possui alguns tipos de regras os quais podem ser utilizados juntamente com o conhecimento expresso nas ontologias, aumentando seu poder de expressividade. Estas possibilitam a criação de regras para raciocínio sobre indivíduo, permitindo inferir novos conhecimentos a partir de informações já conhecidas. Com as regras SWRL é possível definir que um indivíduo qualquer (a) tem uma relação do tipo pai com outro indivíduo (b) e este, por sua vez, tem uma relação do tipo irmão com um terceiro indivíduo (c), "a" terá "c" como tio. Assim, para uma representação mais poderosa utiliza-se a linguagem OWL juntamente com as regras SWRL.

Ao executar uma regra, seu resultado irá afetar diretamente a estrutura da ontologia, ou seja, toda vez que as restrições presentes em uma regra forem aceitas, um novo conhecimento sobre o domínio será aplicado à ontologia.

4.6.2 SQWRL

A linguagem SQWRL - *Semantic Query-Enhanced Web Rule Language* (O'CONNOR; DAS, 2008) permite a criação de consultas à ontologias de forma análoga ao SQL. A diferença entre as duas linguagens é que o resultado da execução de uma regra SWRL irá refletir diretamente na estrutura de classes, propriedades e indivíduos da ontologia, já o resultado de uma consulta SQWRL apenas retorna informações da ontologia que satisfazem determinadas restrições e podem ser manipulados posteriormente utilizando uma linguagem de programação qualquer, como JAVA.

O SQWRL utiliza uma biblioteca com os métodos do SWRL, podendo utilizar-se destas regras para fazer buscas em OWL. O código a seguir mostra a sintaxe utilizada pela linguagem SQWRL para consultas.

```
Classe(?c) ^ propriedade(?c, ?v) -> sqwrl:select
```

As informações precedidas por "?" são passados ao motor de buscas. Note que o elemento propriedade possui dois valores sendo passados ao motor de buscas, eles correspondem ao domain e ao range da propriedade em questão. Ao declarar qual será a busca, coloca-se uma seta "->" e então se declara o comando sqwrl:select que fará o retorno dos elementos que estão entre parênteses. Pelo fato de poder trabalhar usando as regras SWRL, o SQWRL torna-se uma poderosa linguagem para buscas em OWL, podendo aproveitar ao máximo a expressividade semântica representada em OWL.

4.7 Considerações do capítulo

Este Capítulo apresentou discussões relacionadas a utilização de ontologias como forma de representação do conhecimento de um domínio de aplicação, bem como linguagens para realização de inferências e consultas sobre as mesmas. Pode-se concluir que a linguagem OWL é a mais adequada para estes fins devido suas características e poder de expressividade, além de ser uma recomendação do consórcio W3C. Quanto as linguagens de manipulação de ontologias, as mais indicadas são SWRL e SQWRL, uma vez que foram desenvolvidas para serem aplicadas especificamente em ontologias OWL podendo trabalhar com toda expressividade existente na OWL.

No capítulo a seguir é apresentada a ontologia desenvolvida para o projeto, a qual representa o conhecimento existente em um ambiente pervasivo de *homecare*.

5 UMA ONTOLOGIA PARA A REPRESENTAÇÃO DO CONHECIMENTO DO DOMÍNIO DE *HOMECARE*

Este capítulo apresenta a ontologia desenvolvida para ambientes *homecare*, a qual tem por objetivo mapear o conhecimento existente neste tipo de domínio. A ontologia foi criada de forma a atender as relações médico-paciente envolvidos em um tratamento doméstico, bem como as outras entidades envolvidas neste processo e suas funcionalidades.

A ontologia em questão foi desenvolvida através da linguagem OWL DL devido ao seu poder de expressividade e garantia de que suas computações terminam em tempo finito. Ela representa o conhecimento existente em um sistema *homecare* pervasivo, cujo objetivo é agilizar o processo de tratamento de pacientes. Dentre as classificações citadas na seção 4.2, é possível afirmar que a ontologia criada no âmbito desta pesquisa é classificada como *ontologia de domínio*, por representar um domínio específico, onde expressa um vocabulário e descreve situações de um ambiente *homecare*.

A ontologia foi projetada a partir de reuniões com uma junta médica de um hospital universitário onde o objetivo era esclarecer como é realizado o trabalho de profissionais da saúde durante o fluxo de tratamento de pacientes. O fluxo de tratamento de pacientes em hospitais ou em casa podem ser considerados de certa forma semelhantes, uma vez que, independente do local, o paciente deve ser monitorado constantemente, assegurando que caso aconteça uma piora em seu estado de saúde, ele seja atendido prontamente. O trabalho de médicos em ambos os locais, envolve questões como revisão de problemas do paciente, prescrição de exames, medicamentos e tratamentos, e análise de exames. Desta forma, as reuniões com a junta médica do hospital foi de grande valia para o desenvolvimento do projeto e, o processo de construção da ontologia descrito neste capítulo utiliza informações retiradas destas reuniões.

Além desta junta médica, utilizou-se a pesquisa de (LAERUM; FAXVAAG, 2004), onde se levantaram as tarefas médicas mais comuns no tratamento de pacientes. Esta pesquisa foi fundamental para a criação das relações entre as principais entidades desta ontologia.

Nesta ontologia estão descritas as diversas classes de um ambiente *homecare*, bem como suas propriedades, as quais tem o objetivo de definir características das classes, e assim criar um elo de ligação entre as mesmas. Estas classes e propriedades representam as entidades e relações existentes neste tipo de ambiente.

Na ontologia, as classes representam as entidades que fazem parte de ambientes de *home-*

care genéricos (por exemplo, *Médico*); as propriedades de dados ou atributos que caracterizam estas classes (por exemplo, *código, nome, especialidade*); e também as propriedades de objeto, ou relacionamentos, que são usados para ligar duas classes distintas (por exemplo, *Médico analisa Histórico*).

Esta ontologia foi desenvolvida com a linguagem OWL através do software Protégé 3.4.4 (KNUBLAUCH et al., 2004), que conta com uma grande quantidade de usuários, além de uma excelente documentação.

5.1 Classes

A partir de informações adquiridas nas reuniões com a junta médica e do trabalho de (LAERUM; FAXVAAG, 2004), e considerando que o objetivo principal do trabalho é uma representação de um ambiente *homecare* pervasivo, verificou-se a necessidade da criação de um conjunto de classes. As classes e suas descrições são apresentadas na tabela 5.1.

As entidades foram definidas pois podem, de alguma forma, interferir no fluxo de tratamento do pacientes. Além destas classes ainda foi definida uma classe abstrata (ou seja, nenhum indivíduo faz parte dela), chamada *Pessoa* que é superclasse de *Profissional, Paciente, Acompanhante, Visitante e Familiar*. A classe *Profissional* também é uma classe abstrata, a qual contém as classes *Médico, Enfermeiro e Técnico em Enfermagem*. A terceira classe abstrata é *Exame* formada por *Hemograma, Raio-X, Exame de Urina e Outro Exame*. Por fim, a quarta classe abstrata é *Casa* formada por *Cômodo* e *Móvel*. As classes que compõem uma superclasse são chamadas de subclasses, criando assim um sistema hierárquico na ontologia. Com isto, estas subclasses herdam automaticamente as propriedades (atributos) que foram definidas para sua superclasse. A figura 5.1 apresenta um grafo com a hierarquia de classes da ontologia.

As classes de uma ontologia podem possuir dois tipos de propriedades: de dado e de objeto. As propriedades de dado (também chamadas de atributos) são classificadas como as características que os indivíduos de cada classe possuem. Por sua vez, as propriedades de objeto (também chamadas de relacionamentos ou relações) tem por objetivo criar uma relação entre duas classes distintas. A seguir estes dois tipos de propriedades serão descritos em detalhes.

5.2 Propriedades de dado (atributos de classes)

As propriedades de dado de uma classe são características que ela pode possuir e que serão usadas para diferenciá-las umas das outras. A função das propriedades é identificar as classes e

Tabela 5.1: Descrição das classes da ontologia

Classe	**Descrição**
Médico	Responsável por realizar o tratamento clínico do paciente.
Enfermeiro	Responsável por auxiliar o médico e assistir o paciente durante o tratamento.
Técnico em enfermagem	Deve auxiliar os enfermeiros e assistir o paciente no que for necessário.
Acompanhante	Pessoa que fica de plantão durante um determinado momento. Pode ser visitante, familiar, ou profissionais.
Paciente	É quem recebe cuidados médicos.
Visitante	Pessoa que visita paciente. Pode ser amigos ou familiares distantes e pode se tornar acompanhante em determinados momentos.
Familiar	Pessoa que possui laços sanguíneos com o paciente. Pode se tornar acompanhante em determinados momentos.
Sintoma	Alteração no estado de saúde do paciente que são usados para diagnosticar doença.
Tratamento	Procedimento médico para recuperação do paciente. Pode variar de repouso até cirurgia.
Medicamento	Remédios administrados ao paciente durante o tratamento
Sinais vitais	São constituidos de frequência cardíaca, pressão arterial e temperatura. São monitorados constantemente.
Hemograma	Tipo de exame que paciente realiza.
Raio-X	Tipo de exame que paciente realiza. É uma imagem.
Urina	Tipo de exame que paciente realiza.
Outro Exame	Nesta classe são inseridos demais tipos de exame que o médico possa solicitar.
Tarefa	São atividades realizadas pelos usuários do sistema. Não necessariamente envolvem dispositivos computacionais.
Animal de estimação	Classe necessária, pois pode estar diretamente ligada ao estado de saúde do paciente. Por exemplo em casos de doenças alérgicas. Uma das causas da doença pode ser a pelagem do animal.
Dispositivo computacional	Corresponde aos dispositivos usados no ambiente. Por exemplo, tablets, smartphone, PDAs e TVs digitais.
Cômodo	Representa os ambientes de uma casa. Por exemplo, sala, quarto ou cozinha.
Móvel	Nesta classe são representados os tipos de móveis do ambiente em questão

também definir ações que possam ser exercidas pelas mesmas (LIBRELOTTO et al., 2011b). A classe *Pessoa*, por exemplo, possui as propriedades *código*, *nome* e *possui_ sintoma*.

As subclasses de *Pessoa* herdam suas propriedades. A propriedade *nome* representa o nome de cada um dos indivíduos que fazem parte de alguma das subclasses de *Pessoa* como as classes *Médico* e *Paciente*, por exemplo. A figura 5.2 as propriedades de dados de cada uma das classes.

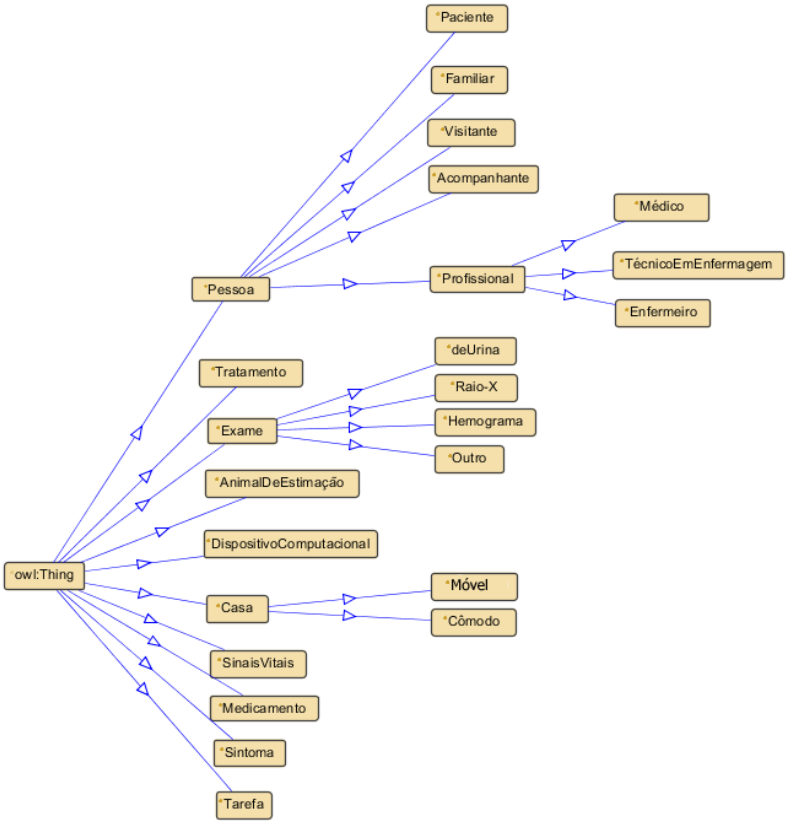

Figura 5.1: Hierarquia de classes da ontologia para *homecare*

- *Pessoa* - identificador e nome
  - *Profissional* - data de admissão e turno
    - *Médico* - CRM, especialidade
    - *Enfermeiro* - COREN
    - *Técnico em enfermagem* - COREN
  - *Acompanhante* - grau de afinidade
  - *Paciente* - doença e idade (diferenciar crianças e adultos)
  - *Visitante* - grau de afinidade e idade (diferenciar crianças e adultos)
  - *Familiar* - grau de parentesco e idade (diferenciar crianças e adultos)
  - *Sintoma* - identificador, tipo e frequência
  - *Tratamento* - identificador, data de início, data de fim e tipo
  - *Medicamento* - identificador, nome, posologia, fabricante e hora
  - *Sinais Vitais* - temperatura, batimento e pressão
  - *Exame* - identificador e data
    - *Hemograma* - hemácias, leucócitos e plaquetas
    - *Raio-X* - imagem
    - *Urina* - densidade, pH, glicose, proteínas, leucócitos e hemácias
    - *Outro* - nome, descrição, resultado e observação
  - *Tarefa* - identificador, nome e hora
  - *Animal de Estimação* - identificador, raça, pelagem e agressividade
  - *Dispositivo Computacional* - identificador, modelo, tamanho e alerta
  - *Casa* - identificador
    - *Cômodo* - quarto, sala, banheiro, cozinha e outro
    - *Móvel* - mesa, cadeira, sofá, cama, estante e outro.

Figura 5.2: Classes e seus atributos

5.3 Propriedades de objeto (relações entre classes)

A ontologia foi desenvolvida para ser aplicada ao sistema de *homecare* pervasivo, o qual terá o poder de sugerir possíveis próximos passos aos profissionais da saúde com relação ao tratamento de pacientes. Sendo assim, os relacionamentos deveriam ser criados para proporcionar informações relevantes para tornar possível estas sugestões de ações a serem executadas pelos profissionais.

Uma propriedade de objeto de uma ontologia representa uma relação que liga duas classes distintas. Um exemplo de relação é *Exame visualizado_por Médico*, que associa indivíduos da classe *Exame* com indivíduos da classe *Médico* através da propriedade *visualizado_por*. Ao perceber que este relacionamento foi executado, o sistema gerenciador de contexto disponibiliza em uma tela de computador algumas ações que o médico possa realizar após visualizar um exame como, por exemplo, prescrever medicamentos ou solicitar outros exames. A tabela 5.2 mostra os relacionamentos criados para a ontologia.

