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

ASSOCIAÇÃO DO CONTEXTO DE INTERESSE DO USUÁRIO ÀS ATIVIDADES CLÍNICAS NA ARQUITETURA CLINICSPACE

DISSERTAÇÃO DE MESTRADO

Alencar Machado

Santa Maria, RS, Brasil
2010

ASSOCIAÇÃO DO CONTEXTO DE INTERESSE DO USUÁRIO ÀS ATIVIDADES CLÍNICAS NA ARQUITETURA CLINICSPACE

por

Alencar Machado

Dissertação apresentada ao Curso de Mestrado em Computação do Programa de Pós-Graduação em Informática (PPGI), Área de Concentração em Computação, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do grau de **Mestre em Computação**

Orientadora: Prof.ª Iara Augustin

Santa Maria, RS, Brasil

2010

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

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

ASSOCIAÇÃO DO CONTEXTO DE INTERESSE DO USUÁRIO ÀS ATIVIDADES CLÍNICAS NA ARQUITETURA CLINICSPACE

elaborada por
Alencar Machado

como requisito parcial para obtenção do grau de
Mestre em Ciência da Computação

COMISSÃO EXAMINADORA:

Iara Augustin, Dra.
(Presidente/Orientadora)

Renata Galante, Dra. (UFRGS)

Eduardo Piveta, Dr. (UFSM)

Santa Maria, 10 de dezembro de 2010.

AGRADECIMENTOS

É com muita satisfação que, após o término da elaboração deste trabalho, chega o momento de agradecer às pessoas que junto comigo fizeram esta pesquisa ser possível.

Agradeço a Deus por sempre de alguma forma, iluminar o meu caminho e colocar pessoas especiais para caminharem comigo.

Agradeço à minha noiva Michele, por compreender meus momentos de ausência e vivenciar toda esta caminhada, por estar sempre aqui ao meu lado. Com certeza seu apoio foi fundamental para essa realização.

Agradeço aos meus pais Edemar e Dalila, meus irmãos e sobrinhos, pois a convivência com eles me faz sempre querer mais e tentar ser uma pessoa melhor a cada dia.

Agradeço ao meu ex-professor Odaylson Eder, por ter um dia falado para procurar a professora Iara Augustin, que hoje considero uma grande amiga, pois somente com sua ajuda consegui chegar até aqui. Obrigado Iara Augustin por orientar-me neste trabalho, por ser a pessoa que realiza esse sonho de seus alunos, mostrou-me a importância de uma professora orientadora e amiga, sua orientação neste trabalho, somente ex alunos seus podem mensurar, espero que ainda possamos ter muitas tardes de conversas para troca de idéias, sempre acompanhados do chimarrão, assim, podendo continuar trabalhando juntos no GMob e escrevendo essa história.

RESUMO

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

ASSOCIAÇÃO DO CONTEXTO DE INTERESS DO USUÁRIO ÀS ATIVIDADES CLÍNICAS NA ARQUITETURA CLINICSPACE

Autor: Alencar Machado
Orientadora: Iara Augustin
Data e Local da Defesa: Santa Maria, 10 de dezembro de 2010

Atualmente, busca-se deixar os sistemas mais orientados ao usuário-final, diminuindo a distância entre a forma como o usuário realiza suas atividades e a modelagem destas nos Sistemas de Informação em Saúde, disponibilizando ao médico formas adaptativas e personalizadas de utilização, configuração e controle do sistema, baseadas em um histórico de uso ou em um perfil. Para atender tais requisitos, o projeto ClinicSpace prototipa uma arquitetura de software que é desenvolvida sob o ponto de vista do usuário (médico), orientada às atividades clínicas, consciente do contexto, baseada em tecnologias móveis e pervasivas, e utiliza técnicas da programação do usuário-final. Identificar o contexto que afeta uma atividade clinica dentro de um hospital, bem como identificar formas de o médico expressar seu interesse em um determinado contexto, associando contexto às suas atividades diárias, são os objetivos desta dissertação. A partir da identificação de alguns elementos básicos de contexto, entre eles paciente, dispositivo e recursos ambientais, foi criado a descrição ontológica deste domínio para ser associado às tarefas clínicas modeladas pelo médico. Para introduzir contexto associado às tarefas na arquitetura ClinicSpace, foi alterada a interface de edição de tarefas da arquitetura. Além disso, diversos serviços dos subsistemas envolvidos foram re-implementados na arquitetura, destacando-se o Serviço de Acesso a Tarefas, o qual foi re-implementado para suportar o carregamento da ontologia que descreve as tarefas e contexto, e, assim, disponibilizar estruturas computacionais (objetos Java) para utilização pela Interface de Edição de Tarefas e Contexto utilizada pelo médico. Testes de carga foram realizados para identificar o impacto que a solução utilizada introduz na arquitetura ClinicSpace, permitindo refiná-la e melhorá-la para uso pelo sistema.

Palavras-chave: Sistemas de Informação em Saúde, ClinicSpace, Contexto, tarefas clínicas, Serviço de Acesso a Tarefas, consciente do contexto, ontologias.

ABSTRACT

Master's Dissertation
Post-Graduate Program in Informatics
Federal University of Santa Maria

Association of the Context of User's Interesting to Clinical Activity in Architecture ClinicSpace

Author: Alencar Machado
Advisor: Iara Augustin
Santa Maria, December 10, 2010

Currently, we seek to make the systems more oriented to end-user, reducing the distance between how the user carries out its activities and modeling these in Information Systems in Health, the doctor providing adaptive and personalized forms of use, configuration and control system, based on historical usage, or a profile. To meet these requirements, the project ClinicSpace, prototyped a software architecture that is developed from the point of view of the user (doctor), led to clinical activities, context-aware, based on mobile technologies and pervasive, and uses the techniques of programming end-user. Identify the context that affects an activity within a hospital clinic, and identify ways to express your doctor interest in a particular context, linking context to their daily activities, are the goals of this dissertation. From the identification of some basic elements of context, including the patient, device and environmental resources, was created to describe this ontological domain to be associated with clinical tasks modeled by the doctor. For introduce context associated with the tasks in the architecture ClinicSpace, changed the editing interface tasks of architecture. Moreover, various departments of the subsystems involved have been re-implemented in architecture, highlighting the Access Service Task, which was re-implemented to support the loading of the ontology that describes the tasks and context, and thus provide structures computational (objects Java) to use the Interface Editing Tasks and Context used by the doctor. Load tests were conducted to identify the impact that the solution used in the architecture introduces ClinicSpace allowing refine it and refine it for use by the system.

Key-words: *Electronic HealthCare Systems*; context; clinical activities; Access Service Tasks, context-aware, ontology

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Program Interface*
AVU	Ambiente Virtual do Usuário
BDA	Base de Dados pervasiva das Aplicações
EHS	*Electronic Healthcare System*
IETC	Interface de Edição de Tarefas e Contexto
pEHS	*Pervasive Electronic HealthCare Systems*
SAT	Serviço de Acesso a Tarefas
SATA	Serviço de Acesso a Tarefas Ativas
SCT	Serviço de Contexto de Tarefas
SGDT	Subsistema de Gerenciamento Distribuído de Tarefas
SGT	Serviço de Gerenciamento de Tarefas
EXEHDA	*Execution Environment for Highly Distributed Applications*

SUMÁRIO

1 INTRODUÇÃO

A rápida evolução tecnológica está trazendo muitas mudanças na forma como se realizam as atividades diárias dos profissionais em saúde. Para o desenvolvimento de sistemas de informação em saúde, o usuário e a forma de organizar suas atividades são interpretados e remodelados por profissionais da computação; sendo assim, modifica-se a forma pessoal como cada usuário realiza suas atividades. Essa metodologia pode induzir à rejeição enfrentada pelos atuais sistemas de informação em saúde nos hospitais (JHA et al. 2009).

Atualmente, busca-se deixar os sistemas mais orientados ao usuário-final, diminuindo a distância entre a forma como o usuário realiza suas atividades e a modelagem destas nos Sistemas de Informação em Saúde - EHSs (*Electronic HealthCare Systems*), disponibilizando ao médico formas adaptativas e personalizadas de utilização, configuração e controle do EHS, baseadas em um histórico de uso do sistema ou em um perfil.

Convergindo para atender aos requisitos dinâmicos dos sistemas de informação em Saúde, têm-se os conceitos e as tecnologias da Computação Ubíqua. A Computação Ubíqua, (WEISER, 1991) propõe que a computação torne-se invisível ao usuário-final, seja onipresente e totalmente integrada ao ambiente real, não impondo restrições para sua utilização no auxílio à realização da atividade humana cotidiana. A computação deve acompanhar o usuário e adaptar-se ao seu perfil e dispositivo que este porta ou tem acesso no ambiente em que se encontra (AUGUSTIN et al. 2001). Estudos revelam que a Computação Ubíqua na Saúde (conhecida como *pHealth* ou *UbiHealth*) oferece vantagens competitivas aos provedores de serviços de saúde; em particular, aumenta a eficiência do serviço, a qualidade e melhora o gerenciamento da relação com o paciente (VARSHNEY, 2003).

O trabalho descrito nesta dissertação está inserido no escopo do projeto ClinicSpace, o qual propõe a prototipação de uma arquitetura de software de um sistema de informação em saúde ubíquo, baseada no ponto de vista do usário-final (médico). A arquitetura ClinicSpace tem como premissa o argumento que, para diminuir o impacto e interferência do sistema automatizado no ambiente hospitalar e na forma como cada usuário individualmente realiza suas atividades, e o consequente potencial de rejeição por parte destes, o sistema deve equilibrar pró-atividade (agir em nome do usuário) com personalização (forma individual de

cada um realizar sua atividades diárias) e ser desenvolvido sob o ponto de vista de quem o utilizará (orientado ao usuário-final).

Atuais Sistemas de Informação em Saúde são desenvolvidos a partir da visão (i) da organização hospitalar (EHR - *Electronic Health Record*, Prontuário Eletrônico do Paciente) ou (ii) do paciente (*Personal Health System*, Sistemas de Saúde Pessoal, tais como o *GoogleHealth*). A visão do médico não é atendida adequadamente nas modelagens de tais sistemas.

Assim, o projeto ClinicSpace tem como objetivo inovar na construção de EHS dando ao médico o controle e programação do sistema, adequando-o ao seu próprio estilo de trabalho. Para tal, utilizam-se conceitos oriundos da Programação pelo Usuário-Final (*End-User Programming*). Dessa forma, disponibiliza-se, ao médico, uma interface gráfica intuitiva para programação do sistema que o auxiliará na realização de suas atividades profissionais. A programação das atividades clínicas é baseada no conceito de composição de tarefas associadas a elementos do contexto de interesse do usuário. Logo, o médico não precisa ter consciência da estrutura computacional existente, que atua em *background*, nem aprender e adaptar sua atividade a como o EHS está projetado – como ocorre atualmente.

1.1 Problema e Objetivo

Quando uma pessoa realiza uma atividade, normalmente, leva-se em consideração o meio no qual a atividade será realizada, como pessoas e objetos (contexto). Tal característica faz com que, ao realizar uma atividade humana, sempre uma nova avaliação de como realizar essa atividade pode ser feita. Tratando-se de um ambiente hospitalar, onde a atividade de um médico sempre interage com algo voltado à saúde humana, o dinamismo da atividade torna-se mais complexo e difícil de ser gerenciado.

Considerando a premissa na modelagem da Arquitetura ClinicSpace de equilibrar entre a execução pró-ativa e a execução personalizável pelo médico das atividades diárias, torna-se necessário adicionar o conhecimento do meio ao qual a atividade será realizada, permitindo, assim, que esta atividade seja realizada com uma maior sensibilidade ao contexto. Considerando este escopo, a proposta desta dissertação é modelar o contexto e associá-lo às atividades clínicas, inserindo seu tratamento na arquitetura ClinicSpace. Dessa forma, as atividades do médico podem sofrer alterações a cada execução, dependendo do

contexto detectado no momento.

Mais especificamente, os objetivos dessa dissertação são: (i) analisar a modelagem dos subsistemas atuais do ClinicSpace, bem como o modo de interação entre eles; (ii) identificar elementos do contexto existente em um ambiente hospitalar e criar o modelo de contexto necessário para o ClinicSpace; (iii) implementar e documentar um protótipo de interface gráfica para uso de médicos, com a finalidade de associação do contexto de interesse de cada médico às suas atividades; (iv) projetar e modelar uma estrutura para armazenamento da associação do contexto à uma atividade clínica.

1.2 Organização do Texto

O restante do texto está organizado da seguinte maneira. No capítulo 2 são apresentados os conceitos relativos à Computação Ubíqua/Pervasiva na Saúde e ao contexto e sua representação através das diferentes modelagens existentes. O capítulo 3 apresenta a arquitetura ClinicSpace na qual este trabalho está inserido, abordando os conceitos de atividades clínicas e seu gerenciamento. O capítulo 4 apresenta a proposta da modelagem de contexto clínico e sua associação às atividades cotidianas do usuário médico, e descreve a necessária reestruturação dos serviços existentes na arquitetura ClinicSpace para tratar atividades associadas a contexto. No capítulo 5 é apresentado o relato da construção de um estudo de caso para análise dos resultados e das soluções empregadas. Finalmente, no capítulo 6 são destacadas as conclusões e as considerações finais sobre trabalhos futuros a serem desenvolvidos.

2 COMPUTAÇÃO UBÍQUA/PERVASIVA

Este capítulo aborda os aspectos relativos à Computação Ubíqua e Pervasiva, e projetos relativos à área de aplicação Computação Ubíqua na Saúde (UbiHealth). Aborda também os conceitos de Computação Ciente de Contexto considerando a Modelagem de Contexto.

2.1 Características das Aplicações Ubíqua/Pervasivas

Mark Weiser (1991) visualizou que no futuro se estaria vivendo em um mundo repleto de sensores espalhados através dos mais simples objetos, como xícaras, canetas, roupas, janelas, etc. A Computação Ubíqua, proposta por ele, leva em consideração que o ambiente computacional não deve impor restrições ao usuário para utilizá-lo (WEISER, 1991). Assume-se que (i) é necessário existir um ambiente computacional invisível e transparente, com métodos intuitivos para que o usuário possa interagir com a computação sem precisar utilizar de conhecimentos da área, como ocorre hoje; (ii) o ambiente deve identificar quem (usuário) está inserido neste, e assim propor recursos para atendê-lo de forma personalizada.

Considerando a tendência da tecnologia, pode-se prever que, no futuro, estar-se-á sendo auxiliado por uma computação invisível e onipresente, onde todos os lugares estarão repletos de dispositivos de interação real-virtual dos mais variados tipos. É um ambiente reverso ao da Realidade Virtual (real é projetado no virtual), pois na Computação Ubíqua, o virtual é projetado no real, auxiliando as pessoas em seu dia-a-dia, com uma computação adaptada ao contexto do usuário.

Hoje, entretanto, é preciso se focar no desenvolvimento de um modelo, pois as tecnologias existentes não suprem as necessidades para o desenvolvimento integral destes espaços pervasivos (AUGUSTIN; LIMA; YAMIN, 2006).

Com a disponibilização de infra-estrutura de redes de sensores sem fio e de longa distância, vários sistemas e protótipos têm sido desenvolvidos por pesquisadores a fim de demonstrar como este novo paradigma pode beneficiar domínios de aplicações específicos,

como saúde e emergências (*smart hospital*), casa virtual (*smart home*), educação (*pervasive learning*), entretenimento e segurança de residências (*home security*) (AUGUSTIN; LIMA; YAMIN, 2006).

2.2 Computação Sensível ao Contexto

Conhecer os fatos que rodeiam o usuário da aplicação faz com que ela possa interagir e agir mais proveitosamente em prol deste. A aplicação deve ser ciente do contexto, adaptando-se automaticamente às mudanças no ambiente e às necessidades correntes do usuário, sem exigir sua atenção. Tais aplicações devem explorar características do ambiente como localização do usuário, pessoas próximas, hora do dia, etc... fornecendo informações adequadas à situação ou atividade.

Uma das principais áreas de pesquisa dentro da Computação Ubíqua é a Computação Ciente do Contexto (*context-aware computing*), a qual define uma área de pesquisa, relativamente recente, que possui aplicações em diferentes cenários computacionais e que apresenta desafios de implementação importantes.

2.2.1 Definição de Contexto

Ao iniciar uma pesquisa que envolva sistemas cientes de contexto torna-se muito comum se deparar com uma confusão do que é contexto para um determinado domínio, pois se é facilmente propenso a acreditar que tudo que existe é contexto; portanto, é necessário uma delimitação do que é contexto para uma determinada aplicação.

Dey (2006) define contexto como qualquer informação que pode ser usada para caracterizar a situação de uma entidade. Uma entidade é uma pessoa, lugar ou objeto que é considerado relevante para interação entre o usuário e a aplicação, incluindo, o próprio usuário e a aplicação. Dessa forma, toda informação somente é útil quando há relação entre ela e um contexto específico.

Um sistema é ciente do contexto se este usa o contexto para prover informações relevantes e/ou serviços para o usuário, onde relevante depende da tarefa do usuário (DEY;

ABOWD, 2006).

A definição de Dey é a mais utilizada na bibliografia por ser a mais abrangente e dinâmica, considerando aplicações, objetos e suas relações. Porém, a visão de Augustin (2006) para o contexto é a que mais se encaixa com esse trabalho, onde o contexto é definido como toda informação relevante para a aplicação que pode ser obtida por ela, podendo se referir a informações ambientais (recursos físicos), funcionais (recursos lógicos) ou comportamentais (perfil do usuário).

2.2.2 Modelagem de Contexto

Existe um crescente interesse no uso de sensibilidade ao contexto e técnicas de desenvolvimento de aplicações que sejam flexíveis, adaptáveis e capazes de agir automaticamente em nome do usuário (pró-atividade). A modelagem de contexto é um elemento central na construção desses sistemas. Ela define tipos, nomes, propriedades e atributos de todas as entidades que são relevantes para a aplicação.

O grau de refinamento e sensibilidade do modelo determina a "percepção do ambiente" pela aplicação. O modelo deve somente representar as entidades e relacionamentos relevantes.