Analisando a tabela percebe-se que alguns relacionamentos possuem também uma relação inversa. Com este tipo de relacionamento é possível aumentar ainda mais a eficiência da ontologia, já que sua abrangência será maior.

A figura 5.3 mostra um grafo que representa como estas classes estão relacionadas na ontologia proposta.

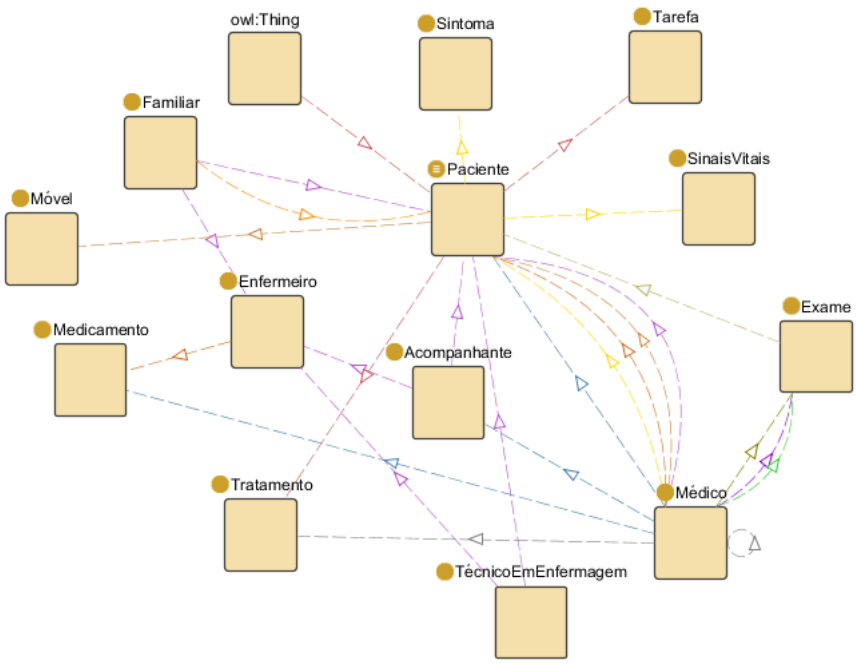

Figura 5.3: Grafo da ontologia de *homecare*

Tabela 5.2: Relacionamentos entre classes na ontologia

Classe	Relacionamento	Classe	Relação Inversa
Medico	revisa_problemas	Paciente	
Médico	analisa_histórico	Paciente	X
Médico	prescreve	Medicamento	X
Médico	requisita	Tratamento	X
Médico	solicita	Exame	X
Médico	compara	Exame	X
Médico	visualiza	Exame	X
Médico	responsável_por	Paciente	X
Médico	recomenda	Médico	X
Médico	prescreve_atestado	Paciente	
Médico	atualiza_histórico	Paciente	
Médico	explica_diagnóstico	Paciente	
Médico	explica_diagnóstico	Acompanhante	
Paciente	possui	Sintomas	
Paciente	possui	SinaisVitais	
Paciente	toma	Medicamento	X
Paciente	realiza	Tarefa	X
Paciente	dono_de	AnimalDeEstimação	
Paciente	ocupa	Móvel	
Medicamento	administrado_a	Paciente	
Medicamento	prescrito_para	Paciente	
Exame	pertence_a	Paciente	
Enfermeiro	assiste	Paciente	X
Enfermeiro	Auxilia	Médico	X
Enfermeiro	administra	Medicamento	X
Acompanhante	auxilia	Paciente	X
Familiar	auxilia	Paciente	X
TécnicoEmEnfermagem	assiste	Paciente	X
TécnicoemEnfermagem	auxilia	Enfermeiro	X

As relações modeladas para ontologia tem por objetivo representar as ligações que as entidades presentes em um ambiente *homecare* podem possuir. O quadro 5.3 apresenta a descrição das relações envolvendo indivíduos da classe *Médico*.

O quadro 5.4 apresenta a descrição dos relacionamentos envolvendo as demais entidades do ambiente *homecare*.

5.4 Instâncias

As instâncias de uma ontologia representam os indivíduos que compõem o domínio em questão. Cada pessoa que faz parte do domínio de *homecare* é considerado uma instância. É possível fazer uma analogia com um banco de dados relacional, onde cada registro de um campo

Tabela 5.3: Descrição dos relacionamentos de médicos

Relacionamento	**Descrição**
Médico revisa problemas do Paciente	Analisar e coletar informações do paciente com o objetivo de criar novos diagnósticos e solicitar novas investigações
Médico analisa histórico do Paciente	Analisar histórico à procura de modificações no estado de saúde do paciente como alteração de sinais vitais e exames
Médico prescreve Medicamento	A partir de um diagnóstico, o médico prescreve um ou mais medicamentos para o tratamento do paciente
Médico requisita Tratamento	A partir de um diagnóstico, o médico requisita um tratamento do paciente (fisioterapia, repouso, cirurgia, etc)
Médico solicita Exame	Solicitação de exames para um investigação sobre o estado de saúde do paciente
Médico visualiza Exame	Médico analisa exame realizados que ele havia solicitado
Médico compara Exame	Comparar exame atual com outros já realizados para verificar alteração do estado de saúde.
Médico responsável por Paciente	Cada paciente é relacionado com um ou mais médicos que serão responsáveis pelo seu tratamento
Médico recomenda Médico	Médico responsável recomenda tratamento com especialista, baseando-se em determinado diagnóstico
Médico prescreve atestado Paciente	Prescrição de atestado médico detalhando o estado de saúde, a partir de um diagnóstico
Médico atualiza_histórico Paciente	O histórico médico do paciente é atualizado com informações de exames e outras análises.
Médico explica_diagnóstico Paciente	Médico dá informações sobre estado de saúde do paciente quando este solicita
Médico explica_diagnóstico Acompanhante	Médico dá informações sobre estado de saúde do paciente para acompanhante

de uma tabela no banco corresponde a uma instância em uma ontologia.

Em uma ontologia, as instâncias são agrupadas ou classificadas em classes. Isso significa que as instâncias de mesmo tipo devem estar reunidas na mesma classe. Em um ambiente *homecare*, por exemplo, se as pessoas Pedro, João e Maria apresentarem alguma enfermidade, elas são classificadas como pacientes e, consequentemente, serão instâncias da classe *Paciente*.

Como a ontologia desenvolvida para este projeto irá utilizar informações baseadas em um contexto específico do ambiente *homecare*, nenhuma instância foi inicialmente definida e alocada às classes. Estas classes serão preenchidas com informações coletadas por sensores que monitoram o ambiente em questão. Estes sensores são descritos em detalhes no capítulo 6.

Tabela 5.4: Descrição das relações que não envolvem médicos

Relacionamento	**Descrição**
Paciente possui Sintomas	Sintomas encontrados no paciente serão utilizados para médico encontrar diagnóstico
Paciente possui Sinais Vitais	Paciente possui um monitoramento constante de seus sinais vitais
Paciente ocupa Cama	Cada cama é relacionada a um paciente, uma vez que esta pode influenciar no tratamento
Paciente realiza Tarefa	Em casos menos graves, paciente pode realizar tarefas diárias normalmente, apenas precisam de acompanhamento médico
Paciente dono de Animal de Estimação	Informações de animais de estimação do paciente podem influenciar no tratamento, como em casos de alergia a pêlos
Paciente ocupa Móvel	O móvel pode influenciar no tratamento. Em casos onde o paciente tem problemas de coluna, a cama que ocupa é relevante.
Medicamento prescrito para Paciente	Medicamento é prescrito para paciente a partir de informações de exames e diagnóstico
Medicamento administrado a Paciente	Medicamento é administrado de acordo com uma posologia prescrita pelo médico
Exame pertence a Paciente	Um exame médico deve estar relacionado a um paciente para que seu médico responsável possa analisá-lo
Enfermeiro assiste Paciente	Em casos mais graves, enfermeiro auxilia paciente em suas tarefas, além de administração de remédios e reportar alterações
Enfermeiro Auxilia Médico	Enfermeiro receber instruções de como proceder no tratamento do paciente, enquanto médico estiver auxente
Enfermeiro administra Medicamento	Enfermeiro administra remédios ao paciente de acordo com a prescrição deixada pelo médico
Acompanhante auxilia Paciente	Acompanhante ajuda paciente em situações onde enfermeiro ou técnico em enfermagem não estão disponíveis
Familiar auxilia Paciente	Familiar ajuda paciente em situações onde enfermeiro ou técnico em enfermagem não estão disponíveis
Técnico em enfermagem assiste Paciente	Técnico ajuda pacientes na realização de suas tarefas em casos mais graves como incapacidade de locomoção
Técnico em enfermagem auxilia Enfermeiro	Técnico realiza tarefas mais rotineiras, auxiliando o trabalho do enfermeiro

5.5 Consultas

As consultas desenvolvidas para a ontologia do sistema foram criadas através da linguagem SQWRL e serão executadas cada vez que as condições definidas nas mesmas forem aceitas pelo motor de inferências. Uma consulta SQWRL é executada, por exemplo, quando o médico se aproxima de uma tela que estaria disposta no quarto do paciente. O sistema identifica o médico e mostra na tela uma sugestão perguntando se ele gostaria de visualizar exames que havia solicitado para aquele paciente. Para isto, o sistema executa a seguinte consulta:

```
Paciente(?p) ^ Medico(?m) ^ Exame(?er) ^
pertence_a(?er, ?p) ^ responsavel_por(?m, ?p) ^
visualizado_por(?er, ?v) ^ swrlb:notEqual(?v, true) ->
sqwrl:select(?er)
```

Ao executar esta consulta o sistema verifica se existem exames do paciente em questão, se o médico é o responsável por aquele paciente e se o exame ainda não foi visualizado. Caso estas premissas sejam verdadeiras, o exame é mostrado na tela para o médico.

Outra consulta desenvolvida para ser aplicada à ontologia retorna os medicamentos que devem ser administrados pelo enfermeiro de plantão para o paciente. Para ilustrar esta situação, deve-se esclarecer que no ambiente *homecare* existe um *dispenser*, seja no quarto do paciente ou em outro cômodo, no qual são armazenados os remédios prescritos pela equipe médica. Quando o enfermeiro se aproxima do *dispenser*, ele é identificado e recebe a lista de medicamentos que o paciente deve tomar em um dispositivo computacional, que pode ser um *tablet, Smartphone* ou uma tela disposta próxima ao *dispenser*. O código da consulta é o seguinte:

```
Enfermeiro(?e) ^ Paciente(?p) ^ Medicamento(?m)
^ assistido_por(?p, ?e) ^ e_para(?m, ?p)
-> sqwrl:select(?p, ?m)
```

Nesta consulta o sistema verificaria quais pacientes são supervisionados por aquele enfermeiro e quais medicamentos o paciente deve tomar. O retorno desta consulta será uma lista dos medicamentos de cada paciente (caso haja mais de um), juntamente com a posologia dos mesmos.

Caso este *dispenser* seja localizado em um cômodo diferente daquele em que o paciente se encontra, outra consulta pode ser executada para que o enfermeiro saiba qual a posologia do remédio deve ser aplicada. Ao entrar no quarto do paciente, os medicamentos trazidos pelo profissional são identificados e ele recebe em um dispositivo computacional a posologia desses remédios para que sejam administrados corretamente ao paciente. O código para esta consulta é o seguinte:

```
Paciente(?p) ^ Medicamento(?m) ^ posologia(?m, ?pos)
^ pertence_a (?m, ?p) -> sqwrl:select (?p, ?m, ?pos)
```

Neste caso, ao executar esta consulta o sistema identifica o paciente e o remédio. Em seguida verifica-se qual remédio pertence ao paciente e também a posologia de cada remédio. Caso estas condições sejam atendidas, o sistema mostra na tela o nome de cada remédio e sua posologia.

É possível perceber nas consultas acima que além dos construtores da SQWRL também é utilizado um construtor da linguagem SWRL. Isto é possível, pois como a SQWRL foi desenvolvida baseando-se nas regras SWRL, os operadores definidos para SWRL também podem ser utilizados na especificação de consultas (FREITAS et al., 2011).

5.6 Inferências sobre a ontologia

Uma grande vantagem da utilização de ontologias na representação do conhecimento de um domínio se deve a possibilidade de utilizar regras de inferência para a dedução de novos fatos a partir de informações já conhecidas e descritas. Estas regras serão executadas a partir de um motor e estes novos fatos serão armazenados juntamente com as outras informações na ontologia.

Para este projeto foram criadas regras de inferência através da linguagem SWRL que serão executadas sempre que suas premissas forem aceitas pelo motor de inferências, assim como nas consultas SQWRL. A seguir serão descritas algumas das regras desenvolvidas para serem utilizadas com a ontologia.

O primeiro caso no qual pode ser utilizada uma regra de inferência refere-se a um contexto onde estão mapeados na ontologia um paciente e um familiar. Sabe-se que muitas vezes quando uma pessoa está doente, familiares bastante distantes aparecem para prestar solidariedade, mas que não tem um grau de parentesco ou intimidade alto com o paciente. Nestes casos, o paciente pode sentir-se constrangido de falar detalhadamente sobre seu problema e, consequentemente, não gostaria, por exemplo, que seu médico analisasse exames sobre seu estado de saúde na frente deste familiar. Por ter uma relação de sangue com o paciente, na ontologia esta pessoa está mapeada como sendo um indivíduo da classe *Familiar*.

Como nem os familiares são próximos ao paciente, foi desenvolvida uma regra que diz que, se um familiar tem um grau de parentesco caracterizado como sendo “Baixo” então ele também deve ser considerado um visitante e será mapeado na ontologia como tal. A regra que especifica isto é a seguinte:

```
Familiar(?f) ^ Paciente(?p) ^ parente_de (?f, ?p) ^
grau_de_parentesco(?f, ?grau) ^ swrlb:equal(?grau, "Baixo")
```

```
-> Visitante(?f)
```

Quando esta regra é executada, o motor consegue inferir que um indivíduo que faz parte da classe *Familiar* também deve fazer parte da classe *Visitante*, mesmo que esta informação não tenha sido descrita anteriormente. Isto é possível devido às informações contidas na propriedade de dado "grau_de_parentesco" e na propriedade de objeto "parente_de" que liga os indivíduos das duas classes.

Com isso, se o médico chega enquanto este familiar está presente, o sistema não executará nenhuma tarefa que possa comprometer o desejo do paciente de não informar detalhes de seu estado de saúde para seu parente. O sistema irá procurar outra forma de disponibilizar a informação ao médico como, por exemplo, através de um *smartphone* ou *tablet* pessoal. Esta ação pode ser gerenciada através de uma consulta SQWRL que utiliza operadores SWRL.