Por outro lado, sensibilidade ao contexto introduz uma variedade de desafios à Engenharia de Software (HENRICKSEN; INDULSKA; 2006). O modelo de contexto deve suportar múltiplas representações do mesmo contexto, em diferentes formas e em diferentes níveis de abstrações, devendo também ser capaz de capturar o relacionamento existente entre representações alternativas (HENRICKSEN et al. 2002).

Contexto pode ser um tipo de representação do conhecimento. Assim, segundo Bettinia et al. (2010), é bastante natural a investigação de algum *framework* para representação do conhecimento e raciocínio que possam ser apropriados para manipulação de contexto. Esse *trade-off* (escolha) entre expressividade e complexidade de raciocínio tem sido mais dirigido para pesquisas na representação simbólica do contexto, nas últimas duas décadas.

2.2.3 Estratégias para Modelagem de Contexto

Pesquisas recentes no campo da consciência do contexto trazem uma abordagem predominantemente centrada em infra-estrutura (HENRICKSEN; INDULSKA; 2006), devido à complexidade na construção de aplicações sensíveis ao contexto. Através da utilização de infra-estrutura se é capaz de coletar, gerir e disseminar informações de contexto para as aplicações que necessitam. Estas pesquisas têm desenvolvido *frameworks* para integrar um conjunto de modelagens bem definidas de contexto e abstrações de programação com o suporte infra estrutural.

Nas propostas de modelagem é comum se encontrar uma abordagem baseada em objetos, na qual informações de contexto são estruturadas em torno de um conjunto de entidades, as quais descrevem fisicamente ou conceitualmente cada objeto, como pessoas e seus relacionamentos. Propriedades de contexto, tal como nomes de pessoas, são representados como atributos. Uma entidade é ligada a um atributo e a outras entidades como um relacionamento unidirecional, conhecido como associação (HENRICKSEN et al. 2002). Estas associações descrevem as dependências entre os elementos de contexto, construindo as relações que proverão informações agregadas.

2.2.4 Exigências na Modelagem de Contexto

A modelagem de contexto é um ponto importante para a geração de sistemas sensíveis ao contexto, devido ao fato de tudo estar vinculado ao contexto.

A pesquisa de Strang and C.L-Popien (2005) demonstra que Sistemas para Computação Ubíqua fazem grandes exigências, em qualquer abordagem de modelagem de contexto, em termos de:

- *Composição distribuída (dc)* - qualquer sistema de computação ubíqua é um derivado de um sistema de computação distribuída que carece de uma instância central a ser responsável pela criação, implantação e manutenção de dados e serviços, nas descrições de contexto. A composição e administração de um modelo de contexto e os seus dados variam de acordo com o elevado dinamismo em termos de tempo, topologia de rede e fonte;

- *Validação parcial (pv)* - é desejável ser capaz de validar o conhecimento parcial sobre a estrutura contextual. Isto é particularmente importante devido à complexidade das interrelações contextuais, o que torna qualquer intenção de modelagem propensa a erros;

- *Riqueza e qualidade de informação (qua)* - informações disponibilizadas por sensores variam ao longo do tempo, bem como a riqueza de informações fornecidas por diferentes tipos de sensores que caracterizam uma entidade em um ambiente de Computação Ubíqua. Assim, um modelo de contexto adequado para uso nessa área deve inerentemente dar suporte à qualidade e riqueza das informações;

- *Incompleto e ambíguo (inc)* - o conjunto de informações contextuais disponíveis em qualquer ponto do tempo que caracterizam as entidades competentes em ambientes de Computação Ubíqua é normalmente incompleto e/ou ambíguo; principalmente se essas informações são obtidas a partir de redes de sensores. Isto deve ser coberto pelo modelo, por exemplo, construído sobre um conjunto de dados incompletos em nível de instância;

- *Nível de formalismo (for)* - é sempre um desafio descrever fatos contextuais e relacionados de modo preciso e rastreável. Por exemplo, para executar a tarefa "documento impresso na impressora mais próximo de mim", requer-se o uso correto do termo usado para definição da tarefa, por exemplo o que "perto" significa para "mim". É altamente desejável que cada uma das partes na interseção do ambiente pervasivo tenha a mesma interpretação dos dados trocados e seu significado (chamado entendimento compartilhado);

- *Aplicabilidade para ambientes existentes (app)* - a partir da perspectiva de implementação, é importante que um modelo de contexto seja aplicável no âmbito atual de uma infra-estrutura de ambientes de Computação Ubíqua, por exemplo, um quadro de serviços, como Web Services.

2.2.5 Modelos de Contexto

Atualmente, existem alguns modelos propostos para representação de contexto, que são descritos a seguir.

Modelo Chave-Valor

O modelo de pares chave-valor é a mais simples estrutura de dados para modelagem de informações contextuais, é de fácil gerenciamento e programação, porém falta a capacidade de estruturar uma forma mais sofisticada de informação para permitir algoritmos de recuperação de contexto. Nesse modelo, normalmente, as informações de contexto são representadas apenas textualmente ou graficamente por tabelas

CHAVE	**VALOR**
Localização	Consultório
Atividade	Atendimento ao Paciente
Dispositivo	PC desktop

Tabela 1: Modelo Chave-Valor

Como ilustrado na tabela 1, podem ser associadas ao sistema ações que devam ser tomadas caso as informações de contexto informadas tenham determinados valores. Esta abordagem normalmente é fácil de programar, porém faltam recursos para a estruturação de algoritmos sofisticados que permitam a recuperação eficiente da informação de contexto.

Esquemas de Marcação

Comum a todas as abordagens de modelagem para esquema de marcação é uma estrutura de dados hierárquica, constituída por marcações com atributos e conteúdos. O conteúdo das *tags* de marcação é, geralmente, definido recursivamente por outras marcações.

Um exemplo deste tipo de abordagem é o padrão CC/PP (W3C, 2007), que descreve as capacidades do dispositivo e as preferências do usuário. Atualmente, o CC/PP está na versão 2.0, estruturada através do *Framework* de Descrição de Recursos (RDF), que é uma linguagem utilizada pela W3C para modelagem de metadados e descrição de documentos

XML (*eXtensible Markup Language)*, permitindo uma maior flexibilidade na criação de novos vocabulários.

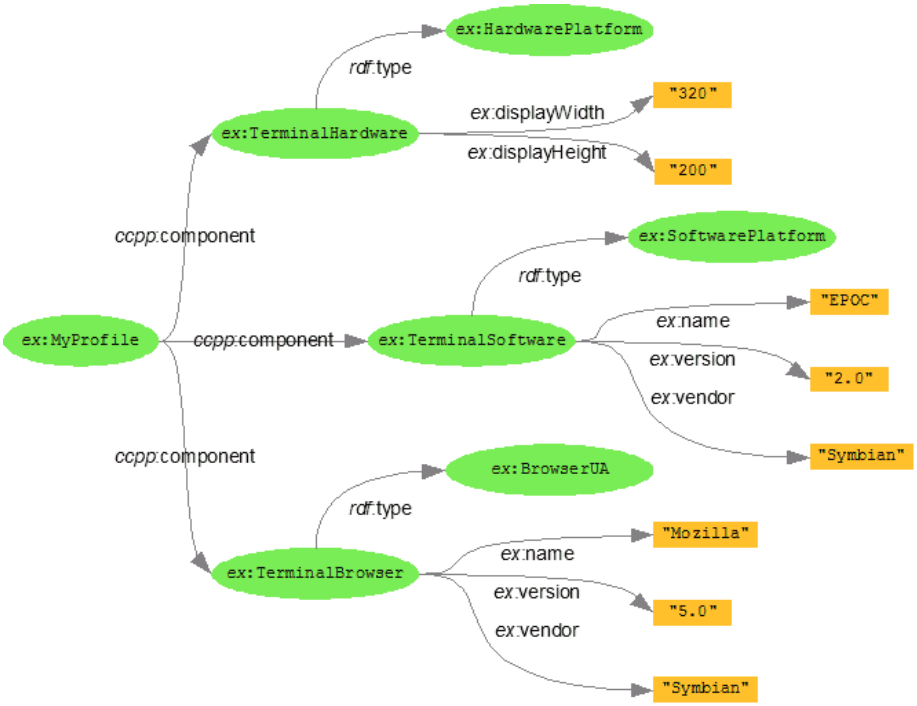

Figura 1:Exemplo de um perfil CC/PP (W3C, 2007)

Como mostra a figura 1, a descrição de cada componente é uma subárvore cujos galhos são as capacidades ou preferências associadas a esse componente. Embora RDF faça a modelagem de uma ampla gama de estruturas de dados possíveis, muitas vezes, estes podem ser descritos através de um pequeno número de CC/PP atributos, cada um tendo um valor atômico simples.

Modelos Gráficos

Modelos baseados em gráficos são úteis para realizar a análise do sistema, porém normalmente não são implementados. Essa categoria de modelos inclui modelos gráficos baseados em ORM (*Object Role Modeling*), grafos contextuais e em UML (*Unified Modeling Language*). A UML provê diferentes tipos de diagramas e também permite a separação dos conceitos, utilizando uma série de diagramas focados em diferentes aspectos do sistema.

Um exemplo do modelo gráfico baseado em UML é o CMP *(Context UML Profile*) (SIMONS, 2007), ilustrado na figura 2. Este exemplo representa o relacionamento entre a

entidade de contexto *Pearson* e as outras entidades do domínio, estereotipadas como *ContextItem.*

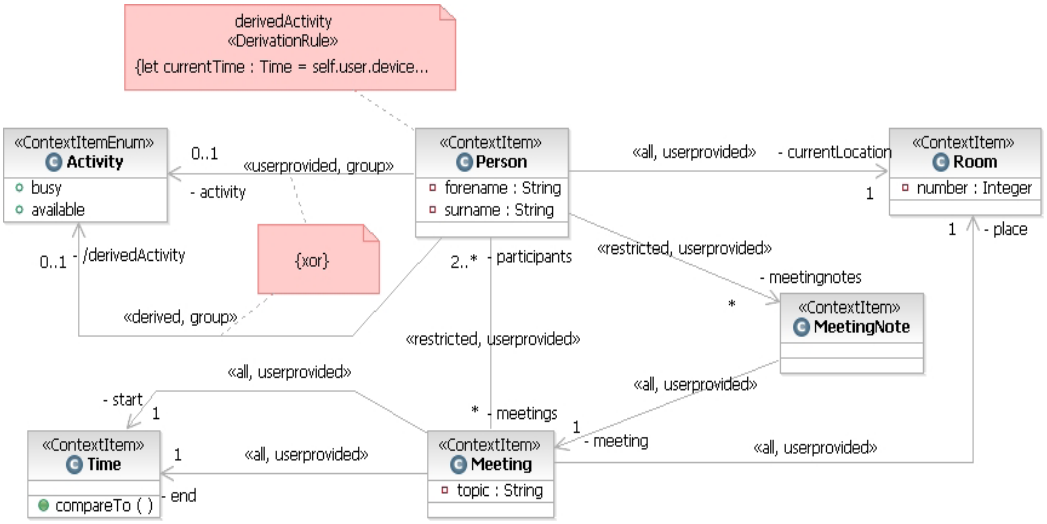

Figura 2: Sistema de reunião aplicado ao CMP (SIMONS, 2007)

Outro exemplo de modelagem gráfica de contexto (figura 3) é o *Context Modelling Language* (CML), proposto por Henricksen e Indulska (2006) para ajudar os projetistas na tarefa de explorar e especificar os requisitos do contexto de uma aplicação. Ele provê uma estrutura que permite anotar tipos de fatos, de acordo com cada entidade inserida no contexto, suportando uma variedade de restrições através de suas associações entre os relacionamentos.

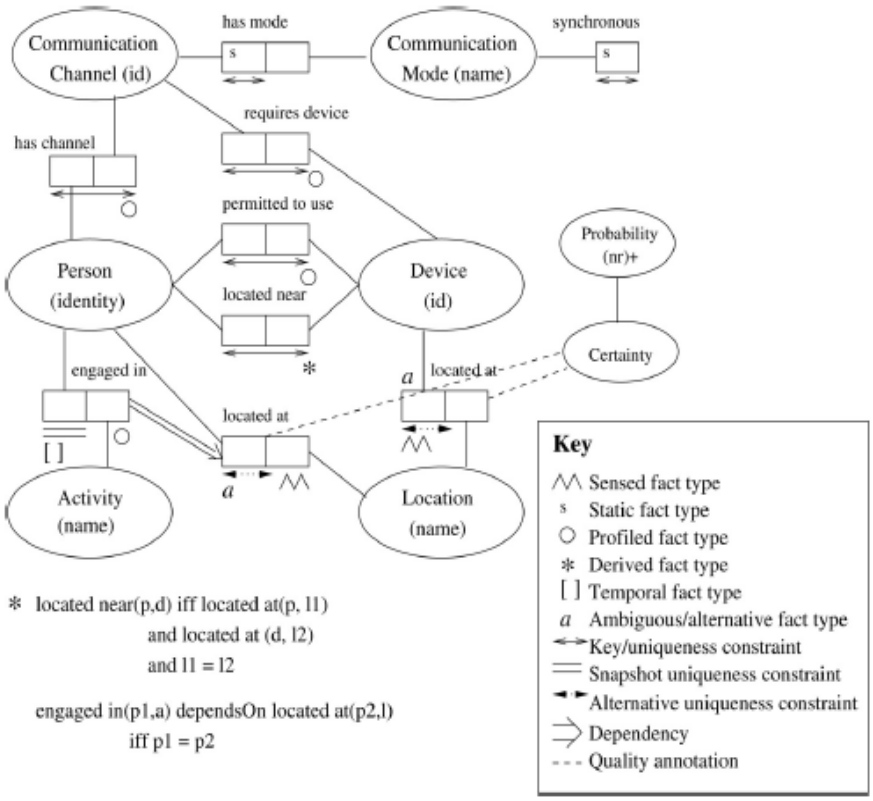

Figura 3: Exemplo do modelo CML (HENRICKSEN; INDULSKA; 2006)

Este modelo permite capturar atividades do usuário na forma de um tipo de realidade temporal, que abrange passado, presente e futuro das atividades.

Modelos baseados em Objetos

Modelos baseados em objetos buscam utilizar orientação a objetos para definir e estruturar as informações de contexto, tais como encapsulamento e reuso, através das especificações de herança, procurando cobrir os problemas decorrentes da dinâmica do contexto em ambientes ubíquos. Os detalhes do processamento do contexto são encapsulados em objetos, ficando ocultos para o restante dos componentes do sistema.

Um exemplo dessa abordagem é a pesquisa de Henricksen (2002), conforme exemplificado na figura 4. As informações de contexto são apresentadas por um conjunto de entidades que, por sua vez, descrevem objetos físicos ou conceituais, como um canal de comunicação ou uma pessoa. As identificações das entidades são expressas através de atributos, os quais apresentam as características de cada entidade.

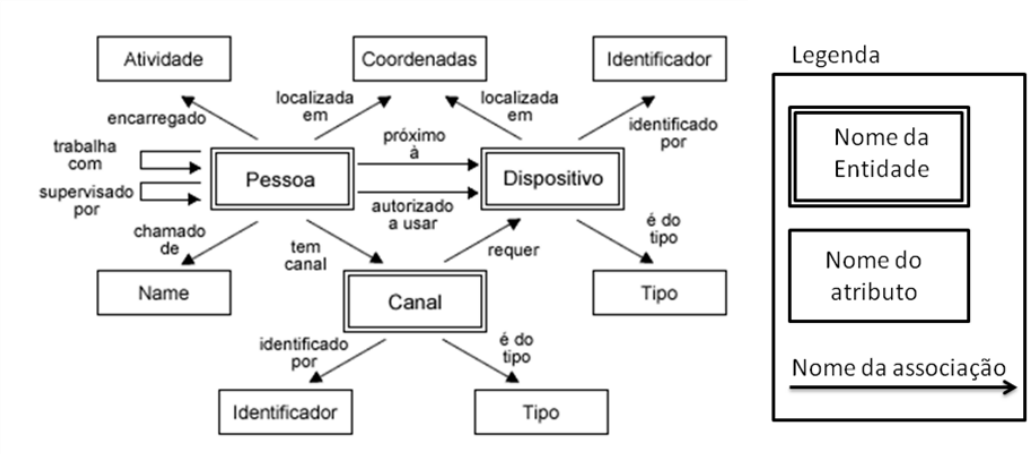

Figura 4: Modelo orientado a objeto adaptado (HENRICKSEN et al. 2002)

Os modelos orientados a objetos são fortes em relação à possibilidade de composição distribuída, pois novos tipos de informações de contexto podem ser adicionados e instâncias podendo ser atualizadas de forma distribuída (STRANG; POPIEN, 2005).

Modelos baseados em Ontologias

Informações de contexto baseados em ontologias exploram a representação e raciocínio lógico por múltiplos propósitos (BETTINIA et al. 2010): para descrever um dado de contexto complexo que não pode ser representado por linguagens simples; b) para prover uma semântica formal para o dado do contexto é possível tornar-se distribuída e/ou integrando contexto entre diferentes fontes; c) para avaliar o raciocínio, através de ferramentas que checam tanto a consistência quanto o conjunto de relacionamentos descrito em um cenário, e mais importante, reconhecer que um determinado conjunto de dados básicos de contexto e suas relações na verdade, revelam a presença de uma caracterização mais abstrata de contexto.

Segundo Bettinia et al. (2010), a escolha do formalismo na modelagem de informações de contexto baseados em ontologias é tipicamente OWL-DL ou algumas de suas variações, uma vez que ela está se tornado um padrão para vários domínios de aplicação e é suportada por um grande número de servidores de raciocínio. OWL-DL possibilita uma modelagem particular de um domínio, definindo classes, indivíduos, características de um indivíduo (propriedades *datatype*), e relacionamentos entre indivíduos (propriedades dos objetos).

Várias linguagens OWL têm sido propostas para representar a descrição distribuída para os dados do contexto. Entre a mais promissora proposta está SOUPA (CHEN et. al, 2004) para modelagem de contexto em ambientes pervasivos, e CONON (ZHANG et. al, 2005),

ontologias para ambientes de casas inteligentes. Estas ontologias distribuídas podem ser integradas com modelos específicos de aplicação de contexto por meio de extensões da linguagem OWL, tal como proposta por BOUQUET (2004).

Comparações entre os modelos

Strang and C.L-Popien (2005) realizaram uma pesquisa comparativa entre os modelos. A tabela 2 mostra este comparativo através dos requisitos levantados para uma abordagem de contexto.

Abordagens requisitadas	*dc*	*pv*	*qua*	*inc*	*for*	*app*
Chave Valor	-	-	-	-	-	*
Esquema Marcação	*	*	-	-	*	*
Orientado a Objetos	*	*	*	*	*	*
Baseado em Lógica	*	-	-	-	*	-
Gráfico	-	-	*	-	*	*
Baseado em Ontologias	*	*	*	*	*	*

Tabela 2: Comparativo entre os modelos (STRANG and C.L-POPIEN, 2005)

Como demonstrado por Strang and C.L-Popien (2005), a modelagem baseada em ontologias atingiu a maior satisfação dentre os requisitos elencados. Nele, utilizam-se as principais formas de modelar o contexto, as quais contemplam o maior número de requisitos.

Utilizando os resultados dessa pesquisa, foi realizada a modelagem de contexto deste trabalho (ontológico e gráfico) e estruturado a camada de software para contextualização dos dados brutos como informações de contexto para uma tarefa clínica (capítulo 4). Assume-se que a junção das melhores formas de modelar o contexto apresenta um resultado mais satisfatório e robusto.

Não existe um consenso para a utilização de um dos modelos acima citados que seja plenamente rico na sua utilização. Algumas pesquisas descrevem propostas de modelagem de contexto genéricas para vinculação a sistemas ubíquos. Porém, as abordagens propostas não suprem todas as necessidades existentes em um ambiente clínico.

2.3 Projetos de Pesquisa em Computação Ubíqua na Saúde

O desenvolvimento da Computação Ubíqua e Pervasiva está em uma escala crescente, sendo alvo de diversas pesquisas, incluindo seu emprego na área clínica, chamado de UbiHealth ou pHealth.

2.3.1 Temas em estudo na UbiHealth

Existe um interesse crescente de pesquisas direcionado a soluções de assistência domiciliar. Várias propostas de pesquisa em *HomeCare* sugerem o desenvolvimento de soluções capazes de registrar e analisar a mobilidade dos usuários, detectar sinais vitais e avisar aos clínicos, assistir aos pacientes com necessidades especiais em suas atividades diárias com uso de novos dispositivos. Protótipos de pesquisas disponíveis em residências permitem o acompanhamento constante das condições dos pacientes, os quais muitas vezes integram mecanismos de alertas para fornecer respostas rápidas para situações de emergências.

Um desses projetos é o Continua Health Alliance (WARTENA, 2009), dedicado ao estabelecimento de um sistema de saúde pessoal, interoperável, com soluções para ampliar a assistência domiciliar, favorecendo independência na gestão de saúde e bem estar pessoal. Tem como objetivos desenvolver diretrizes para prover interoperabilidade na construção de sensores, redes domésticas e serviços de saúde. Tal projeto trabalha com líderes das indústrias de assistência médica, a fim de desenvolver novas formas para diminuir custos no fornecimento de sistemas em *TeleHealth*. Atualmente, conta com mais de 200 empresas associadas em todo o mundo, destacando-se: GE Healthcare, Cisco, IBM, Intel, Nokia, Panasonic, entre outras.

O projeto MATCH - *Mobilising Advanced Technologies for Care at Home* (TURNER et al, 2010) aborda um apoio automatizado em rede para desenvolvimento de cuidados clínicos domiciliares (*home care networks*). O projeto descreve dois componentes chaves para uma proposta de arquitetura: (i) um serviço de registro, que suporta registros genéricos e extensíveis, componentes, dispositivos e recursos domiciliares, e utiliza ontologias para descrição e descoberta baseadas em semântica; (ii) um sistema de políticas, automatizando o

apoio de como uma rede doméstica deve prestar serviços de saúde.

Existe um ramo de pesquisa voltado ao tratamento de doenças específicas dentro da *HomeCare*. Conforme discussões realizadas no workshop UbiHealth do UbiComp (2010), a maioria das pesquisas tem focado em tecnologias ubíquas para desordens somáticas, como diabetes, hipertensão, obesidade. Isto foi evidenciado desde os primeiros workshops, ainda que outras pesquisas tenham se dirigido à demência e autismo, com foco menor em outras desordens mentais como depressão, ansiedade e manias. Essas desordens são universais, afetam todas as regiões do planeta, e torna-se importante contribuir para aliviar o sofrimento do paciente.

Algumas pesquisas estão relacionando Computação Afetiva, *HomeCare* e Computação Ubíqua. Embora em fase inicial, algumas pesquisas já apresentam resultados promissores sobre o assunto (UBIHEALTH, 2010). O desenvolvimento de aplicações de saúde para gerir contexto emocional é bastante exigente, o que requer uma infra-estrutura capaz de detectar estados emocionais dos pacientes. Para tal, deve-se levar em conta informações do contexto, como os usuários reagem a situações, bem como suas condições físicas. Com esse objetivo, as propostas de *middleware* devem fornecer suporte integrado para modelagem de contexto, aquisição e raciocínio (TALEB et al, 2010).

Taleb et al (2010) descrevem o PEACH *Framework* , uma solução de *middleware* sensíveis ao contexto, que promove e apoia o desenvolvimento de aplicações de saúde emocional para ambientes de Computação Pervasiva, procurando suprir, em qualquer lugar e a qualquer momento, os serviços de saúde para os indivíduos monitorados (usuários de drogas). O PEACH *Framework* provê um conjunto básico de facilidades para integrar biossensores capazes de monitorar condições psico-físicas do paciente, agregando dados sensorados para detectar situações onde a assistência do paciente é necessária, e para compor e gerir grupos voluntários dispostos a ajudar o paciente em situações de emergência. O PEACH reconhece a necessidade de considerar diferentes funções de gestão de serviços em saúde emocional. No PEACH, cada paciente porta um dispositivo móvel e, assim, pode ser identificado e monitorado. Além disso, as condições do paciente são monitoradas através da exploração de diferentes entidades de sensores, em caso de situação de emergência detectadas por alterações emocionais um centro de vigilância é alertado e este aciona um grupo de pessoas, composto por voluntários dispostos a ajudar o paciente.

Estas pesquisas estão utilizando redes de assistência domiciliares (*HomeCare*), buscando monitoramento, através da aplicação das tecnologias da Computação Ubíqua,

voltadas para pacientes idosos que, normalmente, são tratados à domicilio, como é o caso dos portadores de doenças demências, como Alzheimer.

2.3.2 Sistemas Pervasivos Pró-Ativos

A área de *smart hospital* é a que mais influenciou o projeto ClinicSpace, entre os projetos relacionados estão o *Activity-Based Computing*, Task Computing, Aura e Gaia, os quais são descritos nesta seção.

2.3.2.1 Projeto Activity-Based Computing (ABC)

O Projeto *Activity-Based Computing* (BARDRAM; CHRISTENSEN, 2007) apresenta uma proposta para utilização de Computação Baseada em Tarefas destinadas aos Ambientes de Saúde, e influenciou muitas das decisões tomadas no desenvolvimento da arquitetura ClinicSpace. O framework ABC foi desenvolvido para ser uma plataforma de programação onde as aplicações orientadas às atividades poderiam executar. O framework lida com a complexidade computacional de gerenciar atividades distribuídas e colaborativas, adaptando-as aos serviços e recursos existentes no ambiente.

O projeto ABC define a computação baseado em atividades sob os aspectos:

(i) *Centrado na Atividade* - a atividade computacional abrange um conjunto coerente de serviços e dados necessários para apoiar o usuário na realização de algum tipo de trabalho (atividade). Por exemplo, a atividade colaborativa de fazer um diagnóstico, para um paciente em um hospital, pode ser modelada no ABC como uma atividade computacional, que inclui serviços para exibir e manipular os dados do paciente, buscar os resultados de testes de análise de sangue, mostrar as imagens recentes de raio X;

(ii) *Ciclo de vida de uma Atividade* - um usuário participa de várias atividades e pode alternar entre elas, suspendendo e retomando uma atividade. Este principio ajuda os usuários a lidarem com as interrupções;

(iii) *Armazém de Atividades* - uma atividade é armazenada em uma infra-estrutura (por exemplo, um servidor) e pode ser distribuída em uma rede. Assim, uma

atividade pode ser suspensa em uma estação de trabalho e continuar em outro lugar. Este princípio apóia a mobilidade do usuário;

(iv) *Adaptação da Atividade* - uma atividade se adapta aos recursos disponíveis no dispositivo (computador, por exemplo) em que é retomada. Tais recursos são, por exemplo, a largura de banda de rede, CPU, ou dispositivos de exibição em um dado;

(v) *Colaboração entre Atividades* - uma atividade é compartilhada entre os usuários de colaboração. Tem uma lista de participantes que podem acessar e manipular a atividade. Por conseguinte, todos os participantes de uma atividade podem retomá-la e continuar o trabalho de outro usuário. Além disso, se dois ou mais usuários retomam a mesma atividade, ao mesmo tempo, em diferentes dispositivos, eles serão notificados em seus dispositivos de apoio, iniciando uma conferência em tempo real. Este princípio permite que os usuários colaborem diretamente dentro de uma atividade;

(vi) *Contexto de Atividade* - uma atividade é ciente de contexto, isto é, capaz de se adaptar e se ajustar de acordo com seu contexto de uso. *Context-awareness* pode ser usado para adaptar a interface do usuário de acordo com a situação atual de trabalho do usuário - por exemplo, mostrando dados médicos para o paciente a ser tratado, ou pode ser usado em um sentido mais técnico, onde a execução de uma atividade, e correspondente descoberta de serviços, é ajustada aos recursos disponíveis em sua proximidade. Este princípio permite aos usuários criar atividades que são adaptadas ao contexto de trabalho atual.

A abordagem de contexto utilizada no projeto ABC é focada na questão da descoberta das atividades do usuário, baseada no contexto em que ele se encontra e em modelos de atividades previamente conhecidos. ABC consiste em um sistema abstrato construído sobre arquiteturas que promovam a capacidade de captação e programação de aplicações conscientes de contexto, no caso, o JCAF (BARDRAM, 2003).

Nesse projeto ainda é disponibilizado um modelo de programação para o desenvolvimento de serviços e aplicações, mas a utilização desse modelo requer

conhecimentos de programação que, geralmente, apenas profissionais de computação possuem. Assim, a programação de atividades não é uma tarefa simples e acessível aos profissionais clínicos. Além disso, não são providos mecanismos para permitir que os profissionais personalizem as atividades, inserindo nelas a sua forma particular de executá-las.

2.3.2.2 Projeto Task Computing

O projeto *Task Computing*, pertencente à empresa Fujitsu, é um novo paradigma para que usuários se concentrem nas tarefas que desejam realizar com dispositivos computacionais, em vez de como realizá-las. Diminui a lacuna entre o que os usuários realmente querem fazer e os recursos dos dispositivos e/ou serviços que possam estar disponíveis em seus ambientes. Através do uso de um *Client Task Computing* (TCC), usuários não só podem compor as tarefas de serviços disponíveis semanticamente descritas, mas descobrir, criar, gerenciar e manipular os serviços (SONG; LABROU; MASUOKA, 2004).

"Task Computing Environment (TCE)" é uma *framework* que suporta tarefas computacionais, provendo suporte para fluxos de trabalho, descreve serviços semanticamente e gerencia serviços para usuários finais. TCE proporciona um ambiente de programação para pesquisadores e desenvolvedores, através de componentes acessíveis por Web Services.

A arquitetura é composta por quatro camadas distintas, como mostra a figura 5:

(i) camada de realização, abrangendo o universo dos dispositivos, aplicações e E-service;

(ii) camada de serviços contendo funcionalidades disponibilizadas como serviços, os quais contém uma descrição semântica associada;

(iii) camada de middleware, responsável pela descoberta, execução, gerenciamento, monitoramento, criação e publicação dos serviços e tarefas;

(iv) camada de apresentação, busca prover a abstração da complexidade da tarefa para o usuário, apresentando um ambiente com funcionalidades dinâmicas para a execução das tarefas pelo usuário.

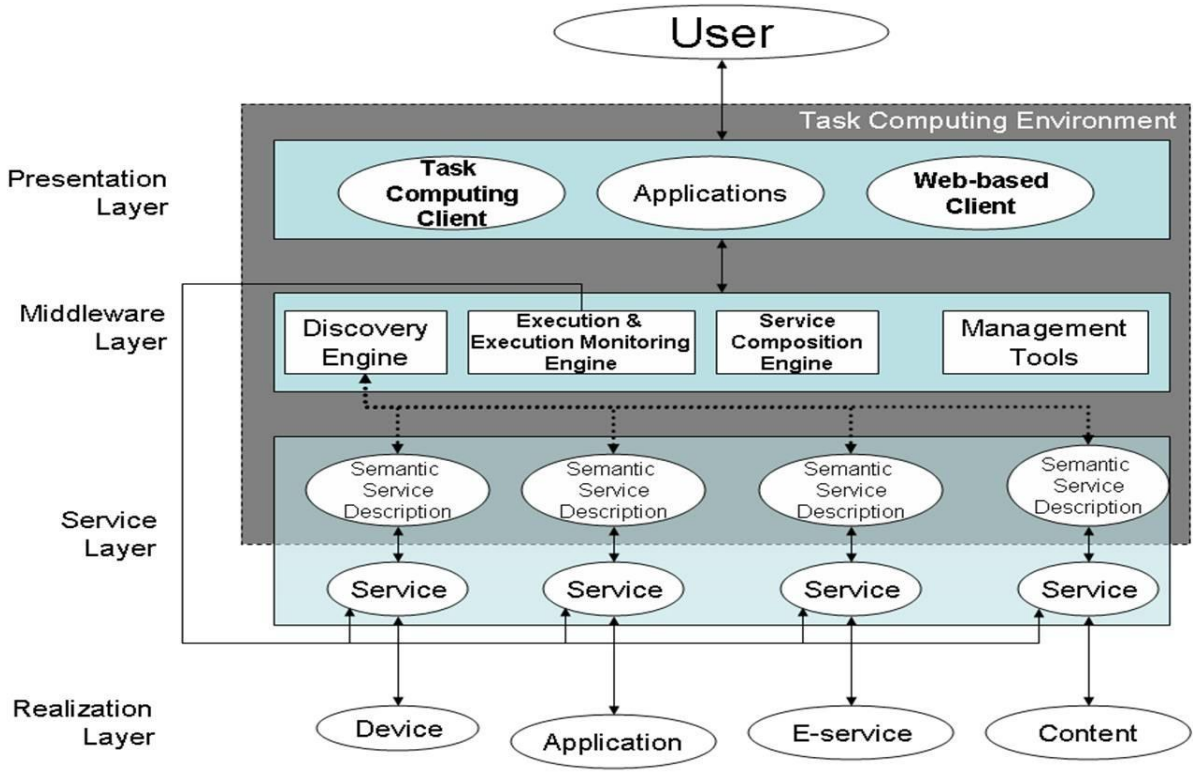

Figura 5: Arquitetura do ambiente Task Computing. Fonte: (SONG; LABROU; MASUOKA, 2004).

Task Computing é um ambiente onde o usuário pode executar tarefas que não foram (nem implicitamente, nem explicitamente) projetadas para o sistema, multiplicando, assim, os usuários das fontes de funcionalidade (dispositivos, aplicações, conteúdos e serviços de e-mail).

A separação semântica da descrição do serviço e da implementação do serviço torna possível para os usuários incorporar os serviços Web existentes no *framework*, fornecendo descrições semânticas para eles.

Segundo SONG, LABROU; MASUOCA (2004), o *Task Computing* é um framework muito útil e poderoso para ambientes como hospitais, escritórios e casas, onde o usuário final pode integrar e manipular facilmente funcionalidades computacionais, através de dispositivos em torno dele e serviços remotos através da web, permitindo-lhe facilmente definir, executar e controlar tarefas complexas. O foco principal está na infra-estrutura de suporte a execução de serviços, pois é baseada em informações de contexto para a definição dos serviços mais apropriados. Embora exista uma sinalização de possível utilização em ambientes clínicos, o projeto não apresenta uma modelagem de tarefas capaz de suportar a personalização pelo usuário.

2.3.2.3 Projeto Gaia

O projeto Gaia (www.cs.uiuc.edu/gaia) visualiza um futuro no qual os espaços habitados pelas pessoas são interativos e programáveis. Os usuários interagem com seus escritórios, casas, carros etc. Podem configurar o comportamento do habitat, assim se beneficiar a partir dos recursos disponíveis. Dados e tarefas estão sempre acessíveis e são mapeados dinamicamente para os recursos convenientes e presentes na localização corrente pelo sistema GaiaOS, assim podendo estender o seu espaço habitado para dispositivos pessoais que se integram ao ambiente.

Este ambiente interativo, centrado no usuário, exige uma infra-estrutura de software para operar com os recursos, observar propriedades do contexto, assistir ao desenvolvimento e execução das aplicações. Assim, o *Middleware* Gaia objetiva o gerenciamento de recursos e fornece ao usuário interfaces orientadas ao espaço físico. A pesquisa limita o espaço físico para o espaço usado pelos professores: salas de aula, escritórios e salas de leitura.

O projeto Gaia busca construir uma infra-estrutura de software para auxiliar no desenvolvimento de aplicações para ambientes pervasivos. Neste ambiente, usuários interagem com um número de dispositivos simultâneos, registrando seus dispositivos aos recursos do ambiente, buscando adaptação das aplicações de acordo com as mudanças no contexto do ambiente.

Segundo ROMAN (2002) os ambientes computacionais ubíquos são referenciados como espaços ativos, uma extensão para espaços físicos.