Outro caso em que é possível utilizar uma regra de inferência trata-se de situações em que o paciente esta recebendo uma visita de um amigo próximo, por exemplo. Por não ter nenhum vínculo sanguíneo com o paciente, esta pessoa foi mapeada na ontologia como sendo apenas um indivíduo da classe *Visitante*. Como foi descrito anteriormente, um dos atributos da classe *Visitante* refere-se ao grau de afinidade que este tem com relação ao paciente. Neste caso, se o visitante possuir um alto grau de afinidade com o paciente, ele também deve ser mapeado para a classe *Acompanhante*. Isto é possível através da seguinte regra:

```
Paciente(?p) ^ Visitante(?v) ^ grau_de_afinidade(?v, ?grau)
^ idade(?v, ?id) ^ swrlb:equal(?grau, "alto")
^ swrl:b:greaterThan(?id, 17) -> Acompanhante(?v)
```

De maneira geral, esta regra diz que apenas os visitantes que tenham um alto grau de afinidade também são considerados acompanhantes. Mesmo sem ter sido previamente definido como indivíduo da classe *Acompanhante*, o motor de inferências vincula o amigo do paciente a esta classe devido ao atributo "grau_de_afinidade" ser alto. Isto é necessário pois pode haver situações em que o enfermeiro ou o técnico em enfermagem se ausentam do ambiente e então o acompanhante passa a ser a pessoa que está visitando o paciente, ou seja, será a responsável por auxiliar o paciente caso ele precise. Além disso, por ter um alto grau de afinidade, é possível que o paciente permita que o acompanhante saiba detalhes de seu estado de saúde, como foi descrito no exemplo onde o médico chega na casa e precisa analisar os exames ou histórico do

paciente. Sendo acompanhante, o visitante terá permissão de visualizar os exames que o médico está analisando, uma vez que estes estarão sendo mostrados em uma tela de computador.

Uma terceira regra desenvolvida que pode ser usada para ilustrar o funcionamento do motor de inferências, refere-se a uma situação em que o paciente deve ter sua temperatura controlada, uma vez que este fator é fundamental para o sucesso do seu tratamento. Suponha-se que caso a temperatura do paciente aumente demais (por exemplo: 39°C) o médico deve ser notificado pois o paciente pode estar passando por uma crise. Neste caso, é necessário estabelecer uma regra para criar uma instância do atributo *alerta* da classe *DispositivoComputacional* através do código a seguir:

```
Paciente(?p) ^ SinaisVitais(?sv)
^ DispositivoComputacional(?dc)
^ alerta(?dc, ?a) ^ possui(?p, ?sv) ^ temperatura(?sv, ?t)
^ swrlb:greaterThan(?t, 39) -> alerta(?t)
```

Ao executar esta regra, uma informação de texto é enviada para o atributo *alerta* da classe *DispositivoComputacional*, porém até este momento o médico não foi notificado. Para que isto aconteça, é necessário a execução de uma consulta, através do seguinte código:

```
DispositivoComputacional(?dc) ^ alerta(?dc, ?a)
^ swrlb:notEqual(?a, true)-> sqwrl:select(?a)
```

Neste ponto, o motor de inferências verifica que existe um alerta ainda não visualizado e mostra na tela do dispositivo computacional uma sugestão perguntando se o médico deseja analisar naquele momento.

5.7 Restrições de propriedades

Foram criadas restrições na ontologia a fim de torná-la mais coerente com relação ao que realmente é permitido acontecer dentro de um ambiente *homecare*. As restrições, como o próprio nome já diz, têm a função de restringir os indivíduos que pertencem a determinadas classes. Assim é possível definir classes anônimas que irão englobar os indivíduos que satisfazem determinada restrição. Em OWL existem três tipos de restrições de propriedades (HORRIDGE et al., 2004): restrições de quantificador, restrições de cardinalidade e restrições do tipo temValor.

As restrições de quantificador são usadas quando indivíduos de uma classe devem possuir pelo menos um ou mais valores de um conjunto. Um exemplo pode ser visto no relacionamento

dos indivíduos da classe *Paciente* com os da classe *Sintoma* através da propriedade *possui*. Esta restrição diz que os indivíduos da classe *Paciente* devem possuir pelo menos um sintoma, caso contrário não seriam considerados paciente.

Por sua vez, as restrições de cardinalidade são usadas para restringir a quantidade de valores que uma determinada propriedade pode assumir. Este tipo de restrição é utilizado em propriedade de dados para garantir que determinado atributo exista. Um exemplo disso acontece quando cria-se uma restrição do atributo *doença* da classe *Paciente* especificando que o paciente deve ter pelo menos uma ocorrência desse atributo, caso contrário não será considerado paciente. A figura 5.4 mostra uma imagem do *software* Protégé com as restrições criadas para os indivíduos da classe *Paciente*.

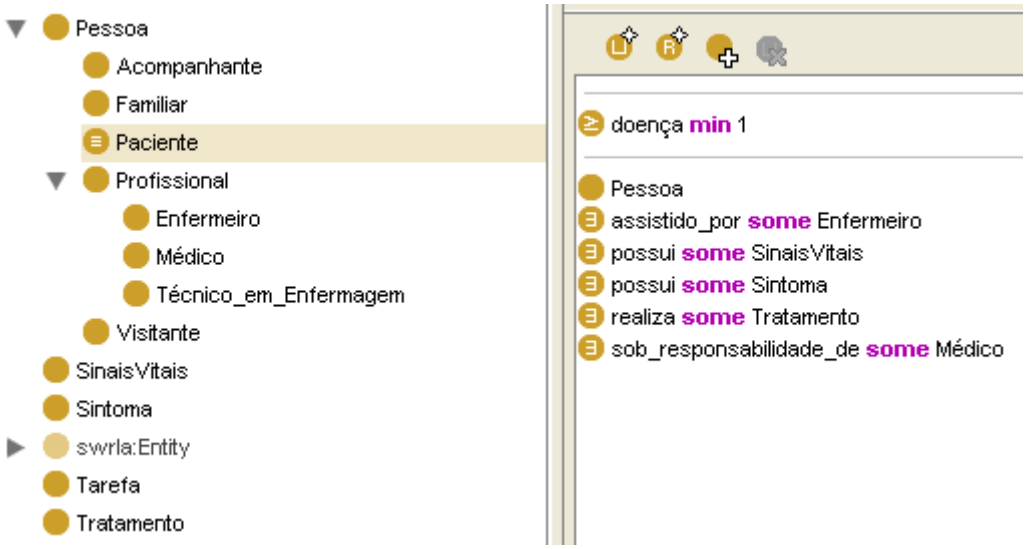

Figura 5.4: Restrições de propriedade da classe *Paciente*

Pela figura pode-se perceber que um paciente deve possuir pelo menos uma relação com indivíduos das classes *Enfermeiro* (propriedade *assistido_por*), *Sintoma* (propriedade *possui*), *SinaisVitais* (propriedade *possui*), *Tratamento* (propriedade *realiza*) e *Médico* (propriedade *sob_responsabilidade_de*). Além disso, ele precisa ter pelo menos uma doença. Caso estas restrições não sejam satisfeitas, o indivíduo não pode ser considerado paciente, o que tornará a ontologia inconsistente.

Por último as restrições do tipo temValor, que são usadas em casos em que a restrição deve ser aplicada ao *range* da propriedade (extensão de aplicação). Em outras palavras, essa restrição descreve o conjunto de indivíduos que se relacionam com um outro indivíduo específico através da mesma propriedade. Na ontologia que representa o ambiente *homecare* pode ser vista na relação entre médico e paciente. Em um caso hipotético, apenas para ilustrar a situação, foi

criado um indivíduo para a classe *Paciente* chamado “João” e dois indivíduos para a classe *Médico* chamados “Paulo” e “Maria”. A descrição dos médicos que são responsáveis por ele pode ser definida através dessa restrição. A figura 5.5 ilustra esta situação.

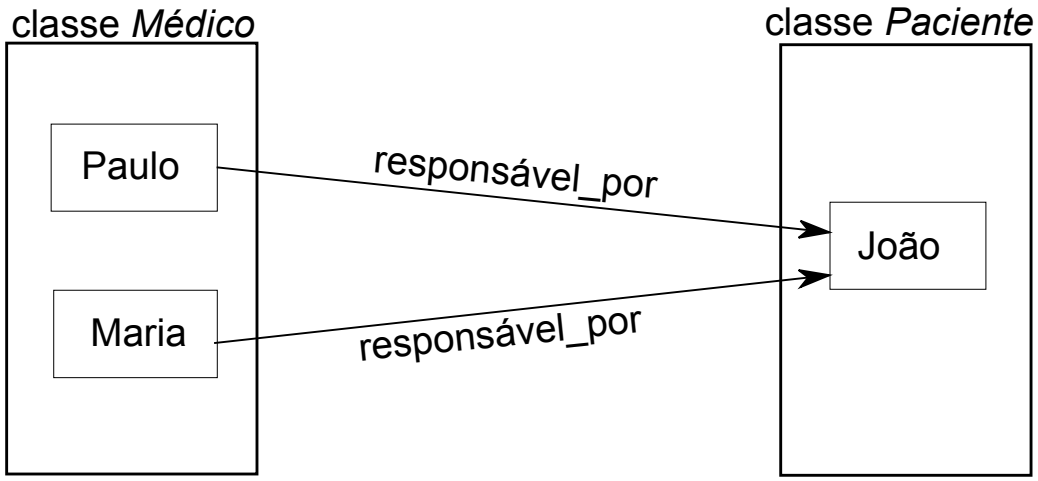

Figura 5.5: Restrição do tipo temValor

Conforme mostra a imagem acima, neste exemplo existem dois indivíduos da classe *Médico* que possuem uma relação com um paciente através da propriedade *responsável_por*. Deve-se salientar qualquer um dos indivíduos envolvidos poderiam fazer parte de outra relação, seja com eles mesmos ou com outros não definidos no caso.

5.8 Sumário do capítulo

Este capítulo apresentou em detalhes a ontologia que representa o ambiente *homecare*. Foram descritas as classes com suas propriedades de dado (atributos) e propriedades de objeto (relação entre duas classes). Também foram descritas algumas das regras de inferência desenvolvidas com a linguagem SWRL e as consultas com a linguagem SQWRL, assim como as restrições cujo objetivo é validar os elementos da ontologia tornando-a fiel ao conhecimento do ambiente o qual ela representa.

A ontologia faz parte da metodologia proposta pelo trabalho e é de fundamental importância para o funcionamento do sistema pervasivo, uma vez que todas as informações referentes as entidades existentes no ambiente *homecare* serão armazenadas nela e, consequentemente, o sistema terá que manipulá-la durante seu fluxo de funcionamento. No capítulo 6 é apresentada a arquitetura de um sistema pervasivo de *homecare* que utiliza esta ontologia em seu fluxo de funcionamento.

6 MODELAGEM DE UMA ARQUITETURA PARA SISTEMAS *HOMECARE* PERVASIVOS

Este capítulo apresenta a modelagem de uma arquitetura de um sistema de saúde pervasivo que será aplicado a ambientes *homecare*. Para o detalhamento da arquitetura, será utilizada a ontologia de um ambiente *homecare* descrita no Capítulo 5.

6.1 Arquitetura para sistema *homecare*

O objetivo principal do sistema *homecare* pervasivo é agilizar o trabalho de profissionais da saúde no processo de tratamento de seus pacientes. O sistema, com base nas informações captadas do contexto do ambiente, é capaz de sugerir ações a serem tomadas pelos profissionais em seus fluxos de trabalho. Desta forma, este capítulo tem por objetivo descrever uma arquitetura para um ambiente *homecare*. A figura 6.1 apresenta a arquitetura de um sistema de saúde pervasivo voltado para este tipo de ambiente.

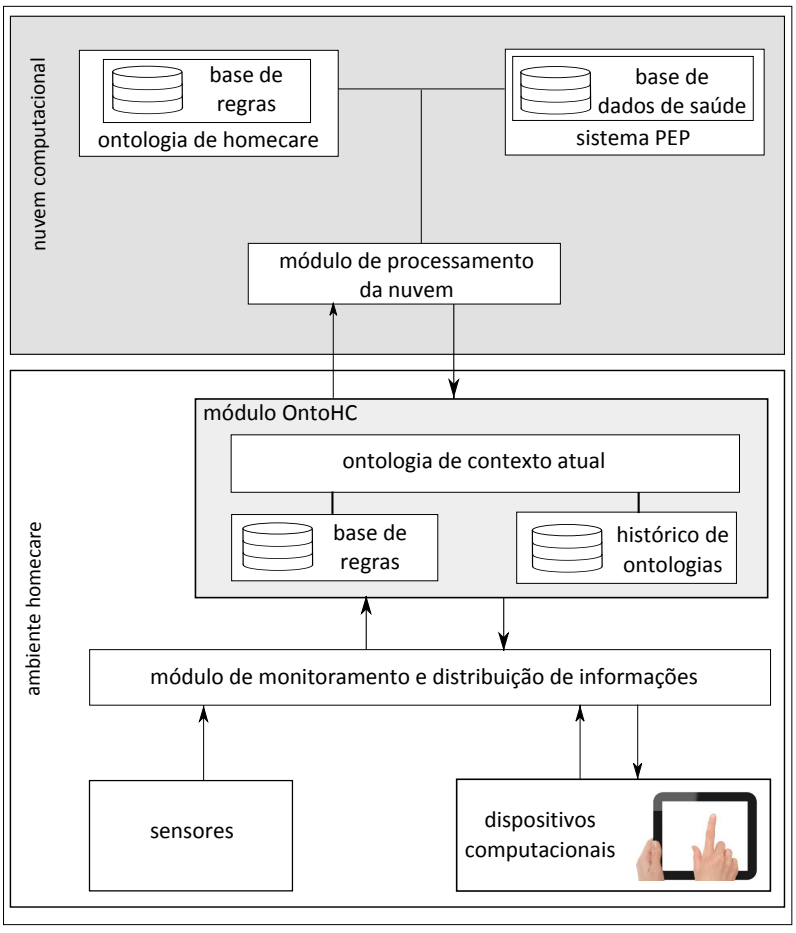

Figura 6.1: Arquitetura do sistema *homecare*

Nesta arquitetura, existem dois domínios os quais devem trocar informações sobre as entidades envolvidas. O primeiro deles refere-se a uma nuvem computacional que armazena informações referentes ao hospital pervasivo. Nesta nuvem encontram-se a ontologia que representa o ambiente *homecare* descrita no Capítulo 5, juntamente com uma base que armazena as regras de inferência e as consultas sobre a mesma; e o sistema de prontuário eletrônico de pacientes (PEP) com sua base de dados. Neste ponto, a ontologia não possui nenhuma instância, ou seja, é apenas uma estrutura ontológica composta pelas entidades e relações que possam existir em um ambiente *homecare*. A partir dela será criada uma ontologia formada apenas pelas entidades de um contexto atual específico para que sejam disparadas as aplicações para os usuários.