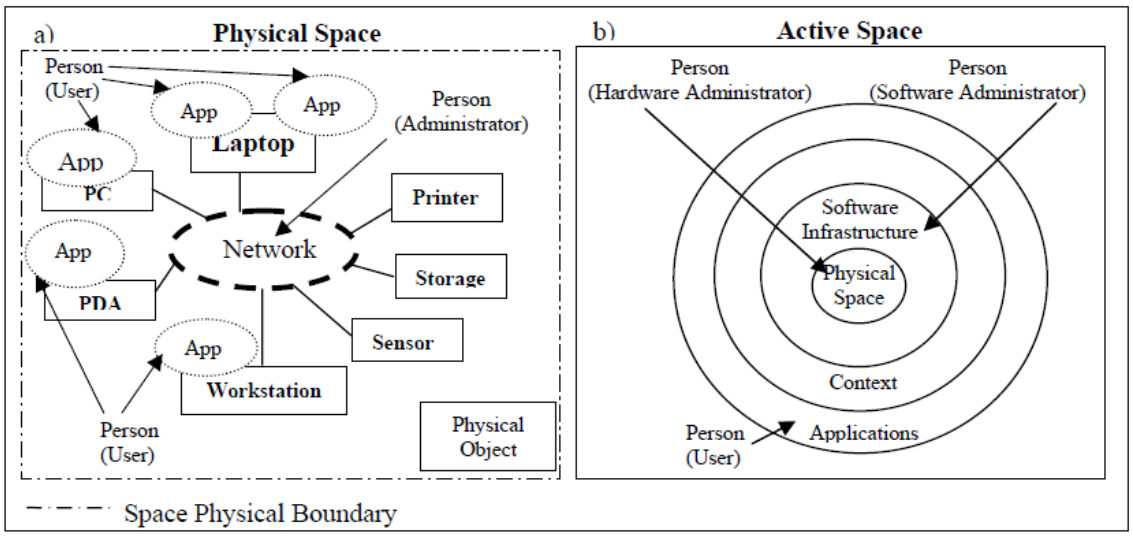

Figura 6: Espaço Físico e Ativo (ROMAN, 2002)

Como ilustrado na figura 6 (a), espaços físicos são regiões geográficas limitadas, com fronteiras físicas bem definidas, contendo objetos físicos, dispositivos de rede heterogêneos e usuários com atividades variadas. A definição de espaço ativo (figura 6 (b)) é um espaço físico gerenciado por uma infra-estrutura de software baseada na sensibilidade ao contexto. Permite a mobilidade do usuário para interação e configuração de seu ambiente físico. O requisito para o espaço ativo é o desenvolvimento de aplicações móveis com suporte a execução de aplicações centradas no usuário (ROMAN, 2002).

Para tal, utiliza-se o GaiaOS, um meta-sistema operacional que dá suporte à execução de aplicações portáteis em espaços ativos. Gaia coordena as entidades (aplicação, serviço do dispositivo, serviço para pessoas) de software e dispositivos em rede dentro dos espaços ativos, exporta serviços para pesquisar e utilizar recursos, acessar e usar o contexto corrente, e também fornece um *framework* para desenvolver aplicações.

O GaiaOS fornece serviços de localização, contexto e repositórios de serviços. O sistema é construído como um sistema de objetos distribuídos.

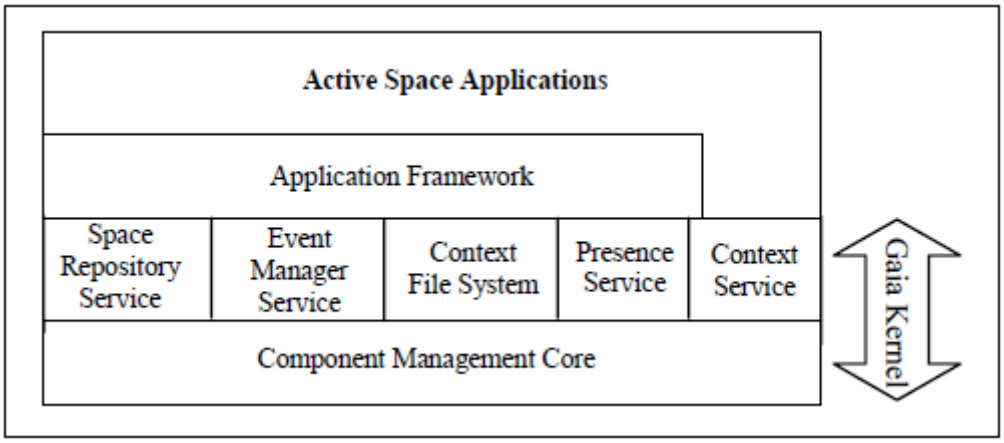

Figura 7: Arquitetura GAIA (ROMAN, 2002)

A figura 7 mostra as três camadas do GaiaOS: Gaia Kernel, o Gaia *Framework* de Aplicação e Aplicação.

O Kernel contém cinco serviços: gerenciador de eventos, serviço de contexto, serviço de presença, repositório do sistema de arquivo e contexto. O gerente de eventos, implementado usando CORBA, fornece canais de eventos. Todos os eventos são compartilhados com outros dispositivos, utilizando os canais de eventos.

O Serviço de Contexto fornece informações de contexto sobre todas as entidades do

espaço ativo. Por exemplo: contexto para a localização de pessoas, as condições dentro de uma sala, as condições meteorológicas, etc. A arquitetura de contexto utiliza uma abordagem baseada em predicados lógicos de primeira ordem (RANGANATHAN, 2005), onde as condições de contexto são dadas por *TipoContexto(<sujeito>, <verbo>, <objeto>). Por exemplo*: *Context(temperature, room 3231, is, 98F)* (ROMAN et al, 2003).

Serviço de Presença mantém informações sobre os recursos presentes no espaço ativo, usa um mecanismo de balizamento, onde cada entidade transmite um sinal de presença em intervalos regulares. Se a entidade não transmitir um sinal, Gaia pressupõe que a entidade deixou o espaço (ROMAN et al, 2003).

O projeto Gaia destina-se a ambientes pervasivos como casas e escritórios, sua noção de tarefas objetiva a adaptação de contexto visando suportar as atividades do usuário. Programar uma tarefa é possível através da Linguagem de Programação Olympus, e assim, requer profissional especialista. Assim, a proposta defendida no presente trabalho de permitir aos usuários comuns programarem suas tarefas e personalizá-las adequando-as a sua forma particular de executar, não é possível no GAIA.

2.3.2.4 Projeto Aura

O projeto Aura (www.cs.cmu.edu/~aura), desenvolvido na Carnegie Mellon University, busca solucionar o problema existente de o usuário ter que gerenciar os recursos envolvidos no ambiente e as mudanças dinâmicas e freqüentes para utilização da computação. O objetivo é proporcionar a cada usuário um conjunto invisível de serviços de computação, que persiste independentemente da sua localização.

O Aura consiste em uma arquitetura de apoio às necessidades computacionais da mobilidade do usuário, satisfazendo dois objetivos concorrentes: maximizar a utilização dos recursos disponíveis e minimizar a distração do usuário com problemas para gerenciar os recursos do ambiente.

Para tanto, cunhou o conceito de “aura de informação pessoal” que se espalha pelas diversas infra-estruturas computacionais. O intuito por trás de uma aura pessoal é que ela atua como um *proxy* para a mobilidade do usuário, representando-o em cada ambiente ao qual este se desloca. Quando um usuário está em um novo ambiente, sua aura absorve os recursos adequados para apoiar suas tarefas. Além disso, uma aura captura as restrições que o contexto

físico ao redor do usuário impõe às suas tarefas (SOUSA; GARLAN, 2002).

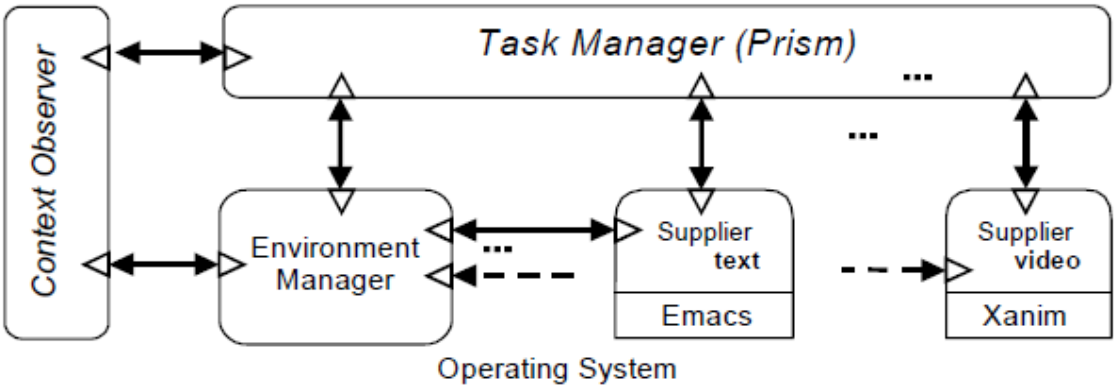

Figura 8: Arquitetura AURA (SOUSA; GARLAN, 2002)

A figura 8 ilustra os quatro tipos de componentes da arquitetura: (i) Gerenciador de Tarefa (*Task Manager*) que incorpora o conceito de aura pessoal, (ii) Observador de Contexto (*Context Observer*) que provê informações de contexto físico e reporta eventos relevantes do contexto ao Gerenciador de Tarefas e ao Gerenciador de Ambiente, (iii) Gerenciador de Ambiente (*Enviroment Manager*) que consiste em um *gateway* do ambiente, realizando o mapeamento necessário para que o Gerenciador de Tarefas encontre os serviços correspondentes às suas necessidades no ambiente pervasivo, (iv) Provedores de Serviço (*Services Supplier*) que disponibilizam os componentes fornecedores de serviço para a arquitetura, incluindo as ferramentas e aplicações disponíveis no ambiente (editores de texto, tocadores de musica, etc...) (SOUSA; GARLAN, 2002).

3 PROJETO CLINICSPACE

Este capítulo apresenta o projeto ClinicSpace, sua origem e os trabalhos realizados pelos mestres FERREIRA (2009), SILVA (2009), RIZZETTI (2009), VICENTINE (2010) e SOUZA (2010) na definição e implementação da arquitetura.

3.1 Motivação e Problemas Abordados

Atuais Sistemas Eletrônicos de Saúde (EHS) são modelados e construídos com a visão corporativa hospitalar, a qual busca prover um gerenciamento em termos de negócios. Pesquisas demonstram (BARDRAM, 2005) que existe uma grande rejeição por parte dos clínicos aos EHS devido a essa estruturação de negócio vinculado aos sistemas de saúde. Este boicote ao sistema traz inúmeros problemas à realização das atividades diárias, como por exemplo, a falta ou minimização de informações históricas de consultas dos pacientes, prejudicando diagnósticos que necessitam de maiores informações históricas para serem providos.

A construção de EHS com a visão centrada no usuário clínico (médico) está se tornando requisito para diminuição da rejeição atual aos sistemas de saúde. Devido às características dinâmicas encontradas na área clínica, sua construção se torna um desafio.

Algumas características do meio clínico são mobilidade e trabalho colaborativo. No cotidiano de um médico, as atividades podem sofrer interrupções, para realização de atividades emergenciais, e o retorno a ela quando possível. A atividade pode ser delegada a outro profissional para sua conclusão (BARDRAM; CHRISTENSEN, 2007). Profissionais precisam deslocar-se pela instituição, e os sistemas devem garantir ao profissional rápido acesso aos dados do paciente, em qualquer ponto do hospital. Logo, o EHS deve ter uma semântica ‘siga-me’ (AUGUSTIN, 2005), deslocar-se junto com o médico (migração) e adaptar-se às situações onde se encontra (contexto), além de necessitar prover informações históricas dos pacientes implementadas em uma base de dados pervasiva: *always on, anytime, anywhere, anydevice, anynetwork.*

As características encontradas nos ambientes clínicos podem ser gerenciadas por sistemas ubíquos (BARDRAM, 2003). Neste sentido, o projeto ClinicSpace busca prover uma infra-estrutura computacional para auxiliar o médico em suas atividades diárias, permitindo que este expresse a forma individual que realiza uma determinada atividade (personalização). Com isso, não é necessário conhecimentos sobre sistemas gerados por analistas, que podem modificar o fluxo de trabalho dos usuários pela falta ou má definição dos requisitos do sistema.

Devido à complexidade computacional necessária para suprir as características do ambiente clínico, o projeto procura realizar rotinas pró-ativas como a entrada de dados via captura de contexto, minimizando a complexidade para o usuário clínico.

Sendo assim, o projeto ClinicSpace usa conceitos da Computação Ubíqua para gerenciamento e adaptabilidade dos sistemas ao meio clínico, buscando equilibrar a pró-atividade (agir em nome do usuário) com a personalização (forma individual de cada usuário realizar uma atividade), visando diminuir a rejeição encontrada nos sistemas EHS atuais.

3.2 Tarefas e seu Funcionamento

Procurando prover o máximo de integração com o usuário, a arquitetura ClinicSpace é projetada através da concepção da teoria da atividade (RANGANATHAN; CAMPBELL 2005). O termo *atividade* refere-se ao processo realizado por humanos de forma colaborativa, coordenada, distribuída, em um espaço determinado. A Teoria da Atividade humana modela-a com seis componentes: sujeito, objeto, ferramentas, regras, comunidade e colaboração.

Aplicando ao ambiente clínico o modelo genérico proposto pela teoria da atividade tem-se o médico atuando como sujeito, que tem por objetivo diagnosticar e/ou tratar o paciente que, por sua vez, atua como objeto da atividade. Para realizar a atividade, o médico utiliza ferramentas de mediação, as quais consistem em registros, procedimentos, equipamentos e recursos. Essa mediação, geralmente, segue regras especificadas através de guias clínicos. A atividade é executada no ambiente clínico, sendo esta a sua comunidade, onde há uma diversidade de profissionais que, normalmente, fazem uma divisão do trabalho para a sua realização.

Ao modelar essas noções na arquitetura ClinicSpace, a atividade clínica foi decomposta em um conjunto de tarefas – ações que tem o auxílio de recursos computacionais, e seguem a

forma particular de cada indivíduo de realizá-la (personalização). Tarefas simples são compostas por subtarefas (operações) e, quando agrupadas, formam uma tarefa composta que segue um fluxo de execução (workflow) (FERREIRA, 2009a). Assim, as tarefas podem ser modeladas usando o conceito de composição, tendo como ponto de partida as subtarefas – aplicações disponibilizadas pela arquitetura ClinicSpace (SILVA, 2009).

As subtarefas foram categorizadas como: (i) *identificação*, subtarefas que permitem identificação automática de profissionais e pacientes; (ii) *busca*, subtarefas responsáveis por recuperar informações de profissionais e pacientes; (iii) *preenchimento*, subtarefas responsáveis pelo registro de informações dos pacientes no pEHS; (iv) *visualização*, subtarefas que permitem a visualização das informações dos pacientes. (SILVA, 2009a).

O usuário interage com o sistema, personaliza suas tarefas, as quais são armazenadas com o vínculo a esse usuário. Assim, cada médico tem um conjunto de tarefas vinculadas ao seu perfil. Estas tarefas inicialmente são projetadas a partir de um conjunto de 11 tarefas básicas (subtarefas), disponibilizadas como ícones na Interface de Edição de Tarefas (IET[1]) (SILVA, 2009b), determinadas a partir do estudo de Laerum and Faxvaag (2004). A Tabela 3 apresenta as tarefas mínimas (subtarefas).

1	Revisar os problemas do paciente
2	Procurar informações específicas em registros do paciente
3	Obter os resultados de novos testes ou investigações
4	Adicionar notas diárias sobre as condições do paciente
5	Requisitar análises clínicas em laboratórios bioquímicos
6	Obter resultados de exames clínicos
7	Requisitar raios-X, ultra-som e tomografias computadorizadas
8	Obter resultados de raios-X, ultra-som e tomografias computadorizadas
9	Requisitar tratamentos
10	Escrever Prescrições
11	Registrar códigos para diagnósticos ou procedimentos executados

Tabela 3: Tarefas Mínimas (subtarefas)

[1] A interface de edição inicialmente foi proposta pelo trabalho de SILVA (2009) e contemplava a modelagem das tarefas, somente nesta dissertação, os elementos de contexto foram vinculados a interface, assim a tornado uma Interface de Edição de Tarefas e Contexto (IETC).

As tarefas são modeladas como elementos dinâmicos e de definição recursiva, que podem conter (i) subtarefas ou (ii) uma tarefa inteira composta por subtarefas, e contida em outra tarefa de propósito mais abrangente, vindo assim a formar uma representação em grafo, onde um nó pode conter subtarefas embutidas, como ilustra a figura 9.

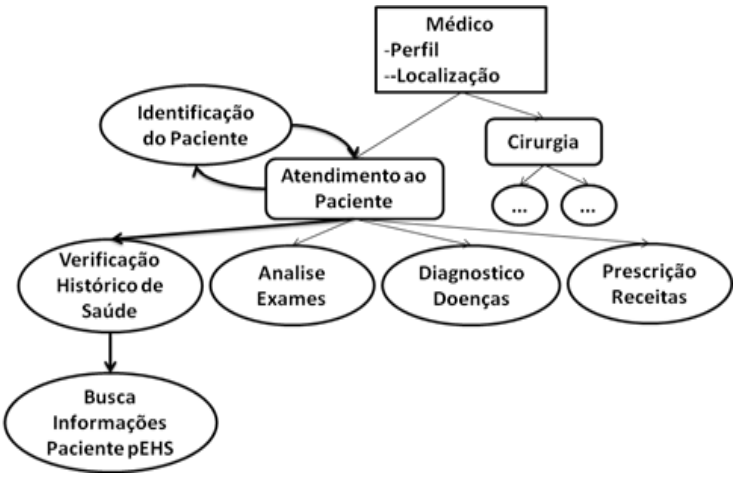

Figura 9: Representação-exemplo do relacionamento entre as tarefas e subtarefas

Quanto mais alto nível for a tarefa, menos recursos físicos ela utiliza, pois somente as subtarefas têm código computacional a ser executado. As subtarefas são os nós finais do grafo. Por exemplo, na figura 9 a tarefa *Atendimento ao Paciente* faz o sincronismo necessário entre as subtarefas (*Identificação do Paciente* e *Verificação de Histórico de Saúde)*. É a tarefa que contém as informações necessárias para iniciar suas subtarefas. Deste modo, a subtarefa de *Verificação de Histórico de Saúde* não pode ser iniciada antes que a subtarefa de *Identificação do Paciente* retorne com o paciente que o médico deseja; desta forma, sabendo qual o paciente que está sendo atendido, é possível iniciar as outras subtarefas (*Analise de Exames*, *Diagnostico de Doenças*, *Prescrição de Receita*).

3.2.1 Ciclo de vida e atributos das Tarefas

Para que a execução computacional das tarefas possa ser gerenciada, foi incorporado um ciclo de vida às tarefas (Figura 10), as quais são gerenciadas pelo Serviço Gerenciador de Tarefas (SGT) (ver seção 3.3.3).

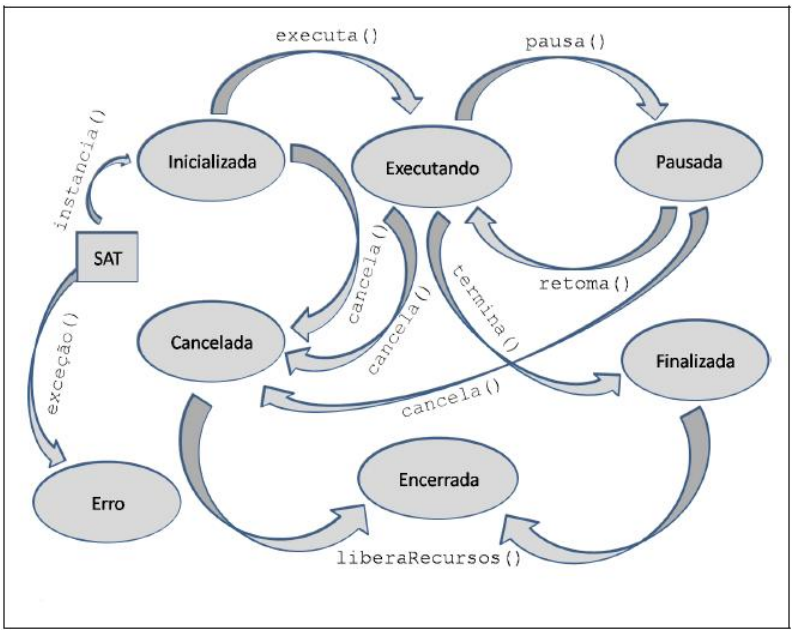

Figura 10: Ciclo de vida das tarefas (FERREIRA, 2009)

Como apresenta a figura 10, a tarefa é iniciada pelo Serviço de Acesso a Tarefas. Neste momento, ela pode ser instanciada, caso todos os recursos necessários (recursos físicos) estejam disponíveis, ou pode levantar uma exceção. Segundo Ferreira (2009a), as tarefas após instanciadas, obrigatoriamente, estão em um dos cinco estados:

- Inicializada *(Initialized).* A tarefa foi criada e inicializada recentemente*;*
- Executando *(Running).* A tarefa está ativa e executando no momento;
- Retomada (*Resumed*). A tarefa, que já estava em espera anteriormente, está ativa no momento e pronta para ser executada novamente;
- Pausada *(Paused).* A tarefa foi interrompida e pode ser ativada (retomada) a qualquer momento;
- Finalizada (*Terminated*). A tarefa foi finalizada por ter encerrado a sua operação.

Computacionalmente, as tarefas possuem um conjunto de informações que é importante para a correta utilização e gerenciamento dessas. As informações de uma tarefa são (SILVA, 2009a):

- Criador. Identifica o criador da tarefa, sendo que cada tarefa possui apenas um único criador;
- Estado. Para o gerenciamento das tarefas, é necessário o conhecimento de em qual estado ela se encontra. Lembrando que uma tarefa obrigatoriamente encontra-se em um dos cinco estados citados;

- Descrição. Descrição textual das tarefas;
- Contexto. Esta informação é o foco principal desta dissertação e será abordado com mais detalhes no capítulo 4;
- Código. As tarefas possuem um código associado a elas, onde se encontram as instruções de código que serão executadas para a obtenção das funcionalidades particulares de cada tarefa.

Para atender aos requisitos acima, acredita-se que uma nova forma de projetar sistemas de informação em saúde (EHS) deve ser analisada. A arquitetura ClinicSpace se propõe a atingir esse objetivo.

3.3 Arquitetura ClinicSpace

Para permitir o suporte computacional à situação descrita e aos requisitos identificados, está-se desenvolvendo uma arquitetura de software, chamada ClinicSpace, integrando tecnologias e conceitos *da Mobile, Pervasive and Ubiquitous Computing*, *End-User Programming e Contex-Aware Computing*. A descrição das camadas dessa arquitetura se dá, a partir da visão do usuário-final (médico), até a camada de execução e gerenciamento do ambiente pervasivo pelo *middleware* EXEHDA (YAMIN et al, 2005), que integra a arquitetura.

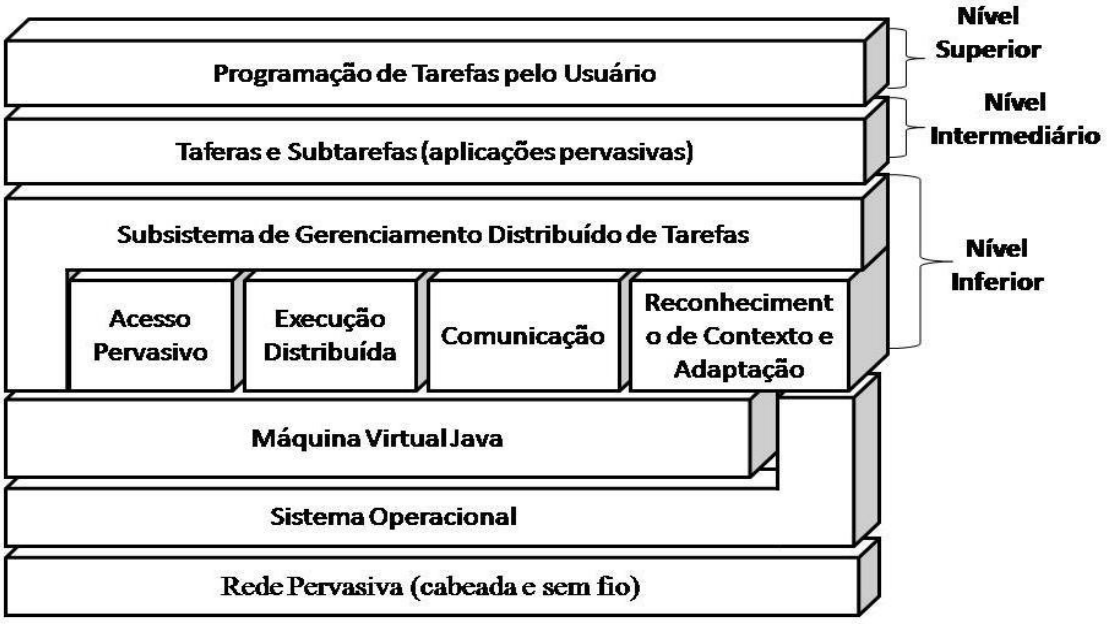

Figura 11: Arquitetura ClinicSpace (FERREIRA, 2009a)

A arquitetura para a programação e gerenciamento personalizado das tarefas, mostrada na Figura 12, foi organizada em níveis que refletem as visões do sistema: (i) nível superior, é composto pelo usuário-final (médico) que interage com a ferramenta para (re)definir suas tarefas que executarão num ambiente pervasivo; (ii) nível intermediário, é composto pelo mapeamento entre tarefas (definidas pelo usuário) e subtarefas (aplicações pervasivas) e pelo gerenciamento de ambas; (iii) nível inferior, é composto pelo conjunto de serviços do *middleware* de gerenciamento do ambiente pervasivo e de suporte à execução das aplicações pervasivas: EXEHDA (FERREIRA, 2009c).

3.3.1 Componentes da Arquitetura ClinicSpace

O ClinicSpace é composto por: (i) Interface de Edição de Tarefas e Contexto (IETC); (ii) Subsistema Gerenciador Distribuído de Tarefas (SGDT) (FERREIRA, 2009b); (iii) bases de suporte: banco de dados de contexto, banco de dados pervasivo de informações do paciente; (iv) pEHS – sistema pervasivo de informações em saúde (VICENTINE, 2010); (v) *Middleware* EXEHDA de gerenciamento do ambiente pervasivo (YAMIN et al, 2005).

O Subsistema Gerenciador Distribuído de Tarefas (SGDT) (FERREIRA, 2009b) faz a mediação entre o *middleware* de controle pervasivo de ambientes (EXEHDA), o Sistema Eletrônico de Saúde Pervasivo (pEHS), as bases de dados, e a Interface de Edição, provendo o acesso às informações de forma transparente, de acordo com as tarefas programadas pelo usuário clínico.

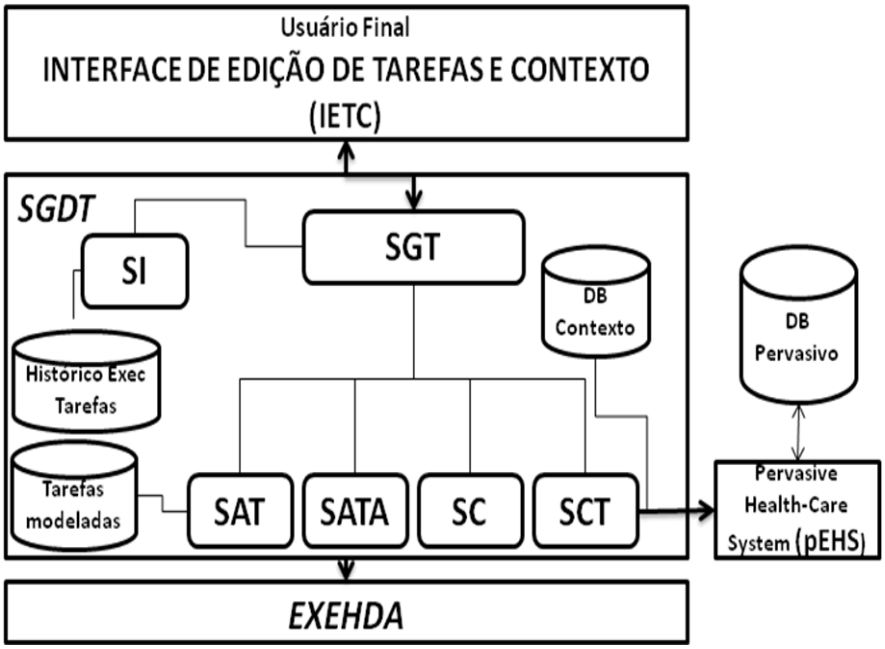

Figura 12: Componentes ClinicSpace

Como ilustra a figura 12, a arquitetura implementa o Subsistema de Gerenciamento Distribuído de Tarefas, seguindo uma linha de modularização existente no EXEHDA; assim, implementa módulos que disponibilizam serviços, que são gerenciados pelo Serviço Gerenciador de Tarefas.

A seguir, é apresentada de forma geral a arquitetura; os detalhes de cada desenvolvimento podem ser encontrados nas referências (FERREIRA, 2009a), (MARCOS, 2010), (RIZETTI, 2009), (SILVA, 2009a) e (VICENTINI, 2010).

3.3.2 Interface de Edição de Tarefas (IET)

Para que o profissional clínico possa criar e editar as tarefas e fluxos de execução, foi implementada a IET (Interface de Edição de Tarefas) (SILVA, 2009), a qual foi estendida nessa dissertação para IETC (Interface de Edição de Tarefas e Contexto), baseada na programação orientada ao usuário final. Tal interface utiliza os recursos providos pela arquitetura ClinicSpace para sua execução, com características ubíquas, tornando a infra-estrutura invisível para o usuário final.

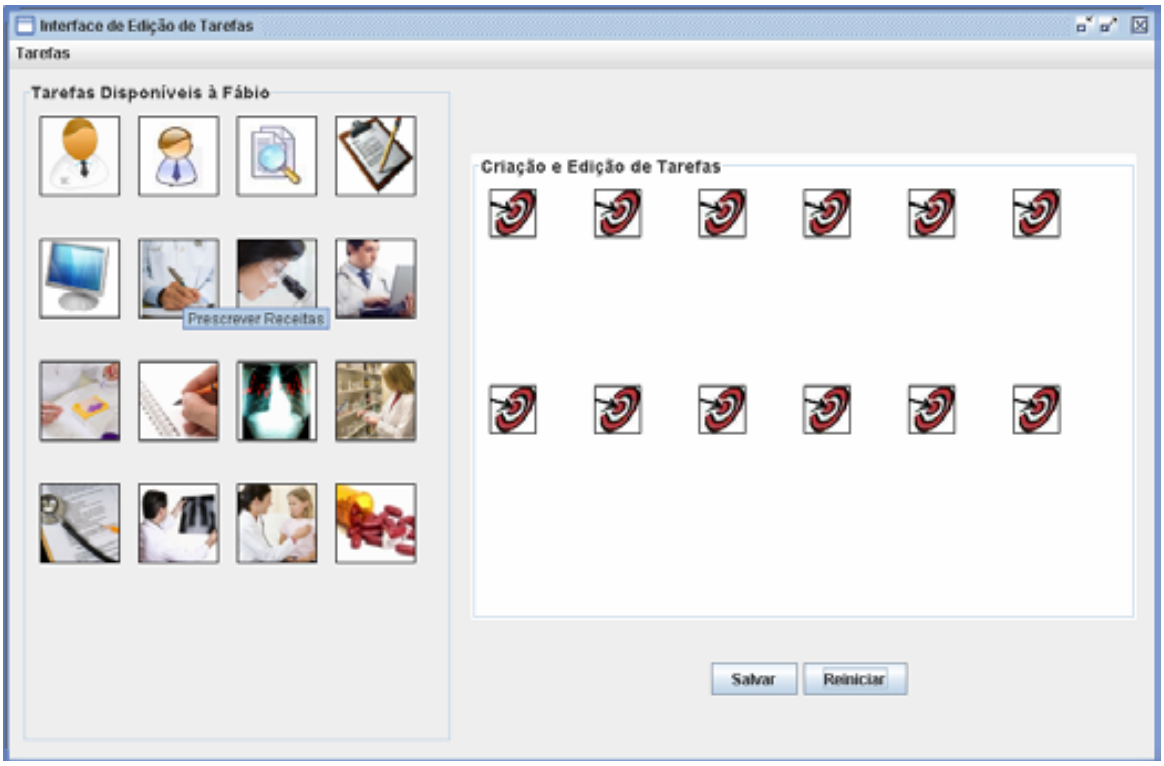

Figura 13: Interface de Edição de Tarefas (SILVA, 2009b)

A primeira versão da interface (figura 13) foi concebida no trabalho de Silva (2009b). Em sua primeira execução, a ferramenta carrega as 11 tarefas básicas (ver seção 3.2) que são a base para que o médico possa compor novas tarefas e fluxos de execução. Para tal, o médico arrasta o ícone correspondente à tarefa desejada e assim pode compor um fluxo de execução para modelar sua atividade clínica. Posteriormente, o médico salva o fluxo desejado, e a ferramenta edita as configurações em um arquivo descritor XML.

Associações dos elementos de contexto a uma atividade clínica é objeto desta dissertação, juntamente com a possibilidade de o usuário expressar quais elementos de contexto são de seu interesse para uma atividade específica. Tais requisitos resultaram em uma nova definição da interface e são apresentados no capítulo 4.

3.3.3 Serviço de Gerenciamento de Tarefas (SGT)

O serviço SGT é o núcleo do SGDT e é responsável pelo gerenciamento do ciclo de vida das tarefas. Atua como uma camada mediadora entre a aplicação e os demais serviços, sendo construído com base no *design patterns Facade (*GAMMA, 1995*),* realizando a

abstração de todos os serviços para a interface gráfica. Após a modelagem das tarefas pelo usuário, o SGT instancia o Serviço de Acesso a Tarefas (SAT) e inicia o serviço de armazenamento das tarefas provido por ele. Também, gerencia as interrupções existentes na execução das tarefas utilizando o Serviço de Acesso a Tarefas Ativas (SATA), assim como faz chamadas a aplicações existentes no pEHS (ver seção 3.3.9) necessárias à execução de uma subtarefa específica. Maiores informações são descritas por Ferreira (2009a).

3.3.4 Serviço de Acesso as Tarefas (SAT)

O SAT é responsável por realizar o carregamento das tarefas do usuário. Este serviço utiliza o *middleware* EXEHDA para carregar as tarefas do usuário com possibilidades móveis e execução distribuída.

Tal serviço realiza *parses XML* entre o arquivo descritor e objetos Java para montar a estrutura de dados de um objeto `Tarefa` na memória da aplicação. Para isso acessa a `Base de Tarefas Modeladas` que armazena o descritor das tarefas com vínculo de seu dono. Esta base armazena as tarefas originais (11 tarefas mínimas) e reproduz cópias vinculadas para cada usuário, após sua modificação pela IETC. Este serviço sofreu alteração para suportar a execução dinâmica do contexto de interesse do usuário (médico).

3.3.5 Serviço de Acesso a Tarefas Ativas (SATA)

O serviço SATA provê a possibilidade de interrupção e retomada de execução de tarefas, bem como a migração de tarefas entre diferentes dispositivos utilizados pelo usuário. Este serviço mantém a informação do estado da tarefa (ver seção 3.2.1). Por exemplo, uma tarefa pode estar ativa quando foi inicializada pelo SGT e ainda não foi finalizada, neste estado ela pode sofrer interrupções devido à troca de dispositivo pelo usuário, vindo a entrar no estado de pausada ou cancelada. Tais estados são informados ao SGT para tomada de providências na execução do ciclo de vida da tarefa.