A nuvem computacional também possui um *módulo de processamento das informações* que ali circulam. É neste módulo que será criada a ontologia formada apenas por entidades de um contexto específico. A comunicação entre as partes do sistema é feita através de envio e recebimento de arquivos XML os quais possuem informações relevantes sobre determinada situação. A utilização destes arquivos no sistema será detalhada ao longo do Capítulo. Desta forma, para fazer o processamento dos arquivos XML que chegam até ele, o módulo conta com um *parser* que permite a manipulação dos mesmos. Este *parser* recebe um arquivo XML, contendo *tags* referentes a uma situação específica (por exemplo, identificadores de entidades detectadas por sensores) e manipula estas informações da forma adequada (por exemplo, utiliza identificadores para disparar uma regra referente a uma entidade). Além disso, este módulo possui um sistema que pode ser baseado na *Web* o qual é utilizado pelos profissionais que trabalham diretamente na casa do paciente.

O funcionamento do sistema ocorre da seguinte forma: no ambiente *homecare* existem sensores os quais realizam varreduras frequentes a procura de mudanças de contexto que de alguma forma possam afetar o tratamento médico do paciente. Dentre estas mudanças pode-se citar: chegada do médico para sua visita diária, alteração de sinais vitais do paciente, troca de turno de enfermeiros e técnicos de plantão, horário que um determinado medicamento deve ser administrado e horário que um paciente deve realizar uma tarefa. Quando detectadas informações deste tipo, elas são inseridas em um documento XML e enviadas para o *módulo de monitoramento e distribuição de informações*.

O *módulo de monitoramento e distribuíção* é o responsável por tratar da maneira adequada as informações enviadas pelos sensores e enviá-las para o *módulo OntoHC* ou para os dispositivos computacionais. Primeiramente as informações são enviadas para o *módulo OntoHC* para

que este verifique se a ontologia de contexto atual que ali está armazenada possui informações das entidades detectadas pelos sensores. Caso possua, este módulo busca em sua base regras que possam ser executadas envolvendo tais entidades para que aplicações sejam disparadas para serem utilizadas pelos usuários. Um contexto em específico refere-se a situação atual do que está acontecendo no ambiente *homecare*, por exemplo, quais entidades estão presentes e que relações existem entre elas. Desta forma, uma ontologia de contexto atual refere-se a todas as classes e propriedades que representam as entidades presentes no ambiente *homecare* em um determinado momento.

Porém, caso as entidades detectadas pelos sensores não estejam representadas na ontologia, ela será enviada para o repositório de ontologias e uma nova será criada, a qual possui as classes e relações referentes as entidades detectadas. Para isto, o *módulo OntoHC* envia um arquivo XML com informações das instâncias para o *módulo de processamento da nuvem*. Este módulo por sua vez cria uma nova estrutura ontológica formada pelas classes e relações referentes as entidades detectadas e preenche esta estrutura com informações atualizadas provindas do sistema PEP.

Neste momento, o *módulo de processamento da nuvem* envia para o *módulo OntoHC* uma ontologia que representa o contexto atual do ambiente em questão. O *módulo OntoHC* utiliza a base de regras de inferência e consultas para verificar se existem aplicações que possam ser disparadas para serem utilizadas pelos usuários do ambiente, sejam eles, clínicos, acompanhantes ou o próprio paciente. Caso exista, o *módulo de monitoramento e distribuição de informações* é comunicado e este envia a aplicação para um dos dispositivos computacionais para que o usuário possa utilizá-la, interagindo com o sistema através de interfaces. Um exemplo deste processo acontece quando o *módulo OntoHC* verifica, através da consulta na base de regras, que existe um relacionamento entre um indivíduo da classe *Médico* e um indivíduo da classe *Exame* através da propriedade *visualizado_por* que está marcado como *false*, ou seja, existe um exame do paciente que o médico ainda não visualizou. O *módulo OntoHC* informa isto ao *módulo de monitoramento e distribuição*, o qual envia uma notificação para o dispositivo computacional mais próximo do médico, perguntando se ele deseja analisar o exame naquele momento.

Cada uma das partes que compõem a arquitetura do sistema *homecare* pervasivo são descritas separadamente a seguir.

6.2 Sistema de Prontuário Eletrônico de Pacientes - PEP

A nuvem computacional possui um módulo com o sistema de Prontuário Eletrônico de Pacientes (PEP) que possui uma base de dados com informações gerais sobre o estado de saúde dos pacientes como, por exemplo, seu histórico clínico e os medicamentos que atualmente estão sendo administrados a ele, bem como as aplicações para manipular estas informações. As informações da base de dados do sistema PEP são utilizadas para preencher os campos da estrutura da ontologia de contexto atual que será criada, os quais serão chamados de indivíduos ou instâncias. Somente após instanciar informações vindas da base do PEP, é que o *módulo de processamento da nuvem* envia a nova ontologia para ser manipulada pelo *módulo OntoHC*.

Nesta arquitetura, propõe-se a utilização do sistema PEP em uma nuvem computacional (VAQUERO et al., 2008), uma vez que ao hospedá-lo nas nuvens, não apenas um ambiente *homecare* poderá utilizar o mesmo, mas qualquer outro ambiente *homecare* que decidir por fazê-lo. Em outras palavras, com o sistema PEP hospedado nas nuvens, qualquer paciente que possui seu histórico médico armazenado naquela base de dados, pode receber tratamento clínico em sua casa. Além disso, se um paciente troca de ambiente de tratamento, por exemplo, sai de sua casa e passa a receber os cuidados na casa de um parente, o sistema PEP não sofre alterações.

Mais especificamente em relação a arquitetura aqui proposta, um ganho significativo com a utilização do sistema em uma nuvem computacional é a possibilidade de alcançar o nível de abstração proposto pela computação pervasiva, onde os usuários (médicos, enfermeiros ou acompanhantes) acessam suas aplicações e as informações desejadas em qualquer lugar e a qualquer tempo, sem preocupações em relação a infraestrutura computacional que mantém o sistema em funcionamento, a qual fica "invisível" aos usuários. Dessa forma qualquer dispositivo com acesso a web que contenha um browser poderá acessar as aplicações do PEP.

De fato, não é necessário que o sistema esteja sendo executado em uma nuvem computacional para esta centralização da informação. O mesmo poderia estar em um servidor *Web* do próprio hospital. Grande parte das aplicações destes sistemas não necessitam de um grande processamento, elas basicamente tratam entrada e saída de dados. Desta forma, um servidor *Web* normal poderia ser o suficiente para o funcionamento deste sistema. Contudo, a computação nas nuvens abre também a possibilidade de acesso às aplicações mais robustas para utilização de ferramentas de auxílio a diagnóstico, predição de doenças, etc. Estas sim, necessitam de um maior processamento e podem utilizar as informações contidas no sistema PEP como fonte para

o seu raciocínio.

Além disso, teria-se os ganhos padrão da computação em nuvem, como a manutenção dos sistemas ou elasticidade de processamento de acordo com a necessidade. Neste caso apenas as informações de pacientes se encontram neste módulo, as informações operacionais do hospitais, referentes a custos por exemplo, se encontram em cada hospital separadamente.

6.3 Sensores

Em geral, um ambiente pervasivo deve possuir sensores que monitoram o local a procura de mudanças relevantes para o correto funcionamento do sistema. Em um ambiente *homecare* ocorre da mesma maneira. É necessário que os sensores dispostos no ambiente consigam detectar mudanças no contexto para que de alguma forma sejam utilizadas em prol dos usuários, sejam eles pacientes, clínicos ou acompanhantes.

Estes sensores deve ser capazes de, por exemplo, detectar movimento de pessoas, mudança de local de objetos ou, ainda, horário em que o usuário deve executar uma tarefa, e transformar estes dados em informações relevantes que poderão ser utilizadas da maneira mais adequada durante o fluxo de funcionamento do sistema. Ao detectar uma mudança relevante, os sensores captam informações referentes às entidades envolvidas e as armazenam em um documento XML e posteriormente os envia para o *módulo de monitoramento e distribuição de informações*. A estrutura é apresentada na figura 6.2.

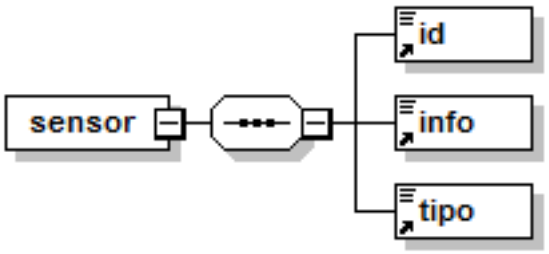

Figura 6.2: Esquema do arquivo XML que informa mudanças de contexto (sensor.xml)

O documento XML acima é formado por três *tags* que armazenam informações relativas ao evento detectado. A primeira *tag* é chamada *id* e refere-se a um identificador do sensor. Esta *tag* é necessária, pois o ambiente em questão pode possuir mais de um tipo de sensor e isto deve ser levado em consideração para o correto funcionamento do sistema. Por exemplo, um evento detectado por um sensor que monitora os sinais vitais do paciente deve ser tratado de uma forma, uma vez que uma alteração brusca nos sinais pode representar uma crise no estado

de saúde do paciente e, consequentemente, deve ser resolvido com urgência. Já um sensor que detecta a chegada e saída de pessoas do ambiente nem sempre representa um caso de urgência.

A segunda *tag* refere-se a informação coletada como, por exemplo, um valor de temperatura de um paciente. Essa *tag* guarda informação referente a uma instância da ontologia. A última *tag* armazena o tipo de informação coletada, por exemplo, o identificador de uma pessoa. Essa *tag* guarda informação referente ao atributo de uma classe da ontologia ao qual pertence a informação da segunda *tag*. Por exemplo, caso seja detectada a temperatura de 37.5 de um paciente, o conteúdo do arquivo terá na *tag info* o valor *37.5* e na *tag tipo* o valor *Temperatura*.

Deve-se deixar claro que para cada entidade detectada em um contexto será gerado um arquivo XML correspondente. Isto significa dizer que, se por exemplo, um paciente está no quarto sendo acompanhado por um enfermeiro, e neste momento entra um médico no ambiente, os sensores detectarão esta mudança de contexto e será criado um arquivo XML apenas para o médico. Isto porque as outras entidades detectadas (instância de *Paciente e Enfermeiro*) já estão mapeadas na ontologia do *OntoHC*. Este arquivo XML é enviado para o *módulo de monitoramento e distribuição de informações*.

para cada dos envolvidos (paciente, médico e enfermeiro).

6.4 Módulo de monitoramento e distribuição de informações

O *módulo de monitoramento e distribuição de informações* tem dois objetivos principais: (a) verificar junto ao *módulo OntoHC* se as entidades detectadas pelos sensores já estão mapeadas na ontologia de contexto atual e (b) disparar aplicações que interagem com os usuários através dos dispositivos computacionais possibilitando-os realizar suas tarefas.

Este módulo é o responsável pelo controle da informação no ambiente. É através dele que as informações oriundas dos sensores conseguem trafegar entre as diferentes camadas do sistema pervasivo.

Assim como o *módulo de processamento da nuvem*, o *módulo de monitoramento e distribuição de informações* também possui um *parser* que o permite manipular separadamente cada uma das informações dos arquivos XML recebidos dos sensores. Ele utiliza informações das *tags* para verificar junto ao *módulo OntoHC* se a estrutura da ontologia de contexto atual ali armazenada contém as entidades detectadas pelos sensores. Caso contenha, o *módulo OntoHC* acessa sua base de regras a procura de ações que possam ser executadas envolvendo uma ou mais entidades detectadas pelos sensores e notifica o *módulo de monitoramento e distribuição*.

Este, por sua vez, irá se comunicar com os dispositivos computacionais os quais executarão a aplicação mais adequada para auxiliar o usuário na realização de uma tarefa.

Esta comunicação é feita através do envio de um arquivo XML contendo informações provindas do *Módulo OntoHC*. A figura 6.3 apresenta a estrutura deste arquivo:

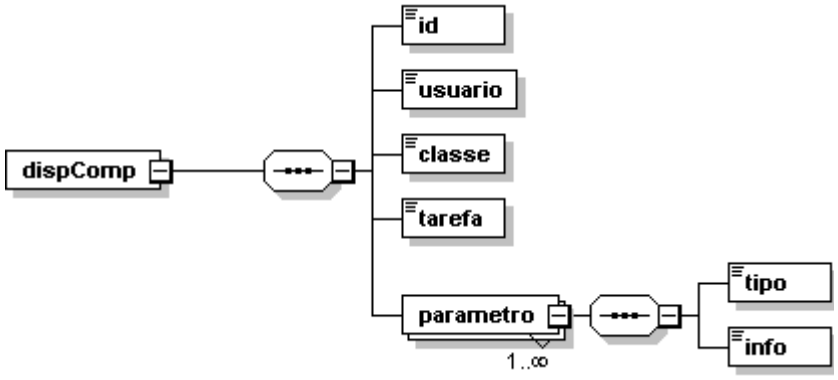

Figura 6.3: Esquema do arquivo XML com informações geradas para interação do usuário com o sistema (dispComp.xml)

O arquivo é formado por cinco elementos. O primeiro deles refere-se ao código identificador do dispositivo. Esta informação é necessária para que o sistema saiba para qual dispositivo enviar uma determinada aplicação. O segundo elemento refere-se ao usuário que irá interagir com o sistema. Nessa *tag* é armazenado um código identificador do usuário. O terceiro elemento, guarda a classe da ontologia a qual pertence a instância armazenada na *tag* usuário. O quarto elemento refere-se a tarefa que o usuário irá realizar através do dispositivo computacional. O conteúdo desta *tag* representa uma propriedade de objeto na ontologia, a qual liga dois indivíduos. Por último, o elemento *parâmetro* terá outras duas *tags*, as quais podem armazenar uma ou mais informações necessárias para a execução da tarefa. A *tag tipo* representa o tipo de informação que está sendo passado por parâmetro, que na ontologia é uma propriedade de dado (atributo). Esta *tag* pode conter, por exemplo, nome de exame, sintoma ou nome de um remédio. Por fim, a *tag info* representa uma instância armazenada na propriedade da *tag tipo*. Um cenário que descreve a utilização deste arquivo XML é quando o *módulo OntoHC* verifica que a temperatura do paciente é de 38,9°C, ou seja, acima de um limite máximo seguro pré-estabalecido. O *módulo de monitoramento e distribuição de informações* envia para o dispositivo computacional mais próximo do médico este arquivo XML onde nas *tags tipo* e *info* do elemento *parâmetro* terão as informações *temperatura* e *38,9* respectivamente. Com base nestas informações o médico pode tomar a decisão mais adequada para que o problema seja resolvido.