3.3.6 Serviço de Contexto para Tarefas (SCT)

O serviço SCT armazena o relacionamento das tarefas do usuário com o contexto configurado no momento da modelagem. Além disso, o SCT identifica elementos do contexto que podem servir para a diminuição da entrada manual de dados nos formulários do pEHS, facilitando o seu uso e diminuindo a propensão a erros humanos (RIZZETTI, 2009). Ainda, o usuário pode realizar o agendamento de execução de determinadas tarefas quando o contexto estiver em um estado pré-determinado. O desenvolvimento desse serviço é tratado em detalhes no trabalho realizado por Rizzetti (2009). Este serviço sofreu alteração para suportar a execução dinâmica do contexto de interesse do usuário (médico).

3.3.7 Serviço de Inferência de Tarefas (SIT)

O SIT tem a finalidade de inferir as próximas tarefas que um usuário deseja executar. Para tanto, monta uma árvore de decisão (algoritmo C 4.5 para a construção de uma árvore de decisão sobre os elementos do contexto)(SOUZA, 2010) baseada em informações do contexto provida pelo *middleware* EXEHDA e o histórico de execução das tarefas do usuário.

O Histórico de Execução das Tarefas é constituído de um Banco de Dados relacional; assim, pode-se traçar o perfil de utilização do usuário e, com isso, realizar inferências sobre as tarefas a serem executadas no futuro. Maiores informações sobre esse serviço são descritas por Souza (2010a).

3.3.8 Serviço de Colaboração (SC)

Esse serviço foi projetado para atender a necessidade de colaboração entre os profissionais em um ambiente clínico. Segundo Bardram & Christensen (2007), a colaboração é um aspecto fundamental no trabalho de médicos, enfermeiros, e outros profissionais da saúde.

Características como colaboração entre uma atividade médica, como diagnóstico, ou a

delegação de uma atividade não terminada por um médico devido à troca de turno de trabalho, estão sendo identificadas para que o Serviço de Colaboração possa ser utilizado em tempo de execução.

Uma forma muito freqüente de colaboração entre os profissionais da saúde é a consulta à opinião de outro profissional (BARDRAM; CHRISTENSEN, 2007). Por isso, foi previsto, para o Serviço de Colaboração, o suporte à comunicação dos usuários, inicialmente por troca de mensagem, podendo ser expandido para formas mais robustas, como vídeo-conferência. Através dessa funcionalidade, os profissionais podem compartilhar informações sobre o paciente, discutir problemas e soluções, solicitar e dar opiniões sobre determinado caso. O desenvolvimento desse serviço é tema de dissertação do mestrando Marcelo Lopes Kroth.

3.3.9 Sistema Eletrônico de Saúde Pervasivo (pEHS)

O pEHS (VICENTINI, 2010) é um sistema voltado a prover funcionalidades relativas ao gerenciamento de informações de saúde do paciente, bem como informações dos clínicos. É denominado pervasivo pois contém uma base de dados acessível em qualquer lugar, a todo tempo, com qualquer dispositivo; sendo assim, o histórico de saúde do paciente é acessível de diferentes instituições de saúde, sem restrições de domínio de rede.

Esse sistema é organizado como um conjunto de aplicações autocontidas, ou seja, é possível combinar as funcionalidades para formar outra aplicação. As aplicações-básicas, projetadas atualmente, constituem as implementações das 11 subtarefas ClinicSpace (básicas). Por exemplo, a subtarefa *obter informações específicas em registros de pacientes* corresponde à chamada de uma aplicação do pEHS que recebe as informações de contexto como parâmetro e devolve as informações obtidas da Base de Dados Pervasiva que ele gerencia, na forma contextualizada. O pEHS é estruturado utilizando o *design pattern Data Access Object* (DAO) (MARLKS, 2004). Os serviços implementados através deste padrão são disponibilizados através do conceito de interfaces para o SGDT (FERREIRA, 2009b). Mais informações sobre o pEHS são descritas por Vicentine (2010).

3.3.10 *Middleware* de Gerenciamento para Ambientes Pervasivos – EXEHDA

O *Middleware* EXEHDA (YAMIN; AUGUSTIN, 2005), provê os serviços de mais baixo nível da arquitetura proposta, abstraindo do programador de subtarefas e aplicações pEHS a dinamicidade e heterogeneidade do ambiente pervasivo, o controle da adaptação, a mobilidade física e lógica de recursos, a portabilidade e a conectividade no tratamento de conexões e desconexões realizadas por dispositivos de acesso sem fio.

O EXEHDA disponibiliza os serviços pertinentes ao gerenciamento pervasivo de ambientes. Os serviços de Acesso Pervasivo gerenciam a *base de dados da aplicação (BDA)*, para disponibilização de dados e código de cunho mais abrangente, não pertencente a usuários específicos (como o banco de dados do contexto e o pEHS). Também gerencia o *ambiente virtual do usuário (AVU)*, para a disponibilização de dados particulares vinculados a usuários específicos (exemplo, tarefas modeladas) em qualquer lugar, a qualquer momento, com qualquer dispositivo. Os serviços de Execução Distribuída provêm à possibilidade de execução distribuída dos serviços e aplicações. Os serviços de Comunicação gerenciam a troca de mensagens e a descoberta dinâmica de outros serviços e recursos separados geograficamente, bem como o tratamento das desconexões existentes em um ambiente pervasivo, devido à infra-estrutura de hardware sem fio. Os serviços de Reconhecimento de Contexto e Adaptação (SRCA) provêm à infra-estrutura para o gerenciamento dos elementos do contexto, disponibilizando através do processo de subscrição (inscrição para recebimento de informação desejada) os dados coletados pelos sensores.

Nesta dissertação dá-se ênfase a este último subsistema, pois é este que fornece as informações dinâmicas provenientes dos sensores (contexto) para as aplicações pervasivas. O SRCA implementa o serviço *Coletor*, que é responsável pela extração das informações brutas (diretamente dos recursos envolvidos) que, posteriormente refinada, dará origem às informações de contexto. Para isto, o serviço *Coletor* aglutina a informação oriunda de vários componentes monitores (*Monitor*) e as repassa aos consumidores registrados (*MonitoringConsumer*). Um componente *Monitor* gerencia um conjunto de sensores parametrizáveis.

Utilizando as ferramentas descritas neste capítulo, modelou-se a associação de contexto clínico às tarefas, o resultado é descrito no próximo capítulo.

4 CONTEXTO NA ARQUITETURA CLINICSPACE

Este capítulo aborda a modelagem do contexto clínico desenvolvida neste trabalho, bem como a forma de associação realizada entre o contexto e às tarefas. Apresenta as alterações realizadas na arquitetura ClinicSpace para a inclusão destes conceitos.

4.1 Modelagem de Contexto Clínico

O contexto clínico até o momento considerado está vinculado aos elementos de contexto *tempo, localização, paciente, dispositivos, sensores, recursos* e *perfil,* que geram informações temporais, atemporais e deduzidas. Como entrada de dados levam-se em conta as informações coletadas pelos sensores, informações disponibilizadas pelos sistemas eletrônicos de saúde (EHR – *Electronic Health Record*) e informações deduzidas através da composição das duas informações conhecidas. A seguir, são detalhados os artefatos para a modelagem de contexto clínico utilizados na arquitetura.

4.1.1 Elementos de Contexto em um Ambiente Pervasivo

Na modelagem clínica proposta, os elementos de contexto podem se relacionar com outros elementos na forma de associação composta. Por exemplo, num *paciente* são colocados *sensores* que monitoram suas funções vitais; um *médico* utiliza um *dispositivo* para a realização de alguma *tarefa* e outros *recursos* (como rede e impressoras) do ambiente.

Na concepção da arquitetura ClinicSpace, uma tarefa tem um conjunto de elementos (contexto) que ela deve ser ciente e se adaptar em tempo de execução. Um médico é *dono* de um conjunto de tarefas, podendo manipulá-las conforme sua preferência.

O médico realiza as atividades ou *tarefas* em um contexto relevante ao sistema pervasivo. Tem uma associação ao elemento de contexto *perfil*, onde o sistema se adapta a

este elemento de contexto e apresenta informações relevantes, considerando o *perfil* do médico.

O *paciente* é o motivo pelo qual existe uma estrutura hospitalar. Tal elemento está relacionado impreterivelmente ao elemento *sensor*, os quais informam de forma implícita os seus dados (*informação dinâmica*) ao sistema. O pEHS disponibiliza as *informações estáticas*, como por exemplo o histórico de saúde do *paciente*. O elemento *localização* disponibiliza coordenadas de lugares onde médicos, *pacientes* e *dispositivos* se localizam e onde as atividades são realizadas, também se relacionam com os *recursos* disponibilizados para esta localização.

O elemento *dispositivo* descreve o aparelho utilizado pelo médico para interação com o sistema pervasivo. Este elemento se relaciona com os *recursos* do ambiente para melhor dispor as informações. As instâncias desse elemento são dispositivos reais, tais como *smartphones*, *celulares*, *PC*, *notebook* e outros utilizados para acessar as informações e disponibilizar da melhor forma possível (de acordo com seus recursos próprios) para o médico.

Os *sensores* são elementos físicos ou lógicos utilizados para coleta de dados do ambiente, vinculados a médicos e *pacientes* para autenticação ao sistema, como também monitoramento de funções vitais. Este elemento é o principal fornecedor de dados implícitos e dinâmicos sobre o paciente às aplicações pervasivas. Possui como principal atributo o *atuador,* que possibilita criar regras para as informações obtidas, as quais disparam ações automáticas pré-definidas anteriormente (pro-atividade do ambiente). Por exemplo, se o sensor de temperatura do paciente está emitindo valores acima de 40º Celsius, é ativada a chamada *"urgência"* ao profissional de saúde.

O elemento *recurso* é quem descreve os meios para auxiliar a obtenção das informações pelos dispositivos, como recursos de rede e hardware. Tal elemento se relaciona com os *dispositivos*.

O elemento *tempo* especifica um data/hora para disparo de uma tarefa, sendo assim configurável pelo médico na associação do contexto a uma tarefa.

4.1.2 Fonte das Informações de Contexto

As características das informações de contexto estão vinculadas a informações dinâmicas (sensoradas e/ou e informadas), informações estáticas (históricas) e informações deduzidas (extraída da composição das duas primeiras) e são baseadas na pesquisa de Henriksen et al. (2002).

As *informações dinâmicas* variam periodicamente de acordo com a relação existente entre as entidades, pois tem um período de validade (vida). Por exemplo, um médico está na sala de pronto atendimento. Esta informação só é útil enquanto a informação "médico saiu da sala" não for verdadeira. A principal fonte de dados dinâmicos são os sensores monitorados pelo *Middleware* EXEHDA. Contudo, as informações históricas de contexto podem ser levadas em consideração para geração de módulos pró-ativos, vindo a inferir futuros contextos. Atualmente, a arquitetura ClinicSpace está utilizando algumas informações históricas no Serviço de Inferência.

Já as *informações estáticas* são obtidas através de Sistemas Eletrônicos de Informações de Saúde (como o pEHS), as quais disponibilizam as informações dos elementos de contexto, como os dados pessoais de um paciente, descrição de um dispositivo entre outras. São informações estáticas, pois possivelmente não mudam de valor durante um período considerável de tempo.

As *informações deduzidas* são criadas a partir da interpretação do conjunto de informações dinâmicas e estáticas. Isso é útil para os sistemas que necessitam de informações geradas a partir de uma dedução lógica das informações, oriundas do contexto corrente (inferência). Estas informações ainda não são geradas pela arquitetura.

4.1.3 Modelo de Contexto proposto para Ambientes Clínicos

A modelagem gráfica do contexto proposta neste trabalho foi baseada nos estudos apresentados no capítulo 2, os quais propuseram artefatos que contemplam a modelagem de elementos de contexto gráfica, ontológica e orientados a objetos.

A representação genérica do contexto clínico é ilustrada na figura 14, onde o diagrama não expressa o tipo de informações trocadas entre os elementos, devido a este não ser contemplado pela UML.

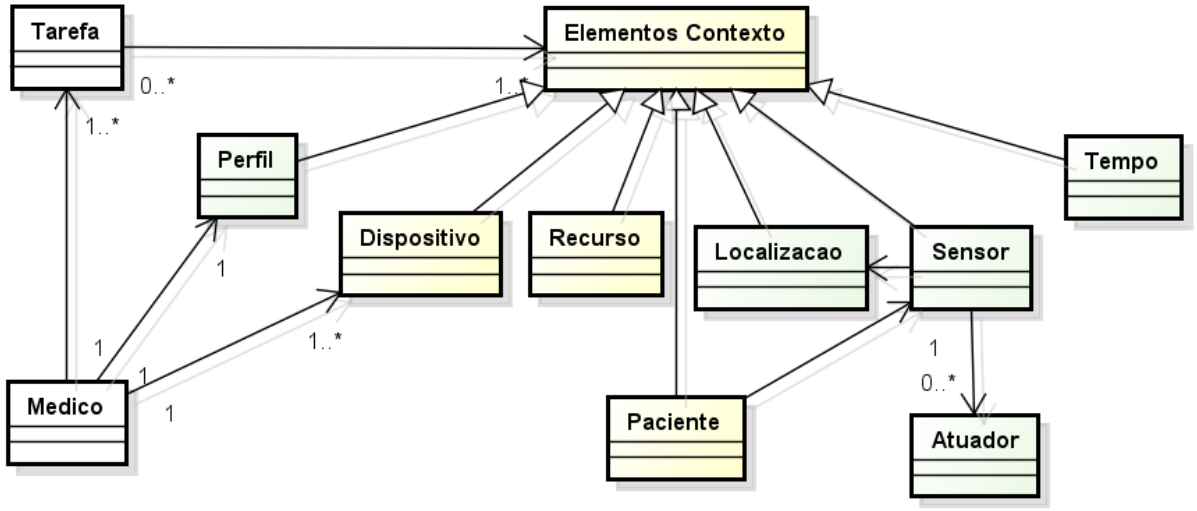

Figura 14: Modelagem do contexto clínico

Na figura 14, percebem-se as relações entre os elementos de contexto, onde paciente está sendo monitorado pelos sensores de sinais vitais, por exemplo. O médico tem uma ou várias atividade a realizar (Tarefa) e, para isso, utiliza algum dispositivo (móvel ou fixo) com recursos obtidos no ambiente, os quais são disponibilizados de acordo com o seu perfil. Deste modo, todos os elementos de contexto derivam de uma classe genérica, a qual encapsula os comportamentos pertinentes a todos os elementos de contexto.

A modelagem do contexto onde a aplicação pervasiva irá executar é necessária para que a mesma conheça os elementos de contextos existentes em um espaço pervasivo que são relevantes para ela. A proposta de um modelo genérico de contexto clínico (i) ajuda a alavancar a área de Engenharia de S*oftware* em Computação Ubíqua na Saúde, sendo esse modelo capaz de integrar elementos de contexto em uma aplicação pervasiva, tal ajuda é evidenciada principalmente pela escassez de trabalhos em Engenharia de *Software* em Computação Ubíqua, somando-se a área clínica, até o momento não foram encontrados trabalhos com esta abordagem (modelagem de contexto clínico) (ii) evita um grande *overhead de controlar todo o contexto de um ambiente*, pois possibilita que a aplicação conheça quais elementos de um ambiente ela deverá levar em conta, quando sua execução distribuída estiver

sensorando uma atividade em um ambiente específico, assim evitando processamentos desnecessários.

4.2 Associação do Contexto às Tarefas

Como visto, na arquitetura ClinicSpace, as atividades são decompostas em tarefas e subtarefas, seguindo a forma particular de cada indivíduo realizá-la (personalização). As tarefas tem descrição ontológica e são implementadas diretamente como objetos Java, gerenciados pelo ambiente pervasivo definido pela arquitetura.

Na ontologia que descreve cada tarefa é especificado o seu contexto, o qual é responsável por manter os contextos associados às tarefas e o vínculo com seu dono (médico). Esta ontologia de tarefas e contexto foi modelada na ferramenta Protégé 3.4.4 (http://protege.stanford.edu) gerando o arquivo OWL-DL, com a estrutura semântica original dos conceitos. Tal versão do Protégé foi utilizada, pois as demais versões como a 4.1 ainda não dão suporte à utilização de inferências (SWRL).

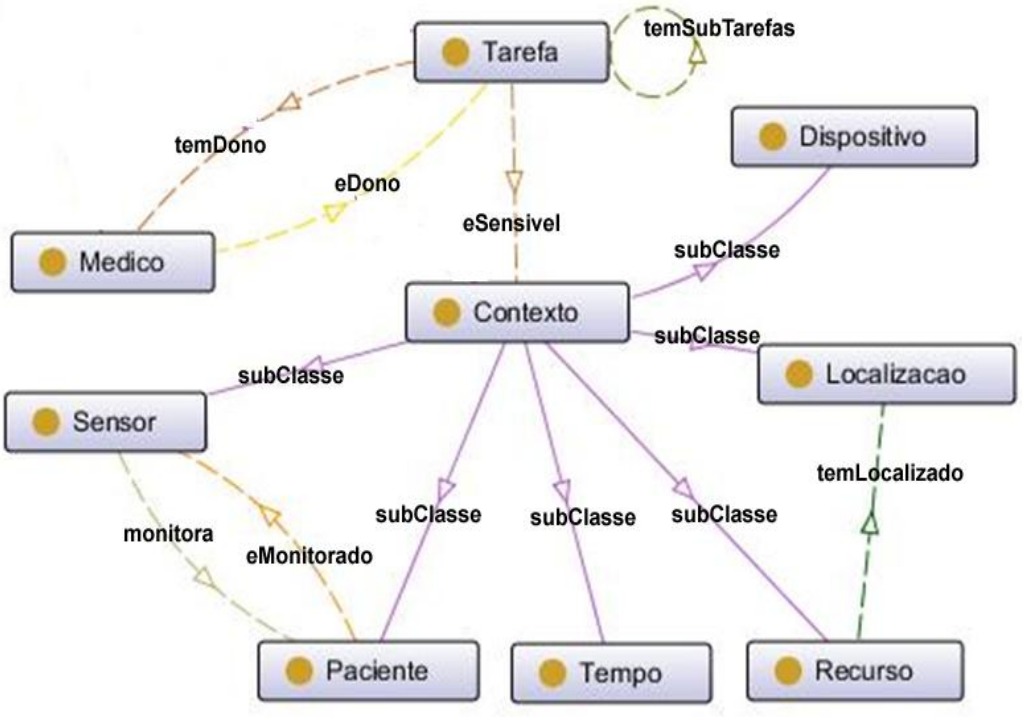

Figura 15: Representação gráfica do modelo ontológico de tarefa e contexto

A figura 15 apresenta a taxonomia ontológica que representa os conceitos `Médico`, `Tarefa` e `Contexto`. Este gráfico foi gerado através do plugin OntoGraf no Protégé 4.1 e algumas relações entre as classes foram omitidas para melhor compreensão da ontologia.

A `Tarefa` é sensível a um `Contexto`, sendo a propriedade *eSensivel* uma propriedade funcional (determina que somente existe *um* individuo `Contexto` vinculado ao individuo `Tarefa`) entre `Tarefa` e `Contexto`. Uma `Tarefa` contém a propriedade *temSubTarefas* que realiza uma relação com outros indivíduos `Tarefa`, sendo assim criando o conceito que uma tarefa em determinado momento pode ser subtarefa de outra tarefa, e por fim uma `Tarefa` tem uma relação com `Médico` determinando seu dono, deste modo médico é dono (propriedade *dono*) de uma ou várias tarefas, e uma tarefa tem um dono (propriedade inversa *temDono*).

A classe *Contexto* representa todos os elementos de contexto descritos anteriormente, cada qual com relações específicas entre eles, tais relações são: (i) propriedade *eMonitorado* que representa a relação entre Paciente, Médico e os Sensores que monitoram seus sinais vitais; (ii) propriedade *monitorado* que realizada a relação inversa com *eMonitorado,* descrevendo quais pacientes e médicos o sensor está coletando os dados; (iii) propriedade *temLocalizacao* que descreve a relação entre Localização e as classe Recurso, Paciente e Médico, fornecendo as coordenadas da localização de cada indivíduo relacionado.

A geração de indivíduos do tipo Tarefa e Contexto utilizando a propriedade *eSensivel* faz a associação destes indivíduos. Para o médico, basta ele selecionar quais indivíduos de um contexto é de seu interesse e associar a uma tarefa específica. Sendo assim, no momento da execução da tarefa pelo seu *dono,* tem-se a informação a quais elementos de contexto ela deve ser sensível.

4.3 Modificações realizadas nos serviços do ClinicSpace

Para que o médico pudesse expressar seu contexto de interesse em uma determinada tarefa, utilizando a ontologia criada para melhor expressividade na modelagem das tarefas e na associação dos elementos de contexto, foram necessárias diversas modificações na arquitetura ClinicSpace existente.

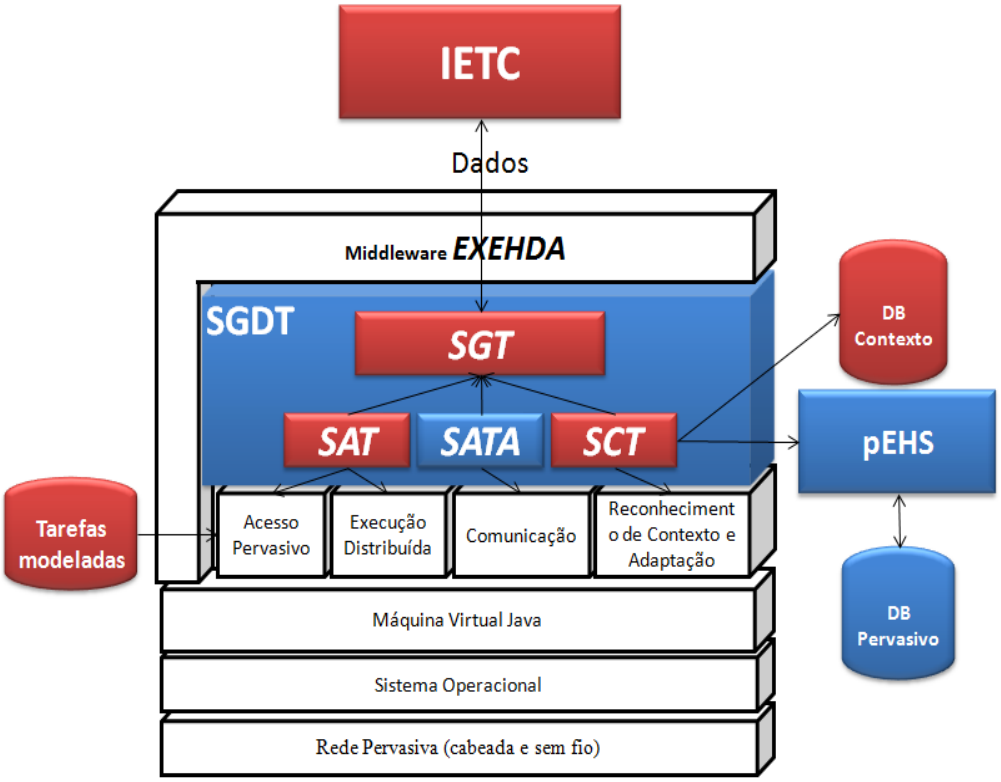

Figura 16: Modificações na Arquitetura ClinicSpace

A Figura 16 destaca as modificações realizadas na Arquitetura ClinicSpace; em vermelho, estão os serviços que sofreram modificações (SGT, SAT, SCT) e a nova Interface de Edição de Tarefas e Contexto (IETC), bem como os repositórios criados (Banco de Tarefas, Banco de Contexto), em azul e branco, os demais serviços que não sofreram modificações.

4.3.1 Interface de Edição de Tarefas e Contexto (IETC)

Interface gráfica de associação dos elementos de contexto de interesse (IETC) foi desenvolvida para permitir ao médico definir a quais elementos de contexto (ver seção 4.1) uma determinada tarefa deve ser sensível, diminuindo assim a carga existente nos gerenciadores de contexto e na implementação da tarefa.

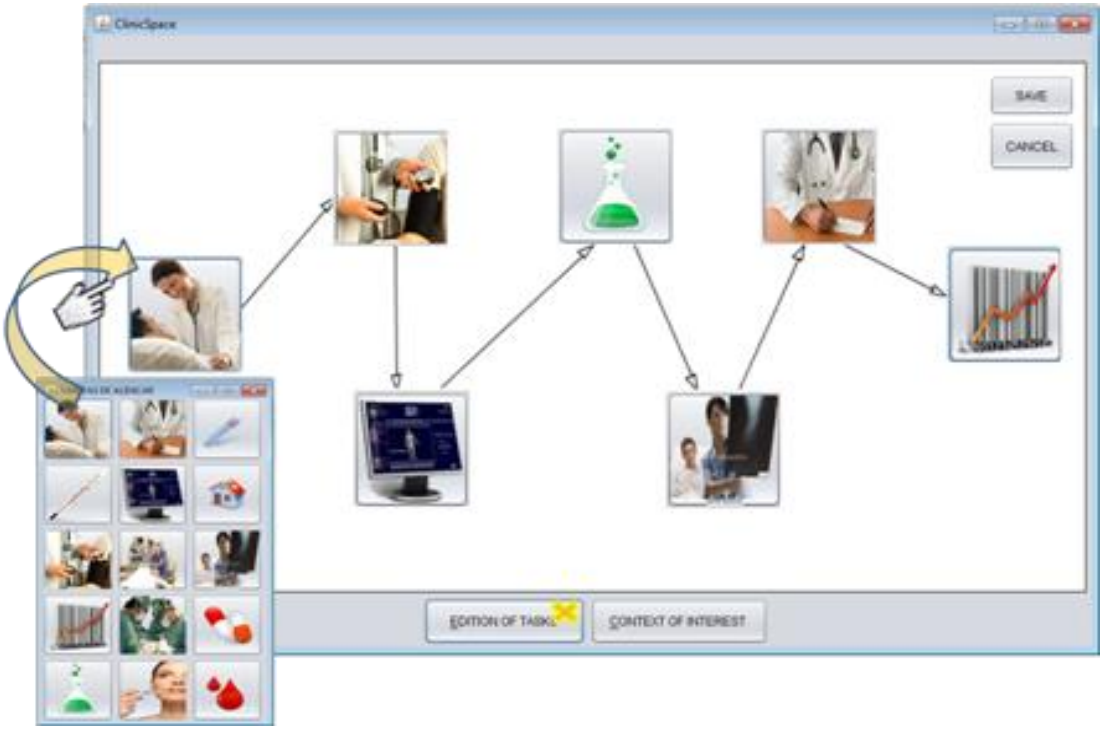

Figura 17: Interface inicial da IETC

A IETC foi re-implementada neste trabalho para ter um apelo mais *touch*, assim dando a possibilidade de o médico interagir com a interface pelo toque dos dedos, como ilustra a figura 17. Tarefas, subtarefas e contextos podem ser representados por ícones que remetam às funcionalidades específicas de cada elemento e, ainda, servem para abstrair aplicações, serviços, recursos, etc. Os diagramas são utilizados para representar o fluxo de execução de tarefas e subtarefas e o fluxo das informações que trafegaram entre os vários elementos. O objetivo é fornecer uma interface intuitiva que possibilite ao usuário clínico, através da técnica de "arrastar-e-soltar", definir a modelagem das suas tarefas diárias.

A funcionalidade de adicionar uma subtarefa foi mantida, as mudanças ocorreram na tela de modelagem, a qual foi implementada levando em consideração técnicas de programação visual das tarefas, baseada nas ferramentas de modelagem UML tais como ArgoUML(http://argouml.tigris.org/) e o plugin UML do NetBeans (http://wiki.netbeans.org/UML/).

Inicialmente, a IETC faz uma chamada ao SGT para ter acesso a todas as tarefas que o médico é dono (*getEDono*()). Desta forma, monta o menu de tarefas sempre gerando um componente *JButton* novo na tela para cada objeto `Tarefa` retornado pela chamada ao método.

Após a montagem do menu, com a funcionalidade de arrastar e soltar (*API Drag and*

Drop) é possível criar um fluxo de execução (workflow) das tarefas, podendo realizar sua modelagem, conforme interesse do médico.

Como apresenta a figura 17, foram adicionadas diversas tarefas montando um fluxo de execução. Após essa configuração, salvando a modelagem, é possível dar um nome intuitivo para essa composição, assim as tarefas adicionadas tornam-se subtarefas dessa nova modelagem criada, que, consequentemente, ao ser salva vai gerar uma tarefa composta por subtarefas.

Para definição do contexto de interesse em uma subtarefa específica, foi criado um menu que apresenta aos elementos de contexto descritos na seção 4.1. Este menu é estático e cada botão nele representa um elemento de contexto. Assim, é possível adicionar um elemento de contexto para uma subtarefa apresentada na ferramenta. Cada elemento de contexto tem uma interface própria para seleção do indivíduo de interesse do médico para aquela tarefa.

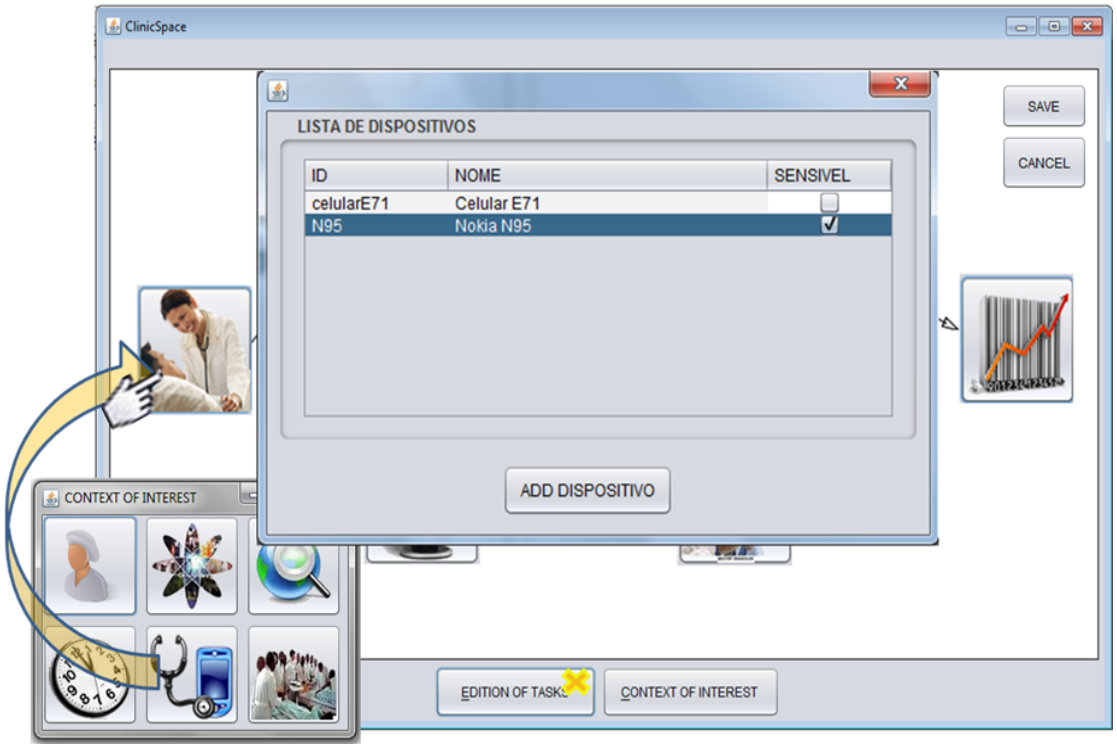

Figura 18: Lista de Dispositivos

A figura 18 apresenta a interface para adicionar elementos de contexto do tipo *Dispositivo*. Assim, é possível tornar a tarefa sensível a uma gama de dispositivos, e para isso é somente necessário que o médico adicione o elemento de contexto `Dispositivo` e selecione na tela de configuração quais dispositivos são de seu interesse.

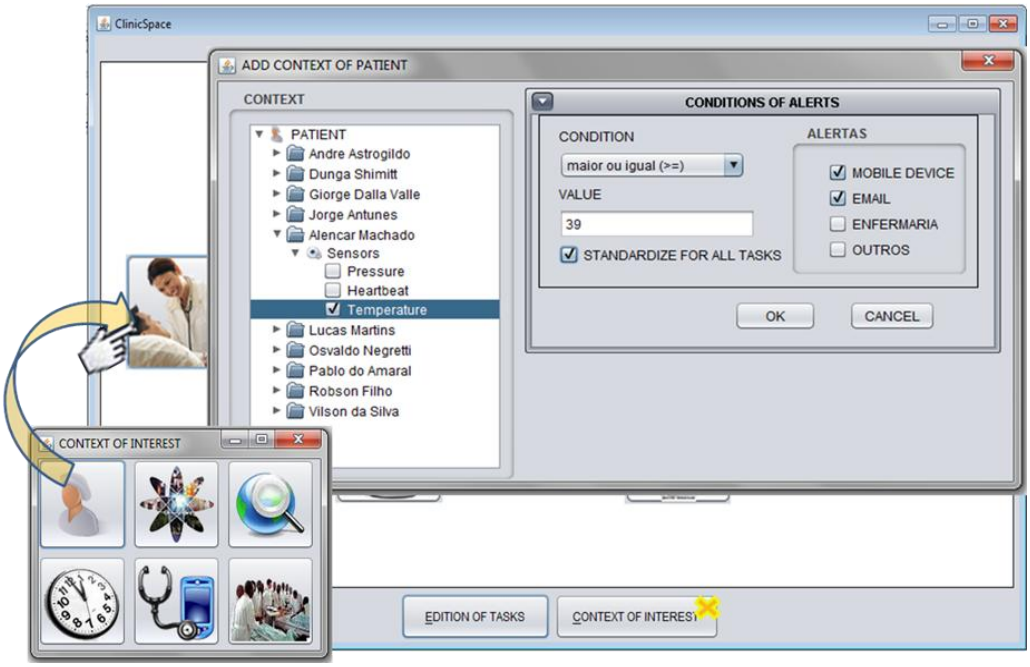

Figura 19: Associação de Contexto as tarefas através da IETC

Como mostra a Figura 19, ao associar o elemento de contexto `Paciente` a uma subtarefa na tela de modelagem, é aberta uma interface que lista todos os pacientes vinculados ao médico. Desta forma, é possível dizer que, se o paciente X estiver no contexto da tarefa, ela deve se comportar de uma forma distinta. Também é possível gerar *atuadores* para cada paciente apresentado para definir ações pro-ativas a serem executadas. Por exemplo, se o sensor de temperatura do paciente Alencar Machado estiver fornecendo valores maiores ou iguais a 39 graus, o médico deve receber uma notificação pelo seu telefone celular e por e-mail. A interface gera uma regra (*atuador*) adicionada ao sensor que fica em execução até que seja verdadeira ou até que o indivíduo Alencar Machado deixe de existir na ontologia.

O *atuador* foi projetado para ser dinâmico e adaptativo em tempo de execução, devendo alterar sua execução de acordo com a regra gerada. Para tanto, o *atuador* recebe um script com a regra e cria uma *thread* desta regra, assim o corpo do código da *thread* torna-se dinâmico, mudando conforme as preferências do médico. Para implementar esse dinamismo de código em tempo de execução, foram utilizados os recursos da linguagem *Groovy* (http://groovy.codehaus.org/), que é uma linguagem de *scripts* 100% compatível com Java.