6.5 Módulo OntoHC

O *Módulo OntoHC* é o responsável pela manipulação da ontologia que representa um contexto atual específico do ambiente *homecare*. Além da ontologia, este módulo possui uma base de dados que contém um conjunto de regras de inferência e consultas e também um repositório onde são armazenadas as ontologias que já foram utilizadas por algum contexto em particular. Estas regras de inferência podem ser executadas enquanto um usuário está realizando uma tarefa. Por exemplo, quando um médico prescreve um novo medicamento para seu paciente, uma nova instância da classe *Medicamento* é criada e deve ser inserida na ontologia através da execução de uma regra. Este processo é gerenciado automaticamente pelo sistema pervasivo. Além disso, esta base também será utilizada para buscar informações da ontologia de contexto atual e enviá-las para nuvem computacional para atualização do sistema PEP.

Quando o *módulo de monitoramento e distribuição* envia informações referentes às entidades detectadas pelos sensores para o *módulo OntoHC*, este utiliza o conjunto de consultas SQWRL armazenados em sua base para verificar se tais entidades já estão mapeadas na ontologia. Caso estejam, ele verifica quais ações podem ser disparadas e comunica o *módulo de monitoramento e distribuição*. Porém, se a ontologia de contexto atual armazenada no *OntoHC* não possuir as classes, uma nova ontologia deve ser criada com as entidades necessárias. Para isto, o *módulo de OntoHC* deve se comunicar com a nuvem computacional informando o *módulo de processamento* sobre quais entidades devem estar presentes na nova ontologia. Esta comunicação é feita através do envio de um arquivo XML criado pelo *módulo OntoHC* a partir das informações provindas do *módulo de monitoramento e distribuição*. A figura 6.4 apresenta a estrutura deste arquivo:

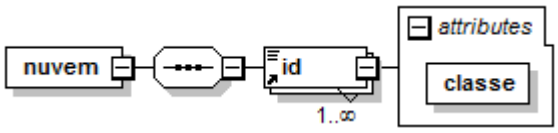

Figura 6.4: Esquema do arquivo XML que informa as classes presentes no ambiente no contexto atual (nuvem.xml)

Este documento XML possui o nome da classe e identificador do indivíduo (instância) que foi detectado pelo sensor. O *módulo de processamento da nuvem* recebe apenas um único arquivo XML por contexto modificado. Ou seja, neste arquivo estão informações das entidades envolvidas. Utilizando o exemplo anterior, em que o médico chega no quarto onde estão o paciente e o enfermeiro de plantão, este arquivo conterá o nome das classes *Médico, Paciente e*

Enfermeiro juntamente com o identificador de cada indivíduo. Com isto, o *módulo de processamento da nuvem* é capaz de criar uma ontologia contendo apenas informações referentes a estes indivíduos (atributos, relações, restrições, regras de inferência e consultas).

Ao utilizar regras de inferência sobre a ontologia de contexto atual que está armazenada no *módulo OntoHC*, é possível que novas informações sobre entidades sejam geradas. Um exemplo disso pode ser visto quando o médico prescreve um novo remédio para o paciente "João". Quando isto acontece, uma nova instância da classe *Medicamento* é criada, a qual terá relacionamentos com o indivíduo "João" da classe *Paciente* através das propriedades *prescrito_para e toma* (*Medicamento prescrito_para Paciente e Paciente toma Medicamento*). Por se tratarem de informações relativas ao processo de tratamento clínico do paciente, elas devem ser inseridas na ficha dele no sistema PEP. Desta forma, antes de o *módulo de processamento da nuvem* criar uma nova ontologia, as informações que estavam representadas na ontologia de contexto atual devem ser enviadas ao sistema PEP. Para isso, junto com o arquivo *nuvem.xml*, o *módulo OntoHC* também envia para o *módulo de processamento da nuvem* um outro arquivo XML com informações de cada entidade representada na ontologia de contexto atual. Estas informações serão inseridas no sistema PEP. A descrição detalhada deste arquivo será feita na seção 6.6.

Ao mesmo tempo em que o *módulo OntoHC* envia os arquivos XML para serem utilizados na nuvem computacional, ele também envia a ontologia que estava armazenada ali para um repositório de ontologias localizado dentro do próprio módulo. Este repositório é formado pelas ontologias já criadas e utilizadas pelo sistema pervasivo. Assim, quando o *módulo OntoHC* receber da nuvem a nova ontologia de contexto atual já instanciada, será possível manipulá-la da forma mais adequada, seja através da execução de regras de inferência, consultas, ou até mesmo ficar a espera de um novo evento no ambiente *homecare* que seja relevante no tratamento médico do paciente.

6.6 Módulo de processamento da nuvem computacional

Este módulo é o responsável pelo processamento que acontece na nuvem computacional. É através dele que o sistema consegue atualizar o sistema PEP, e depois buscar informações atualizadas. O *módulo de processamento da nuvem* também é o responsável por acessar a ontologia de *homecare* ali armazenada para criar uma nova ontologia apenas com classes específicas referentes a um contexto em questão.

Para realizar estes processamentos, o módulo recebe arquivos XML enviados pelo *módulo*

OntoHC. O *módulo de processamento da nuvem* recebe o arquivo *pep.xml* e, com ajuda do *parser*, consegue manipular as informações da ontologia separadamente e enviá-las para o sistema PEP. Isto garante que cada vez que uma nova ontologia for criada e instanciada com informações do sistema PEP, que ela terá informações atualizadas sobre os envolvidos. A figura 6.5 representa a estrutura deste arquivo.

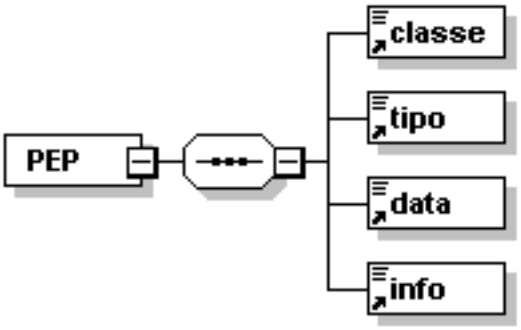

Figura 6.5: Esquema do arquivo XML que descreve os campos oriundos da base de dados do PEP (pep.xml)

Este arquivo XML é formado por quatro *tags* as quais armazenam informações referentes a cada entidade representada. Dessa forma, o *módulo OntoHC* deve enviar ao *módulo de processamento da nuvem* um arquivo para cada uma das entidades que estavam mapeadas na ontologia de contexto atual. A primeira *tag* refere-se a classe a qual pertence as informações. Esta *tag* é necessária, pois é a partir desta informação que a classe será identificada na ontologia. A segunda *tag* refere-se ao tipo de informação que está sendo enviada. Em outras palavras pode-se dizer que esta *tag* representa uma propriedade de dado da classe a qual pertence a informação que está sendo enviada. Por sua vez, a terceira *tag* armazena a data em que ocorreu a modificação naquela instância da ontologia. Ela é necessária, pois o *módulo de processamento da nuvem* irá comparar seu conteúdo com o que já existe no sistema PEP e só irá atualizá-lo caso a data seja diferente. A última *tag* guarda a instância da entidade, ou seja, a qual indivíduo da ontologia que estas informações do arquivo XML pertencem.

Após a atualização da ficha do paciente no sistema PEP, o *módulo de processamento da nuvem* cria a nova ontologia em um arquivo chamado *OntoHC.owl*. A partir das informações presentes no arquivo *nuvem.xml* que havia sido enviado pelo *módulo de monitoramento e distribuição*, *módulo de processamento da nuvem* monta uma nova estrutura ontológica apenas com informações do contexto atual através de processamento com XSL, ou seja, ela busca na ontologia de *homecare* apenas a estrutura das classes, atributos, relações e restrições das entidades detectadas pelos sensores. Com XSL é possível acessar o documento OWL, e selecionar

determinados elementos, criando uma nova ontologia. Estes elementos são relativos às entidades presentes no ambiente *homecare* em um momento específico, caracterizando um *contexto atual*. Depois de criada esta nova estrutura ontológica, o módulo faz consultas na base de dados do sistema PEP à procura de informações referentes àquelas entidades para instanciar esta nova ontologia. Após criar e instanciar esta nova ontologia, ela é enviada para o *módulo OntoHC* para que seja manipulada da forma adequada. Neste ponto, o sistema possui uma ontologia com informações referentes as entidades presentes no contexto atual armazenada em um módulo localizado no ambiente *homecare*. Dessa forma, para que uma aplicação seja executada, o sistema não precisa mais se comunicar com a nuvem computacional a menos que uma nova entidade seja detectada no ambiente. Isto, consequentemente, melhora o desempenho do sistema, uma vez que são utilizados apenas recursos locais para seu funcionamento.

6.7 Dispositivos computacionais

Este módulo representa os dispositivos computacionais que, de alguma forma, possibilitam a interação entre o sistema *homecare* e seus usuários. Entre estes dispositivos pode-se citar telas sensíveis ao toque, *smartphones*, tablets e equipamentos de monitoramento de sinais, enfim, qualquer dispositivo computacional fixo ou móvel, que possua uma capacidade de processamento e com memória suficiente para suportar as aplicações disponíveis no sistema pervasivo. É através destes dispositivos que os usuários poderão interagir com o sistema para utilizar suas aplicações na realização de suas tarefas de trabalho ou rotineiras.

O funcionamento dos dispositivos computacionais no ambiente *homecare* pode ser visto através do exemplo que vem sendo utilizado ao longo do Capítulo em que o médico entra no quarto onde estão o paciente e o enfermeiro. Neste momento, o *módulo OntoHC* constata que existe na ontologia uma relação entre uma instância da classe *Exame* e um indivíduo da classe *Médico* através da propriedade *visualizado_por* que está marcado como *false*, ou seja, o médico em questão ainda não analisou um exame que havia solicitado. O *módulo OntoHC* envia uma notificação sobre isto ao *módulo de monitoramento e distribuição de informações*. Ao receber esta notificação, o *módulo de monitoramento e distribuição* o arquivo *dispComp.xml* para o dispositivo computacional mais próximo do médico, informando sobre tal situação. O médico pode aceitar ou recusar a sugestão de realizar esta tarefa. Caso ele queira visualizar o exame, o dispositivo computacional envia para o *módulo de monitoramento e distribuição* um arquivo XML com informações referentes a esta situação. A figura 6.6 apresenta a estrutura

deste arquivo:

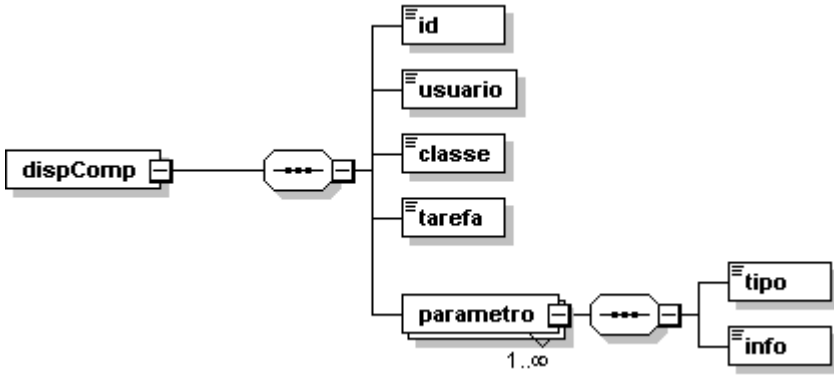

Figura 6.6: Esquema do arquivo XML com informações geradas a partir da interação do usuário com o sistema (dispComp.xml)

Este arquivo possui a mesma estrutura do *dispComp.xml*, porém agora contendo informações necessárias para a realização da tarefa, ou seja, entre os parâmetros que compõem o arquivo, terá uma informação sobre a decisão do médico de visualizar o exame. O primeiro elemento deste arquivo refere-se um código identificador do dispositivo que está sendo utilizado pelo usuário. Este código é necessário para que o sistema consiga verifica se aquele tipo de dispositivo suporta as solicitações do usuário. Por exemplo, se o médico está utilizando seu *smartphone* para acessar o sistema, ele não poderá analisar um exame de raio-X, uma vez que esta imagem não é suportada por este tipo de aparelho. O segundo elemento refere-se ao usuário. Esta *tag* armazena um código que identifica qual instância da classe *Pessoa* está utilizando o sistema. O terceiro elemento refere-se a classe à qual a instância pertence. Estas duas *tags* são necessárias para que o sistema possa verificar quais tipos de aplicações podem ser executadas para cada usuário. Por exemplo, se quem estiver utilizando o sistema for um indivíduo da classe *Técnico em enfermagem*, ele não poderá prescrever um medicamento para um paciente. O quarto elemento armazena uma *string* que identifica qual tarefa o usuário está querendo solicitar. Se o médico deseja visualizar um exame do paciente, esta *tag* conterá a informação *visualizar exame*. Esta *string* é comparada com o elemento RDFS:Label da ontologia. Para cada tarefa descrita na ontologia existe um elemento deste tipo associado.

O último elemento corresponde a uma lista de um ou mais parâmetros necessários para que uma determinada tarefa seja executada. Este parâmetro é formado por outras duas *tags*: *tipo* e *info*. A *tag tipo* corresponde ao nome da propriedade de dado que está sendo passada por parâmetro. Ela pode conter nomes como *nome do medicamento, posologia, nome de exame ou tipo de tratamento*. O conteúdo deste elemento varia de acordo com a tarefa que o médico

deseja realizar. O segundo elemento corresponde a informação em si, ou seja, a instância da propriedade de dado definida na *tag* anterior. O conteúdo dela pode ser *paracetamol* para *nome de medicamento*; *8h em 8h horas* para *posologia*; *endoscopia* para *nome de exame* ou *repouso absoluto* para *tipo de tratamento*.

Podem existir casos em que para execução de uma tarefa sejam necessários mais de um conjunto de informações deste tipo. Isto significa que será necessário mais de um conjunto de parâmetros. Isto está previsto e pode ser visto na cardinalidade do elemento *parâmetro* da figura acima.

Este arquivo é enviado para o *módulo de monitoramento e distribuição*, que busca junto ao *módulo OntoHC* informações sobre o referido exame e então apresentadas no dispositivo computacional mais próximo daquele médico.

6.8 Considerações do capítulo

Este capítulo apresentou uma arquitetura para ambientes *homecare* pervasivos, a qual faz parte da metodologia proposta neste trabalho, detalhando cada um dos módulos computacionais que a compõem. Foram descritas cada uma das partes da arquitetura, assim como o fluxo de funcionamento do sistema.

A utilização desta arquitetura, juntamente com a ontologia proposta no Capítulo 5 possibilita o desenvolvimento de aplicações eficientes, transformando as casas dos pacientes em ambientes *homecare* pervasivos. Com as aplicações desenvolvidas a partir da metodologia proposta, acredita-se que o fluxo de tratamento de pacientes e o trabalho de profissionais que os assistem diretamente em suas casas será aprimorado.

A seguir são descritos dois casos de estudo a fim de exemplificar o funcionamento do sistema, bem como validar a arquitetura.

7 CASOS DE ESTUDO E RESULTADOS

A metodologia proposta foi validada a partir da utilização de dois casos de estudo que descrevem situações bastante distintas de um ambiente *homecare*. O primeiro caso ilustra um cenário onde duas pessoas sofrem de doenças de baixa gravidade e apenas precisam de monitoramento constante. Assim, o sistema monitora a saúde deles e também utiliza aplicações para lembrá-los de realizar suas tarefas rotineiras. Já o segundo caso ilustra um cenário em que um paciente está acamado e deve ter sempre um acompanhante consigo. Neste exemplo o sistema utiliza suas aplicações para auxiliar os usuários (médico, enfermeiro acompanhante) na execução de tarefas relacionadas ao tratamento do paciente. Durante a descrição dos casos detalha-se o fluxo de funcionamento do sistema pervasivo.

A figura 7.1 mostra uma imagem da arquitetura enumerada de acordo com o fluxo de funcionamento do sistema. Desta forma, nos casos de estudo, cada etapa possui um número associado o qual corresponde a uma parte da execução do sistema.

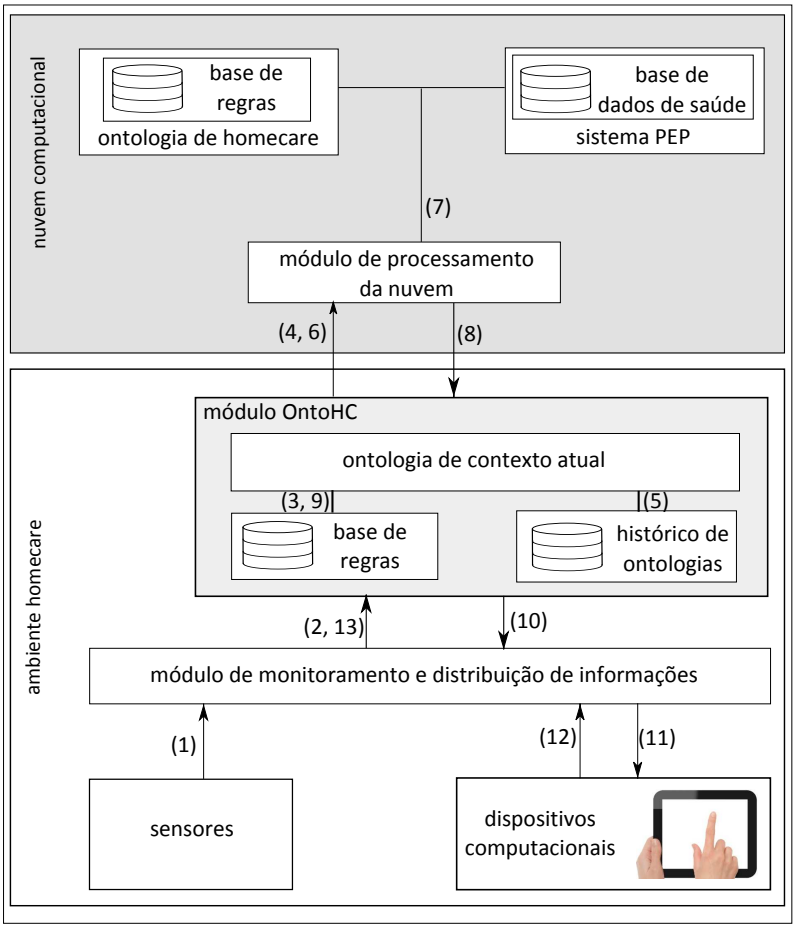

Figura 7.1: Arquitetura do sistema *homecare*

Os casos de estudo são detalhados nas seções a seguir.

7.1 Caso de estudo 1: Lembrete sobre realização de tarefas rotineiras

O primeiro caso de estudo foi retirado de (ZARGHAMI et al., 2011). Neste cenário, John e Marie são pacientes com uma Doença Pulmonar Obstrutiva Crônica (DPOC) de baixa gravidade, ou seja, podem realizar suas tarefas diárias normalmente, desde que tenham monitoramento médico constante. A qualidade de vida é melhorada quando estão mais ativos e controlando seu peso. No entanto, como estão levando suas vidas de forma mais ativa, praticando exercícios regularmente, é necessário um acompanhamento de saturação de oxigênio por questões de segurança. Se o nível de saturação cai abaixo do limite mínimo aceitável em seu quadro clínico, podem ocorrer complicações quanto a seu estado de saúde, levando-os para o hospital onde terão um tempo de tratamento mais longo e mais caro. Além disso, John sofre de um distúrbio auditivo e Marie tem problemas de visão e os dois também sofrem de amnésia e precisam ser lembrados das tarefas que devem realizar em suas rotinas diárias. Nancy é uma acompanhante de idosos profissional e é responsável por criar e gerenciar os serviços de *homecare* instalados em suas casas.

Além dos sensores que monitoram mudanças no contexto do ambiente, suas casas também são equipadas com um *tablet PC*, *PDA*, um distribuidor automático de medicamentos (*dispenser*) e um medidor de saturação de oxigênio. Entre os serviços personalizados existem dois de lembretes para John e Marie, um para avisá-los de fixar o medidor de oxigênio antes de sair de casa e outro para lembrá-los de tomar os medicamentos no horário correto. O sistema adapta os serviços para usar o *Tablet* como dispositivo de lembretes para Marie e para usar lembretes com volume alto no *Tablet* ou *PDA* para John. John e Marie têm preferências diferentes no que diz respeito a forma de receber o alerta para suas tarefas. Por exemplo, quando John está acompanhado, ele prefere receber o alerta sem volume, apenas com seu dispositivo vibrando. Se por algum motivo, o *Tablet* deixa de funcionar em tempo de execução, o sistema pervasivo detecta esta falha e adapta o serviço de alerta para que Marie receba o lembrete no *PDA*. Se o nível de saturação cai abaixo do limite mínimo, um alarme é enviado para o hospital responsável pelos serviços de *homecare* daquele paciente. Em caso de frequentes lembretes ignorados pelos usuários, o sistema pervasivo adapta este serviço. Isto pode acontecer devido a uma piora no problema de audição de John, e então o sistema aumenta o volume do alerta e notifica Nancy. Nancy visita diariamente os dois pacientes para acompanhar a evolução de seu estado

de saúde. Assim ela tem conhecimento suficiente para re-configurar os serviços como, por exemplo, quanto a necessidade de adicionar um novo medicamento no tratamento e atualizar o serviço de lembretes incluindo aquele novo medicamento durante a execução. O trabalho de Nancy é necessário apenas em casos como este em que, por exemplo, um novo remédio deve ser inserido na rotina de administração de medicamentos. Em outras palavras, Nancy realiza tarefas que o sistema não pode fazer automaticamente. Neste caso, se um novo medicamento é prescrito ao paciente, ela o insere no *dispenser* e comunica o sistema sobre isto, informando também a posologia do mesmo. A partir deste ponto o sistema volta a trabalhar sozinho emitindo alertas ao paciente para lembrá-lo de tomar este novo remédio.

O sistema pervasivo de *homecare* que utiliza a metodologia proposta neste trabalho funciona da seguinte forma para este estudo de caso: Primeiramente deve-se deixar claro que, até o momento em que a situação do caso de estudo ocorra, *módulo OntoHC* armazena uma ontologia com informações sobre o paciente *John* e também informações referentes ao seu monitoramento clínico, como, temperatura, sinais vitais, frequência cardíaca e nível de saturação de oxigênio, além de informações como hora atual, a qual é necessária para caso seja necessário tomar algum medicamento. Em um cenário em que o *paciente John* deve realizar a tarefa de *buscar seu neto na escola* às *16:30*, os sensores que monitoram o ambiente detectam uma mudança de contexto, neste caso pode ser a hora de emitir um lembrete sobre esta tarefa, e enviam informações sobre as entidades envolvidas neste contexto para o *módulo de monitoramento e distribuição de informações* (1). Estas informações são enviadas através de arquivos *sensor.xml*, conforme mostram os códigos a seguir:

```
<!--arquivo referente à entidade Paciente-->
<?xml version="1.0" encoding="UTF-8"?>
<sensor>
    <id>SR3338</id>
    <info>PC001</info>
    <tipo>Paciente</tipo>
</sensor>

<!--arquivo referente a entidade hora-->
<?xml version="1.0" encoding="UTF-8"?>
<sensor>
```

```
 <id>SR3344</id>
 <info>16:30</info>
 <tipo>Tarefa</tipo>
</sensor>
```

Os arquivos XML acima mostram que para cada entidade detectada no ambiente será gerado um arquivo XML correspondente, sendo um para o horário e um para o paciente que encontra-se no ambiente.

O *módulo de monitoramento e distribuição*, por sua vez, verifica junto ao *módulo OntoHC* se as entidades detectas estão mapeadas na ontologia (2). Neste caso, a classe *Paciente* já está mapeada conforme descrito anteriormente, porém a classe *Tarefa* ainda não, sendo assim, uma nova ontologia deve ser criada. Ao consultar sua base de regras e constatar que a classe *Tarefa* não está mapeada (3), o *módulo OntoHC* insere no arquivo *pep.xml* informações referentes as entidades da ontologia e o envia para o *módulo de processamento da nuvem* para que o sistema PEP seja atualizado (4). Um exemplo desta situação pode ser vista quando informações referentes a temperatura do paciente devem ser enviadas ao PEP. Neste cenário o atributo *temperatura* da classe *SinaisVitais* possui um determinado valor. Estas informações devem ser inseridas em um arquivo XML como mostra o código a seguir:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<PEP>
 <classe>SinaisVitais</classe>
 <tipo>temperatura</tipo>
 <data>12062008124530</data>
 <info>37,5</info>
</PEP>
```

Após mandar o arquivo *pep.xml* para o *módulo de processamento da nuvem*, o *módulo OntoHC* envia a ontologia que ele está armazenando para o *repositório de ontologias* (5). Feito isto, ele cria um arquivo *nuvem.xml* com identificadores das classes e instâncias que precisam estar mapeadas na nova ontologia. Este arquivo tem a seguinte estrutura.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<nuvem>
```

```
    <id class="Tarefa">TF012</id>
    <id class="Paciente">PC001</id>
  </nuvem>
```

Na ontologia, todos os indivíduos das classes *Tarefa* e *Paciente* possuem um código identificador. Para este caso de estudo, definiu-se o código *PC001* para *John* e *TF012* para a tarefa *"buscar neto na escola"* Este arquivo XML será enviado para a nuvem computacional onde será manipulado pelo *módulo de processamento da nuvem* (6). O primeiro trabalho deste módulo é tratar as informações do arquivo *pep.xml* e enviá-las para o sistema PEP. Feito isto, ele começa a trabalhar com as informações do arquivo *nuvem.xml*. Ele pega as informações contidas ali e cria uma estrutura ontológica, utilizando XSL, apenas com as entidades deste arquivo e então busca na base do sistema PEP informações referentes a estas entidades para preencher a ontologia (7). A seguir é apresentado um trecho do código OWL com informações das entidades detectadas.

```
<Paciente rdf:ID="John">
  <realiza>
    <Tarefa rdf:ID="buscar_neto">
      <nome_tarefa rdf:datatype=
      "http://www.w3.org/2001/XMLSchema#string"
      >buscar neto na escola</nome_tarefa>
      <hora_início rdf:datatype=
      "http://www.w3.org/2001/XMLSchema#time"
      >16:30:00</hora_início>
      <hora_fim rdf:datatype=
      "http://www.w3.org/2001/XMLSchema#time"
      >16:45:00</hora_fim>
      <realizada_por rdf:resource="#John"/>
    </Tarefa>
  </realiza>
  <sob_responsabilidade_de>
    <Médico rdf:ID="Joana">
      <responsavel_por rdf:resource="#John"/>
    </Médico>
</Paciente>
```

Esta nova ontologia é então enviada de volta ao ambiente *homecare* onde será manipulada pelo *módulo OntoHC*. Neste momento o módulo faz consultas a sua base de regras e verifica que existe uma relação entre uma instância da classe *Tarefa* (id = TF012), e uma instância da classe *Paciente* (id = PC001) através da propriedade *realiza*, a qual está marcada como *false* e cujo atributo *hora* de *Tarefa* corresponde à detectada pelo sensor (9). Em outras palavras, o paciente John deve realizar uma tarefa de buscar neto na escola naquela determinada hora.

Com isto, o *módulo OntoHC* comunica o *módulo de monitoramento e distribuíção de informações* sobre tal situação (10). Este módulo então envia uma notificação para o *tablet* de John lembrando-o deste compromisso (11). John pode interagir com o sistema apenas aceitando a notificação e marcando-a como lembrada, por exemplo. Quando ele faz isto, o arquivo dispComp.xml passa a ter as seguintes informações:

```
<?xml version="1.0" encoding="UTF-8"?>
<dispComp>
 <id>TB001</id>
 <usuario>PC001</usuario>
 <classe>Paciente</classe>
 <tarefa>TF012</tarefa>
 <parametro>
 <tipo>lembrada</tipo>
 <info>text</info>
 </parametro>
</dispComp>
```

Este arquivo é enviado para o *módulo de monitoramento e distribuição* (12) que trata as informações da forma correta e as envia para o *módulo OntoHC* para que a ontologia seja atualizada (13).

A partir deste caso de estudo pode-se concluir que o sistema pervasivo para ambientes *homecare* baseado na metodologia proposta por este trabalho pode auixiliar os usuários em suas tarefas mais rotineiras, como no exemplo descrito acima, onde um paciente com problemas de memória recebe lembretes sempre que uma tarefa deve ser realizada. Pode-se perceber também que o usuário não teve dificuldade alguma para utilização do sistema, uma vez que, este trabalhou da forma menos intrusiva possível, apenas aparecendo quando deveria avisar John sobre sua tarefa, utilizando assim, as premissas da computação pervasiva.

7.2 Caso de estudo 2: Agilizando realização de tarefas médicas

Outro caso de estudo foi criado para descrever o funcionamento do sistema e pode ser descrito quando um paciente chamado Pedro está repousando no quarto de sua casa com um acompanhante chamado Jorge. No quarto existe uma televisão digital que está sendo usada para entreter os usuários. Este cenário já foi detectado pelos sensores, e desta forma, as informações referentes a estas entidades já estão mapeadas na ontologia de contexto atual localizada no *módulo OntoHC*.

Em certa hora do dia, Paulo, que é o médico responsável chega a casa do paciente para sua visita diária. Este médico havia solicitado um exame de sangue, mas não teve tempo de analisá-lo antes da visita. Quando ele entra no quarto, os sensores que monitoram o ambiente o identificam como uma nova entidade relevante no tratamento do paciente e enviam informações das entidades presentes para o *módulo de monitoramento e distribuição* (1) através de um arquivo XML cuja estrutura pode ser vista abaixo.

```
<!--arquivo referente a entidade Médico-->
<?xml version="1.0" encoding="UTF-8"?>
<sensor>
 <id>SR3344</id>
 <info>MD3012</info>
 <tipo>Médico</tipo>
</sensor>
```

O *módulo de monitoramento e distribuição* recebe tais informações e envia para o *módulo OntoHC* para que este verifique se as entidades detectadas já estão mapeadas na ontologia (3). Neste caso, apenas a classe *Médico* não está mapeada. Quando *módulo OntoHC* verifica junto a sua base de regras que a entidade não está mapeada, ele cria um arquivo XML contendo informações sobre as entidades presentes na ontologia de contexto atual e as envia para a nuvem computacional (4) para que o PEP seja atualizado. Depois disso, o *módulo OntoHC* envia sua ontologia para o *repositório de ontologias* (5). O arquivo XML usado para atualização do PEP pode, por exemplo, conter informações sobre os batimentos cardíacos do paciente, como mostra o código a seguir.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<PEP>
```

```
 <classe>SinaisVitais</classe>
 <tipo>batimento</tipo>
 <data>12102011132610</data>
 <info>96</info>
</PEP>
```

Depois de enviar a ontologia atual para o repositório, o *módulo OntoHC* cria outro arquivo XML com informações para que seja criada uma nova ontologia, a qual representa o contexto envolvendo as entidades detectadas pelos sensores e envia este arquivo para o *módulo de processamento da nuvem* (6). A estrutura do arquivo nuvem.xml deste caso é a seguinte:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<nuvem>
 <id class="Paciente">PC002</id>
    <id class="Acompanhante">AC2020</id>
 <id class="Medico">MD3012</id>
 <id class="DispositivoComputacional">TV003</id>
</nuvem>
```

O *módulo de processamento da nuvem* utiliza informações do arquivo *pep.xml* para atualizar a base de dados do sistema PEP. Feito isto, o módulo manipula o arquivo *nuvem.xml* para criar uma nova estrutura ontológica, através de XSL. Esta estrutura possui as classes, relações, atributos e restrições que envolvem as entidades detectas. A nova ontologia é instanciada com as informações atualizadas do PEP (7) e então é enviada para o *módulo OntoHC* para que seja manipulada da forma adequada (8). O código a seguir mostra um trecho da ontologia OWL com informações das entidades detectadas.

```
<?xml version="1.0" encoding="UTF-8"?>
<Paciente rdf:ID="Pedro">
 <doença rdf:datatype=
         "http://www.w3.org/2001/XMLSchema#string"
       >Diabetes</doença>
 <sob_responsabilidade_de>
    <Médico rdf:ID="Paulo">
```

```
        <responsavel_por rdf:resource="#Pedro"/>
     <id rdf:datatype=
              "http://www.w3.org/2001/XMLSchema#string"
            >MD3012</id>
    </Médico>
    </sob_responsabilidade_de>
    <id rdf:datatype=
             "http://www.w3.org/2001/XMLSchema#string"
           >PC002</id>
    <auxiliado_por>
    <Acompanhante rdf:ID="Jorge">
      <id rdf:datatype=
             "http://www.w3.org/2001/XMLSchema#string"
             >AC2020</id>
      <auxilia rdf:resource="#Pedro"/>
      <grau_de_afinidade rdf:datatype=
               "http://www.w3.org/2001/XMLSchema#string"
              >alto</grau_de_afinidade>
    </Acompanhante>
    </auxiliado_por>
    <possui>
    <Sintoma rdf:ID="Febre"/>
    </possui>
   </Paciente>
```

Quando o *módulo OntoHC* recebe esta nova ontologia, ele faz consultas sobre ela utilizando sua base de regras (9) e verifica que existe uma relação entre o indivíduo Pedro da classe *Paciente* e um indíviduo da classe *Sangue* (id = ES1213) a qual é subclasse de *Exame* através da propriedade *pertence_a*, ou seja, existe um exame de sangue cujo id é ES1213 que pertence ao paciente Pedro. O módulo também verifica que existe uma relação entre o indivíduo Paulo da classe *Médico* e um indivíduo da classe *Sangue* (id = ES1213) através da propriedade *visualizado_por* que está marcado como *false*. Ou seja, o médico Paulo ainda não analisou um exame de sangue que havia solicitado.

Ao constatar isto, o *módulo OntoHC* avisa o *módulo de monitoramento e distribuição de informações* sobre tal situação (10) e este envia uma notificação para o médico através de um dispositivo computacional (11), neste caso a TV digital. Neste notificação, o sistema pergunta ao médico se ele deseja visualizar o exame naquele momento. Caso ele aceite um arquivo XML é criado contendo estas informações e enviado de volta para o *módulo de monitoramento e distribuição* (12). A estrutura do arquivo é a seguinte:

```
<?xml version="1.0" encoding="UTF-8"?>
<dispComp>
 <id>TV003</id>
 <usuario>MD3012</usuario>
 <classe>Médico</classe>
 <tarefa>visualizado_por</tarefa>
 <parametro>
 <tipo>Sangue</tipo>
 <info>ES1213</info>
 </parametro>
</dispComp>
```

O *módulo de monitoramento e distribuição* recebe este arquivo XML, e avisa o *módulo OntoHC* sobre o que o médico deseja fazer (13). Este módulo então faz uma consulta em base de regras para buscar o exame de sangue e envia informações desse exame de volta para o *módulo de monitoramento e distribuição* (repetindo o passo 10 do fluxo) e este, por sua vez, envia o exame para o dispositivo computacional, para que o médico possa analisá-lo (repetindo o passo 11 do fluxo).

A partir deste caso de estudo pode-se concluir que a metodologia proposta possibilita a criação de sistemas pervasivos que agilizam o fluxo de tratamento clínico de pacientes em ambientes *homecare*. Neste caso, o médico, ao chegar na casa do paciente, ainda não tinha analisado resultados de um exame que havia solicitado. O sistema detectou automaticamente esta situação, e prontamente interagiu com o profissional perguntando se ele gostaria de realizar esta ação. De forma mais ampla, pode-se verificar que o médico não perdeu tempo procurando a ficha com a documentação médica do paciente. Com isto, o processo de realização da tarefa de analisar o exame aconteceu mais rapidamente e, consequentemente, o trabalho do profissional foi aprimorado.

7.3 Considerações do capítulo

A partir da análise dos dois casos de estudo apresentados neste capítulo, é possível afirmar que a metodologia proposta para arquiteturas de sistemas para ambientes *homecare* pervasivos pode ser aplicada em qualquer situação onde o paciente precise (ou possa) receber cuidados médicos diretamente em sua casa. Isto acontece, pois como a proposta do trabalho sempre foi aprimorar os serviços médicos nestes ambientes, buscou-se criar uma arquitetura genérica, e que pudesse ser aplicada nas mais diversas situações. É necessário destacar que, para que esta arquitetura funcione, é preciso que o ambiente em questão possua uma infra-estrutura com dispositivos computacionais que o caracterizem como um ambiente pervasivo.

Este capítulo apresentou dois estudos de casos bastante distintos envolvendo entidades diferentes em situações diferentes, com o objetivo de validar a metodologia proposta pelo trabalho. Um dos casos foi retirado da literatura e o outro foi criado pelo grupo de pesquisa com o objetivo de provar que um sistema pervasivo baseado nesta arquitetura funciona em qualquer situação.

8 TRABALHOS RELACIONADOS

Este capítulo tem por objetivo apresentar alguns trabalhos relacionados a este no que diz respeito a utilização de conceitos de computação pervasiva na proposição de aplicações para ambientes *homecare*.

O primeiro trabalho (BAJO et al., 2010) aborda a temática da utilização de sistemas multi-agentes no desenvolvimento de aplicações pervasivas para a área de *homecare* devido a alta dinamicidade que este tipo de ambiente possui. No trabalho, os autores apresentam uma arquitetura para supervisionar e monitorar pacientes em casa, fornecendo serviços para escala de trabalho dos funcionários envolvidos no tratamento, sistema de alerta, localização e identificação. Para isto, os médicos e enfermeiros receberam PDAs e celulares para comunicação com o ambiente, o qual possui sensores, controle de acesso, e câmeras. O projeto utiliza ontologias para representação do conhecimento desse domínio, porém os autores não fazem uso de inferências sobre ontologias. Além disso, a arquitetura também difere da deste trabalho pelo fato de não suportar aplicações médicas tais quais o paciente utiliza em um hospital pervasivo, ou seja, o médico não pode analisar e solicitar exames ou prescrever medicamentos por exemplo, como foi detalhado em um dos casos de estudo.

Já o segundo trabalho (ISERN et al., 2009) propõe um plano de intervenção individual baseado em uma arquitetura já existente apresentada em (CAMPANA; ANNICCHIARICO; RIAñO, 2006). Os autores abordam apenas serviços personalizados para pacientes idosos e que estejam de alguma forma incapazes. Na arquitetura, são utilizadas ontologias OWL para descrever os atores envolvidos, porém sem realizar inferências sobre as mesmas, apenas usando-as para consultas. Outro detalhe que difere o trabalho do apresentado nesta dissertação é o fato de não disponibilizar aplicações aos usuários, uma vez que a proposta do trabalho é a criação de um plano de intervenção de pacientes em casos de urgência.

Por sua vez, o trabalho de (ZARGHAMI et al., 2011) apresenta uma plataforma que fornece serviços para ambientes *homecare* levando em consideração a questão da dinamicidade e restrições encontradas neste tipo de domínio. Esta plataforma trata de questões como gerenciamento de contexto para adaptar suas aplicações aos usuários e também utiliza dispositivos computacionais móveis para interação do sistema com o usuário. A diferença entre esta plataforma e a metodologia do presente trabalho é que ela não gerencia as aplicações que o sistema oferece. São descritas aplicações fornecidas por terceiros, ou seja, nem todos os serviços são disponibili-

zados pela plataforma proposta. Além disso, os autores não prevêem a utilização de aplicações médicas, ou seja, o fato de levar serviços disponíveis em hospitais até a casa do paciente. Na metodologia proposta nesta dissertação de mestrado isto é possível devido a comunicação entre o ambiente *homecare* e o sistema PEP, localizado na nuvem computacional. Os autores também utilizam ontologias apenas para gerenciamento de planos de serviço, diferentemente deste trabalho de dissertação que faz uso de ontologias para gerenciar o conhecimento do ambiente *homecare*, além de utilizar regras de inferência para manipular ontologias.

A tabela 8.1 apresenta um comparativo das características deste trabalho com outros encontrados na literatura que também propõem o desenvolvimento de arquiteturas de serviços para ambientes *homecare*.

Tabela 8.1: Comparativo entre projetos de aplicações pervasivas com *homecare*

Projeto	Gerencia contexto	Ontologias OWL	Inferência em ontologias SWRL	Fornece aplicações médicas
(BAJO et al., 2010)	X	X	-	-
(ISERN et al., 2009)	X	X	-	-
(ZARGHAMI et al., 2011)	X	X	-	X (em parte)
Metodologia proposta	**X**	**X**	**X**	**X**

Desta forma, pode-se concluir, de acordo com as características apresentadas pelos trabalhos acima relacionados, que a metodologia proposta neste trabalho de dissertação é mais completa, uma vez que une as características das demais, além de suprir as deficiências que as mesmas apresentam.

9 CONCLUSÃO

Segundo (ANDRADE; LOBO, 2007) problemas relacionados ao Programa Previdenciário do Brasil, gastos com planos de saúde e problemas no atendimento em hospitais públicos impedem que as pessoas tenham acesso a um tratamento clínico de qualidade. O desenvolvimento de aplicações voltadas à *homecare* aparece como um incentivo para que mais pessoas decidam pelo tratamento clínico diretamente em suas casas. Este tipo de serviço garante um maior conforto aos pacientes, uma vez que estarão em suas casas, além da possibilidade de ficar mais tempo com seus entes queridos. Com a popularização dessa forma de tratamento, as filas de pacientes em hospitais tendem a diminuir e, consequentemente, o serviço disponibilizado por estas instituições tende a melhorar consideravelmente.

A realização desta pesquisa foi motivada principalmente devido a escassez de aplicações computacionais no tratamento de pacientes que recebem cuidados médicos diretamente em casa. Mesmo com a popularização de dispositivos computacionais, ainda não existem aplicações pervasivas capazes de interagir automaticamente com o usuário, levando em consideração o contexto em que se encontram. O principal objetivo deste tipo de sistema é melhorar a qualidade de vida dos pacientes e agilizar o trabalho dos profissionais que tratam dos mesmos.

Este trabalho apresentou detalhadamente uma metodologia que propõe uma arquitetura que possa ser utilizada como base para o desenvolvimento de sistemas pervasivos para ambientes *homecare* através da utilização de conceitos de computação pervasiva e ontologias aplicados à área da saúde, bem como a definição de ambientes e sistemas de *homecare*. Para o projeto, foi desenvolvida uma ontologia a qual representa o conhecimento existente em um ambiente de *homecare*. A ontologia é formada por classes, propriedades e restrições, as quais representam as entidades e relações entre estas entidades que compõem este tipo de ambiente.

Para a manipulação da ontologia foram desenvolvidas regras de inferência através da linguagem SWRL e consultas sobre a mesma através de SQWRL. O sistema pervasivo utiliza estas regras para deduzir novas informações sobre outras já descritas na ontologia referentes a um contexto em específico. A partir daí o sistema pode utilizar as consultas para buscar informações na ontologia para que aplicações possam ser executadas e serem utilizadas pelos usuários.

Com o objetivo de descrever o fluxo de funcionamento de um sistema baseado nesta arquitetura e também para validação da mesma, foram apresentados dois casos de estudo simulados envolvendo diferentes entidades em situações distintas.

O primeiro caso traz um cenário em que dois pacientes sofrem de problemas pulmonares de baixa gravidade e que podem realizar as tarefas rotineiras sem problemas, apenas necessitando de um monitoramento quanto ao nível de saturação do oxigênio. Além disso um dos pacientes tem problemas auditivos e o outro de visão, e os dois também sofrem de amnésia. Dessa forma, o sistema deve emitir lembretes sobre suas tarefas diárias e dos horários que devem tomar seus remédios. Quando o sistema detecta a hora de determinado remédio ou tarefa que deve ser executada, ele envia um alerta para o dispositivo móvel dos pacientes lembrando-os daquilo.

O segundo caso refere-se a um cenário onde um médico que havia solicitado exames clínicos chega a casa do paciente sem ter analisado os resultados dos mesmos. Quando ele entra no quarto o sistema detecta que o contexto do ambiente foi alterado com esta nova entidade e infere que o médico tem exames por analisar. O sistema interage com o médico através da TV disposta no quarto do paciente perguntando se ele deseja realizar a tarefa naquele momento. Caso aceite, o sistema disponibiliza os resultados dos exames na tela.

Com esta arquitetura, portanto, acredita-se que é possível aprimorar o tratamento de pacientes que, por preferência ou necessidade, recebem tratamento médico diretamente em casa. Como foi descrito nos casos de estudo, o sistema pervasivo pode auxiliar seus usuários nas mais diversas situações utilizando desde aplicações consideradas mais simples, como no caso dos lembretes sobre atividades dos usuários, até mesmo em situações mais complexas, onde o sistema deve oferecer auxílio aos profissionais envolvidos no tratamento de um paciente. Por exemplo, no segundo caso de estudo, o fato de o médico não ter conseguido analisar o resultado do exame previamente não atrapalhou de forma alguma o tratamento do paciente, pelo contrário, o sistema foi capaz de inferir que esta atividade ainda deveria ser executada e tomou as providências para que o médico pudesse realizá-la antes de prosseguir o tratamento.

Como trabalhos futuros e sequência do projeto pretende-se focar em questões não abordadas até o momento. Por exemplo, até aqui estão sendo discutidas questões de segurança e privacidade das informações que circulam pelo sistema. Neste ponto do projeto assume-se que o ambiente *homecare* possui uma infraestrutura computacional que possibilite o desenvolvimento de aplicações pervasivas e que o mesmo trate das questões relacionadas a segurança e privacidade. Dessa forma, para este trabalho assume-se que as informações cheguem ao sistema íntegras e cabe ao mesmo manipulá-las da maneira correta.

Desta forma, como trabalho futuro pretende-se focar em questões como autorização de visualização das informações do paciente, além de questões de como ocorre a autenticação, se

quem está utilizando o sistema pode ou não mostrar informações para outras pessoas, assim como outras questões que envolvem tolerância a falhas e ética.

Também como trabalho futuro, planeja-se desenvolvimento de um servidor *Web* para gerenciar o fluxo de informações que trafega pelo sistema pervasivo.

Por fim, a realização de testes simulados através da construção de um ambiente *homecare* real. Desta forma será possível analisar detalhadamente o funcionamento do sistema, para então disponibilizá-lo para usuários reais.

REFERÊNCIAS

ABOWD, G. D.; ATKESON, C. G.; HONG, J.; LONG, S.; KOOPER, R.; PINKERTON, M. Cyberguide: a mobile context-aware tour guide. **Wirel. Netw.**, Hingham, MA, USA, v.3, p.421–433, October 1997.

ANDRADE, L. M.; MARTINS, E. C.; CAETANO, J. A.; SOARES, E.; BESERRA, E. P. Atendimento humanizado nos serviços de emergência hospitalar na percepção do acompanhante. **Revista Eletrônica de Enfermagem**, [S.l.], v.1, p.151–157, 2009.

ANDRADE, M.; LOBO, E. L. A Importância da visita Domiciliária para o Idoso Portador de Doença Crônica após a Alta Hospitalar. **Informe-se em promoção da saúde**, [S.l.], v.3, n.2, p.12–14, 2007.

ARAUJO, R. B. de. Computação Ubíqua: princípios, tecnologias e desafios. **XXI Simpósio Brasileiro de Redes de Computadores**, [S.l.], 2003.

BAJO, J.; FRAILE, J. A.; PéREZ-LANCHO, B.; CORCHADO, J. M. The THOMAS architecture in Home Care scenarios: a case study. **Expert Syst. Appl.**, Tarrytown, NY, USA, v.37, p.3986–3999, May 2010.

BARDRAM, J. E. Applications of context-aware computing in hospital work: examples and design principles. In: ACM SYMPOSIUM ON APPLIED COMPUTING, 2004., 2004, New York, NY, USA. **Proceedings...** ACM, 2004. p.1574–1579. (SAC '04).

BOSSEN, C.; JØRGENSEN, J. B. Context-descriptive prototypes and their application to medicine administration. In: DESIGNING INTERACTIVE SYSTEMS: PROCESSES, PRACTICES, METHODS, AND TECHNIQUES, 5., 2004, New York, NY, USA. **Proceedings...** ACM, 2004. p.297–306. (DIS '04).

BRAY, T.; PAOLI, J.; SPERBERG-MCQUEEN, C. M.; MALER, E.; YERGEAU, F. **Extensible Markup Language (XML) 1.0 (Fifth Edition)**. 2008.

BRICKLEY, D.; GUHA, R. V.; MCBRIDE, B. **RDF Vocabulary Description Language 1.0**: rdf schema. 2004.

CABANA, M. D.; RAND, C. S.; POWE, N. R.; WU, A. W.; WILSON, M. H.; ABBOUD, P. A.; RUBIN, H. R. Why don't physicians follow clinical practice guidelines? A framework for improvement. **JAMA : the journal of the American Medical Association**, Department of Pediatrics, Robert Wood Johnson Clinical Scholars Program, Johns Hopkins School of Medicine, Baltimore, MD, USA. mcabana@umich.edu, v.282, n.15, p.1458–1465, Oct. 1999.

CAMPANA, F.; ANNICCHIARICO, R.; RIAñO, D. **Knowledge-Based HomeCare eServices for an Ageing Europe**. 2006.

CHEN, G.; KOTZ, D. **A Survey of Context-Aware Mobile Computing Research**. [S.l.: s.n.], 2000.

CHEN, H. **An Intelligent Broker Architecture for Pervasive Context-Aware Systems**. 2004. Tese (Doutorado) — University of Maryland, Baltimore County.

CONNOLLY, D. **Overview of SGML Resources**. 1995.

CONNOLLY, D.; HARMELEN, F. van; HORROCKS, I.; DEBORAH; MCGUINNESS, L.; PATEL-SCHNEIDER, P. F.; STEIN, L. A. **DAML+OIL (March 2001) Reference Description**. 2001.

DEPARTMENT, M. G.; GRUNINGER, M. Designing and Evaluating Generic Ontologies. In: IN PROCEEDINGS OF THE 12TH EUROPEAN CONFERENCE OF ARTIFICIAL INTELLIGENCE, 1996. **Anais...** [S.l.: s.n.], 1996.

DUARTE, O. C. M. B.; JUNIOR, M. B. F. **XML - Extensible Markup Language**. 2000.

FICKAS, S. Clinical requirements engineering. In: SOFTWARE ENGINEERING, 2005. ICSE 2005. PROCEEDINGS. 27TH INTERNATIONAL CONFERENCE ON, 2005. **Anais...** [S.l.: s.n.], 2005. p.28 – 34.

FREITAS, L. O.; GASSEN, J. B.; COPPETTI, M.; LIBRELOTTO, G. R. Aplicando Consultas em Ontologias para Tarefas Médicas. **VIII Simpósio de Informática do Planalto Médio**, [S.l.], 2008.

FREITAS, L. O.; MOZZAQUATRO, B.; AZEVEDO, R. P.; KURTZ, G.; PEREIRA, R.; MARTINI, R.; LIBRELOTTO, G. R. Uma modelagem ontológica de hospitais pervasivos aplicada ao OntoHealth. **Revista do CCEI**, [S.l.], 2011.

GARDE, S.; KNAUP, P. Requirements engineering in health care: the example of chemotherapy planning in paediatric oncology. **Requir. Eng.**, Secaucus, NJ, USA, v.11, p.265–278, August 2006.

GASSEN, J. B.; COPPETTI, M.; FREITAS, L. O.; MARTINS, E.; LIBRELOTTO, G. R. Construção de uma ontologia para um hospital. **VII Simpósio de Informática da Região Centro do RS**, [S.l.], 2008.

GLODBERG, K. H. **XML**: visual quickstart guide (2nd edition). [S.l.]: Peachpit Press, 2008.

GRAU, B. C.; HORROCKS, I.; MOTIK, B.; PARSIA, B.; PATEL-SCHNEIDER, P.; SATTLER, U. **OWL 2**: the next step for owl. 2008.

GROSSO, P.; WALSH, N. **XSL Concepts and Practical Use**. http://nwalsh.com/docs/tutorials/xsl/xsl/slides.html.

GROUP, W. O. W. **OWL 2 Web Ontology Language Document Overview**. 2009.

GRUBER, T. R. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. In: IN FORMAL ONTOLOGY IN CONCEPTUAL ANALYSIS AND KNOWLEDGE REPRESENTATION, KLUWER ACADEMIC PUBLISHERS, IN PRESS. SUBSTANTIAL REVISION OF PAPER PRESENTED AT THE INTERNATIONAL WORKSHOP ON FORMAL ONTOLOGY, 1993. **Anais...** Kluwer Academic Publishers, 1993.

GUARINO, N. Formal Ontology and Information Systems. In: 1998. **Anais...** IOS Press, 1998. p.3–15.

HORRIDGE, T.; KNUBLAUCH, H.; RECTOR, A.; STEVENS, R.; WROE, C. **A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools Edition 1.0. The University Of Manchester**. 2004.

HORROCKS, I.; PATEL-SCHNEIDER, P. F.; BOLEY, H.; TABET, S.; GROSOF, B.; DEAN, M. **SWRL**: a semantic web rule language combining OWL and RuleML. [S.l.]: DARPA DAML Program, 2004. Draft Version,
`http://www.daml.org/rules/proposal/`.

ISERN, D.; MORENO, A.; PEDONE, G.; SáNCHEZ, D.; VARGA, L. Z. **Home Care Personalisation with Individual Intervention Plans**. Berlin, Heidelberg: Springer-Verlag, 2009. 134–151p.

KNUBLAUCH, H.; FERGERSON, R. W.; NOY, N. F.; MUSEN, M. A. The Protégé OWL plugin: an open development environment for semantic web applications. In: 2004. **Anais...** Springer, 2004. p.229–243.

KOH, W.; MUI, L. An Information Theoretic Approach for Ontology-based Interest Matching. In: WORKSHOP ON ONTOLOGY LEARNING'01, 2001. **Anais...** [S.l.: s.n.], 2001. p.–1–1.

LAERUM, H.; FAXVAAG, A. Task-oriented evaluation of electronic medical records systems: development and validation of a questionnaire for physicians. **BMC Medical Informatics and Decision Making**, [S.l.], v.4, n.1, p.1, 2004.

LASSILA, O.; SWICK, R. **Resource Description Framework (RDF) Model and Syntax Specification,**. 1999.

LIBRELOTTO, G.; AUGUSTIN, I.; GASSEN, J.; KURTZ, G.; FREITAS, L. O.; MARTINI, R.; AZEVEDO, R. ONTOHEALTH: an ontology applied to pervasive hospital environments. In: CRUZ-CUNHA, M.; MOREIRA, F. (Ed.). **Handbook of Research on Mobility and Computing**: evolving technologies and ubiquitous impacts. [S.l.]: IGI Global, 2011. p.1077–1090. 10.4018/978-1-60960-042-6.ch065.

LIBRELOTTO, G.; FREITAS, L.; FIORIN, A.; MOZZAQUATRO, B.; PASETTO, L.; MARTINI, R.; AZEVEDO, R. de; PEREIRA, R. OntoHealth: a system to process ontologies applied to health pervasive environment. In: UBI-MEDIA COMPUTING (U-MEDIA), 2011 4TH INTERNATIONAL CONFERENCE ON, 2011b. **Anais...** [S.l.: s.n.], 2011b. p.59 –64.

LIBRELOTTO, G. R.; GASSEN, J. B.; FREITAS, L. O.; SILVEIRA, M. C.; SILVA, F. L.; AUGUSTIN, I.; HENRIQUES, P. R. Uma Ontologia aplicada a um Ambiente Pervasivo Hospitalar. In: CAPSI'08 – 8ª CONFERêNCIA DA ASSOCIAçãO PORTUGUESA DE SISTEMAS DE INFORMAçãO, 2008, Setubal, Portugal. **Anais...** [S.l.: s.n.], 2008.

LOPES, J. L. B. **Sensibilidade ao Contexto na Computação Pervasiva**: avaliando o uso de ontologias. [S.l.]: Universidade Católica de Pelotas - Escola de Informática, 2006.

MCGEE-LENNON, M. R. Requirements engineering for home care technology. In: PROCEEDING OF THE TWENTY-SIXTH ANNUAL SIGCHI CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS, 2008, New York, NY, USA. **Anais...** ACM, 2008. p.1439–1442. (CHI '08).

MCGUINNESS, D. L.; HARMELEN, F. van. **OWL Web Ontology Language Overview**. [S.l.]: W3C, 2004. (REC-owl-features-20040210).

MILLER, L.; MILLER, L.; SEABORNE, A.; SEABORNE, A.; REGGIORI, A.; REGGIORI, A. Three Implementations of SquishQL, a Simple RDF Query Language. In: 2002. **Anais...** [S.l.: s.n.], 2002. p.423–435.

MISKELLY, F. **Memorandum by Dr Frank Miskelly**. 2004.

O'CONNOR, M. J.; DAS, A. K. SQWRL: a query language for owl. In: OWLED, 2008. **Anais...** CEUR-WS.org, 2008. (CEUR Workshop Proceedings, v.529).

SAHA, D.; MUKHERJEE, A. Pervasive Computing: a paradigm for the 21st century. **COMPUTER**, [S.l.], v.36, n.3, p.25–31, 2003.

SATYANARAYANAN, M. Pervasive Computing: vision and challenges. **IEEE Personal Communications**, [S.l.], v.8, p.10–17, 2001.

SCHILIT, B.; THEIMER, M. Disseminating active map information to mobile hosts. **Network, IEEE**, [S.l.], v.8, n.5, p.22 –32, sep/oct 1994.

SHIBU, S.; SANJEEV, J. Pervasive Computing for Automobiles: an approach to maximize user convenience and safety using vanets. **International Journal of Computer and Electrical Engineering**, [S.l.], v.2, n.6, p.1053–1058, 2010.

STUDER, R.; BENJAMINS, V. R.; FENSEL, D. **Knowledge Engineering**: principles and methods. 1998.

VAQUERO, L. M.; RODERO-MERINO, L.; CACERES, J.; LINDNER, M. A break in the clouds: towards a cloud definition. **SIGCOMM Comput. Commun. Rev.**, New York, NY, USA, v.39, p.50–55, December 2008.

WANT, R.; FALCAO, V.; GIBBONS, J. The Active Badge Location System. **ACM Transactions on Information Systems**, [S.l.], v.10, p.91–102, 1992.

WEISER, M. The computer for the 21st century. **SIGMOBILE Mob. Comput. Commun. Rev.**, New York, NY, USA, v.3, p.3–11, July 1999.

YAMIN, A. **Arquitetura para um Ambiente de Grade Computacional Direcionado às Aplicações Distribuídas, Móveis e Conscientes do Contexto da Computação Pervasiva**. 2004. Tese (Doutorado) — Universidade Federal do Rio Grande do Sul, Porto Alegre, RS.

YAMIN, A. C.; BARBOSA, J. L. V.; AUGUSTIN, I.; SILVA, L. C. da; REAL, R. A.; GEYER, C. F. R.; CAVALHEIRO, G. G. H. Towards Merging Context-Aware, Mobile and Grid Computing. **IJHPCA**, [S.l.], v.17, n.2, p.191–203, 2003.

ZARGHAMI, A.; ESLAMI, M. Z.; SAPKOTA, B.; SINDEREN, M. van. Toward dynamic service provisioning in the homecare domain. In: PERVASIVE COMPUTING TECHNOLOGIES FOR HEALTHCARE (PERVASIVEHEALTH), 2011 5TH INTERNATIONAL CONFERENCE ON, 2011. **Anais...** [S.l.: s.n.], 2011. p.292 –299.