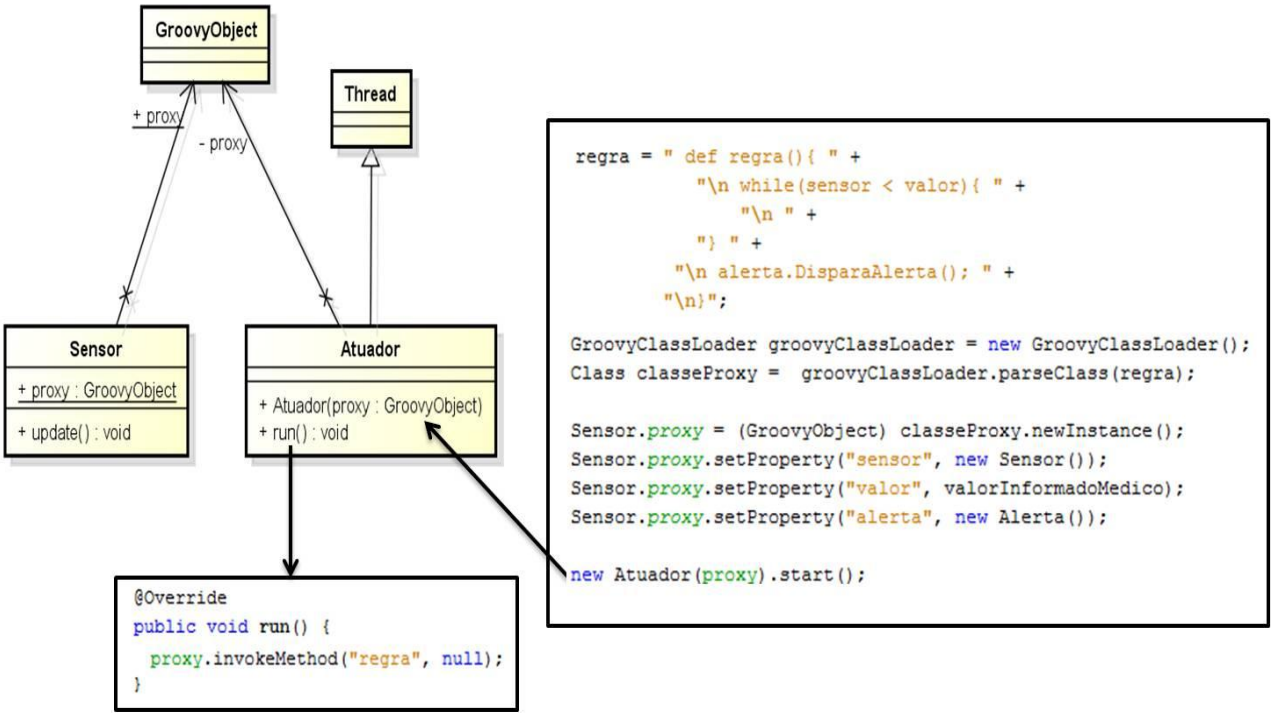

Figura 20: Diagrama de Classes Atuador

Desta forma, conforme mostra a figura 20, a interface monta uma string (regra) contendo o código *Groovy*, a qual é carregada pela classe *GroovyClassLoade*r. Para que a regra seja executada, basta instanciar um atuador, o qual implementa a interface Thread, consequentemente, reescreve o método *run*() para iniciar *Threads* dentro do ClinicSpace.

O resultado do carregamento pelo *Groovy* da regra criada é um objeto referenciado pelo atributo estático *GroovyObject* da classe Sensor. Deste modo, o Sensor tem a referência à *Thread* que está rodando e, desta forma, pode atualizar o valor do sensor para que a regra montada seja validada.

Este modelo gerado é uma solução genérica para gerenciar o dinamismo existente no contexto. Esta solução é estendida para todos os tipos de regras geradas pelo médico. Para a execução dinâmica, passa-se um objeto *GroovyObject* contendo o *script* da regra para o construtor do *Atuador* e faz-se uma chamada ao método *start* da *Thread*.

4.3.2 Serviço Gerenciador de Tarefas (SGT)

O serviço SGT é implementado utilizando o padrão de projeto *Facade*; sendo assim, atua como uma camada mediadora entre as aplicações e os demais serviços. Foi necessário realizar ajustes nos parâmetros e alterações na estrutura dos métodos, bem como inserção de novos métodos para atender a nova forma de modelagem de tarefas com contexto.

A principal alteração realizada neste serviço foi à mudança na forma de iniciar o código proveniente de uma tarefa. Esta funcionalidade está utilizando a API *Reflections [http://download.oracle.com/javase/tutorial/reflect/index.html]* do Java, permitindo executar uma aplicação em tempo de execução no ClinicSpace. Por exemplo, se o SGT necessita iniciar uma tarefa de "*Adicionar notas diárias sobre as condições do paciente*", a descrição ontológica desta tarefa contém a informação da localização desta aplicação no pEHS; sendo assim, precisa gerar uma instância desta aplicação para que o médico possa interagir com a tela da aplicação. Esta instanciação é feita em tempo de execução pelo SGT utilizando os recursos da API *Reflections,* desta forma realiza a introspecção (*Introspection*) no construtor da aplicação e pode passar como argumento o `Contexto`.

A partir das alterações realizadas, uma tarefa precisa implementar a interface chamada *SensivelContexto* (Figura 21) e codificar seu método *sensivelAoContexto*.

```
public interface SensivelContexto {
    public void sensivelAoContexto(Collection contexto);
}
```

Figura 21: Interface SensivelContexto

Assim, fica a cargo do programador de tarefas a forma de adaptação que a tarefa vai sofrer, levando em conta o tipo de contexto recebido. Por exemplo, o método sensível ao contexto *sensivelAoContexto(Collection contexto)* recebe uma coleção de contexto; este contexto pode ser composto por outros contextos ou ser simples, somente conter elementos de contextos (paciente, dispositivo ...). Desta forma, a tarefa identifica o tipo de contexto recebido (composto ou simples) e testa quais instâncias dos elementos de contexto existem nessa coleção, deste modo codifica a adaptação da tarefa de acordo com os elementos existentes no contexto.

4.3.3 Serviço de Acesso a Tarefas (SAT)

A maior modificação na arquitetura ClinicSpace aconteceu no Serviço de Acesso a Tarefa. Anteriormente, as tarefas estavam estruturadas em arquivos XML, onde o SAT realizava *parsers* para carregamento das tarefas, tais arquivos não continham nenhuma referência aos contextos que uma tarefa deveria ser sensível.

Atualmente, o SAT realiza o carregamento da estrutura ontológica apresentada na seção 4.2. Sendo assim, lê os indivíduos existentes na ontologia e cria os mesmos como objetos Java, para seu gerenciamento pelos serviços ClinicSpace.

Para o carregamento da ontologia, bem como sua modificação, foi adicionado à arquitetura ClinicSpace as bibliotecas disponíveis no WebProtégé (http://protegewiki.stanford.edu/wiki/WebProtege); assim, a ferramenta foi capaz de realizar todas as mudanças que o WebProtégé pode fazer em uma ontologia.

A re-implementação deste serviço foi feita através da criação de novas classes com a finalidade de criar, modificar, buscar e deletar as instâncias de todos os indivíduos existentes na ontologia.

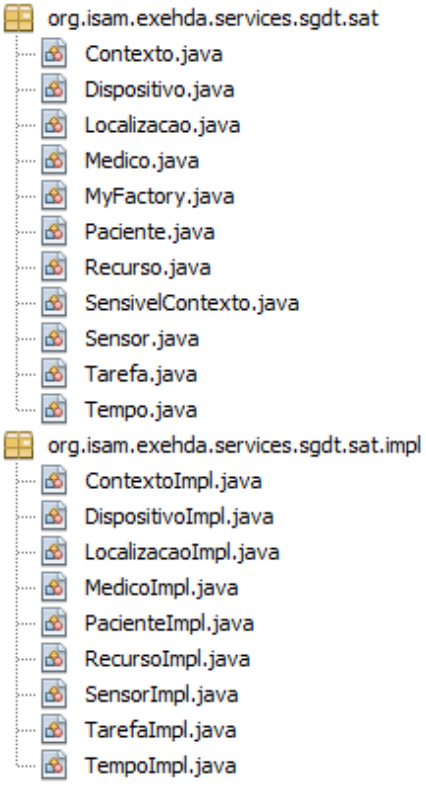

Figura 22: Estrutura de classes criadas

A figura 22 apresenta a estrutura de classes criadas. Tal estrutura foi declarada diretamente em interfaces Java (pacote *org.isam.exehda.services.sgdt.sat*), com classes correspondentes à implementação.

Destaca-se a classe *MyFactory* que tem a finalidade de ser a fábrica para gerenciar todos os objetos existentes na ontologia. Esta classe, inicialmente, precisa receber um objeto *OWLModel* que é obtido através do método *ProtegeOWL.createJenaOWLModelFromURI(),* que deve receber como parâmetro a URL do arquivo OWL. Os demais trechos de código apresentam a manipulação dos indivíduos `Paciente` e são semelhantes para os restantes das classes.

```
static {
    ProtegeJavaMapping.add("http://www.owl-ontologies.com/tarefa.owl#Tarefa",
            Tarefa.class, DefaultTarefa.class);
    ProtegeJavaMapping.add("http://www.owl-ontologies.com/tarefa.owl#Sensor",
            Sensor.class, DefaultSensor.class);
    ProtegeJavaMapping.add("http://www.owl-ontologies.com/tarefa.owl#Paciente",
            Paciente.class, DefaultPaciente.class);
    //...
 }
```

Figura 23: Classe MyFactory

Como ilustra a figura 23, é realizado um carregamento do indivíduo ontológico para uma instância de um objeto através do mapeamento URI/Classe Java.

```
public RDFSNamedClass getPacienteClass() {
    final String uri = "http://www.owl-ontologies.com/tarefa.owl#Paciente";
    final String name = owlModel.getResourceNameForURI(uri);
    return owlModel.getRDFSNamedClass(name);
}

public Paciente createPaciente(String name) {
    final RDFSNamedClass cls = getPacienteClass();
    if (name == null) {
        name = owlModel.getNextAnonymousResourceName();
    }
    return  new PacienteImpl(owlModel, cls.createInstance(name).getFrameID());
}
```

Figura 24: Obtenção do recurso e criação

A Figura 24 ilustra a obtenção do recurso através da URI de um paciente, bem como a criação de novos indivíduos do tipo `Paciente` na ontologia.

```
public Paciente getPaciente(String name) {
    RDFResource res = owlModel.getRDFResource(OWLUtil.getInternalFullName(owlModel, name));
    if (res == null) {return null;}
    if (res instanceof Paciente) {
        return (Paciente) res;
    } else if (res.hasProtegeType(getPacienteClass())) {
        return new PacienteImpl(owlModel, res.getFrameID());
    }
    return null;
}

public Collection<Paciente> getAllPacienteInstances() {
    return getAllPacienteInstances(false);
}

public Collection<Paciente> getAllPacienteInstances(boolean transitive) {
    Collection<Paciente> result = new ArrayList<Paciente>();
    final RDFSNamedClass cls = getPacienteClass();
    RDFResource owlIndividual;
    for (Iterator it = cls.getInstances(transitive).iterator();it.hasNext();) {
        owlIndividual = (RDFResource) it.next();
        result.add(new PacienteImpl(owlModel, owlIndividual.getFrameID()));
    }
    return result;
}
```

Figura 25: Obtenção de indivíduos

A figura 25 apresenta a forma de obtenção de indivíduos na ontologia, podendo ser um indivíduo específico ou mesmo todos os indivíduos de um conceito.

O diagrama de classe (figura 26) UML mostra toda estrutura implementada para disponibilizar ao SGT as possibilidades de remodelar a descrição ontológica das tarefas.

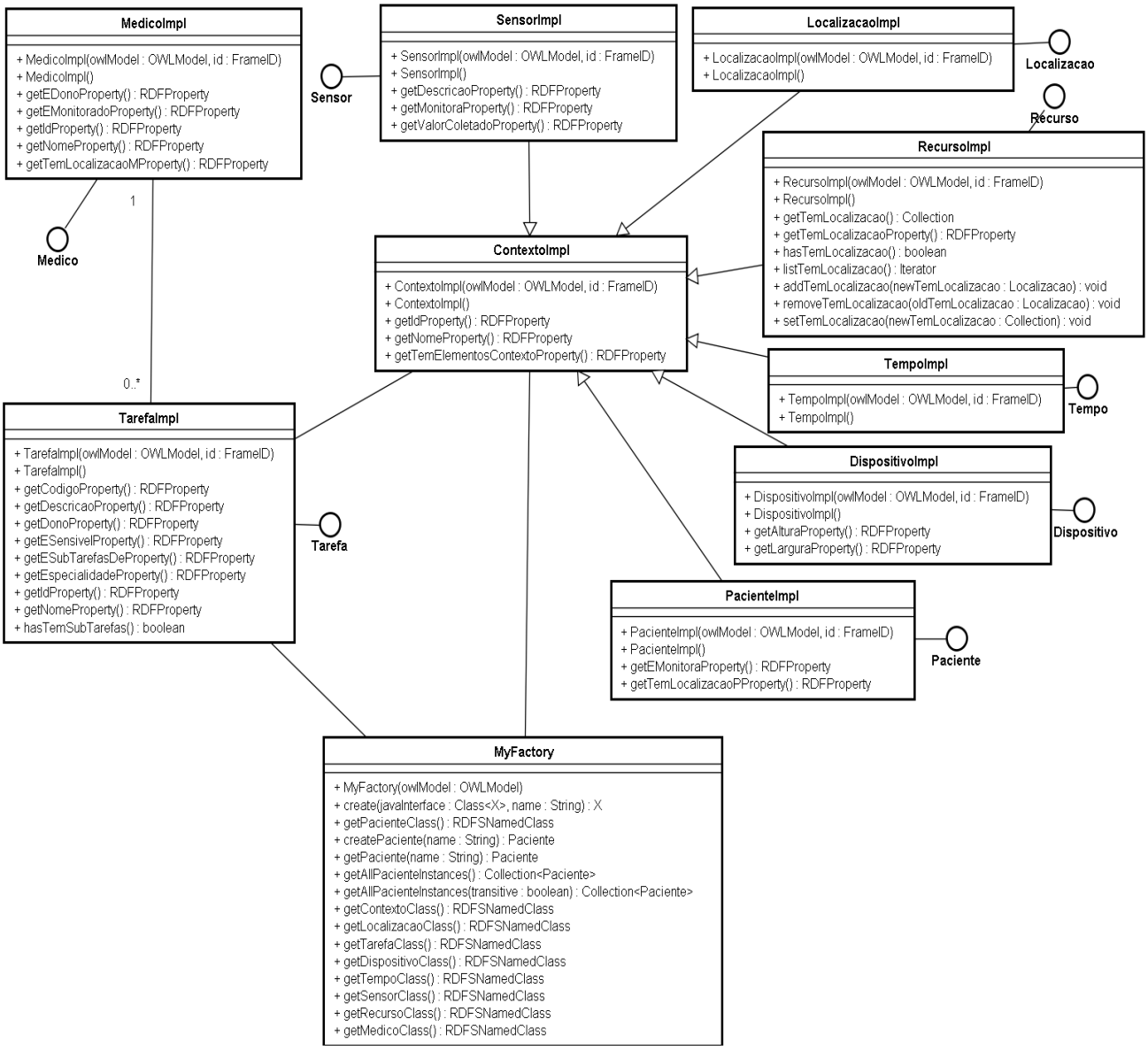

Figura 26: Diagrama UML do novo SAT

4.3.4 Serviço de Contexto para Tarefas

As alterações no SCT se deram devido ao atual modelo ontológico utilizado, pois agora ele recebe um objeto `Contexto` carregado pelo SAT contendo o contexto de interesse para a execução da tarefa.

Quando o SGT solicita para ele (SCT) os dados de um `Contexto`, tal serviço realiza o link correspondente às informações de contexto necessárias para a execução das tarefas. O SCT recebendo um `Contexto` precisa identificar quais elementos de contexto existem neste contexto; após, realiza a conexão com seu banco de dados específico. Por exemplo, o SCT recebe um `Contexto` que tem o elemento de contexto `Paciente`. Neste momento, o SCT já está com o ID do paciente e as informações de quais sensores estão vinculados a este paciente (estas informações foram carregadas pelo SAT). Para receber as informações dos sensores, o SCT precisa subscrever-se (*publish/subscribe)* no SubSistema de Reconhecimento do Contexto e Adaptação do EXEHDA informando os sensores do paciente; assim, este subsistema do EXEHDA vai informar ao SCT (que agora se tornou consumidor das informações dos sensores) as informações coletadas dos sensores. Para carregar os dados do paciente (histórico de consultas e outras), o SCT faz uma chamada a um método do pEHS informando o paciente e recebendo um outro objeto `Paciente` contendo essas informações.

Existindo elementos de contexto `Dispositivo` no contexto, o SCT precisa realizar uma conexão ao banco de contexto para carregar as características do dispositivo, para que a tarefa também seja sensível ao dispositivo.

Tal agregação dos dados, tornando-os informações de contexto, é fundamental pelo seguinte motivo:

> Não é viável projetar uma aplicação móvel para cada ambiente possível e encarregar o usuário de selecionar e ativar a aplicação adequada ao ambiente corrente. É necessário um comportamento adaptativo – consciente do contexto - da própria aplicação. Codificar uma aplicação pervasiva é especificar qual código deve executar no ambiente em que a aplicação está no momento. Deste objetivo deriva a necessidade de adaptação dinâmica ao contexto (AUGUSTIN; LIMA; YAMIN, 2006, p. 7).

4.3.5 Banco de Tarefas (BT)

O Banco de Tarefas é um repositório localizado na Base de Dados da Aplicação (BDA), existente no *Middleware* EXEHDA, contendo a modelagem ontológica das 11 tarefas básica (seção 3.2). Toda vez que é criado um usuário novo na arquitetura ClinicSpace, é gerado um Ambiente Virtual para este usuário (AVU) no EXEHDA. A partir deste momento, as 11 tarefas básicas são copiadas para o AVU do usuário e modificadas de acordo com as solicitações feitas usando a ferramenta IETC.

Este Banco de Tarefas é um diretório gerenciado pelo EXEHDA onde consta o arquivo OWL modelado pelo usuário (médico) da aplicação.

4.3.6 Banco de Contexto (BC)

O Banco de Contexto é um banco de dados relacional que contém o cadastro das informações referentes às entidades de contexto. Neste banco são armazenadas as informações vinculadas a um domínio específico. Por exemplo, quando um paciente dá entrada no hospital é acessado a Base de Dados Pervasiva, através do pEHS, e carregado os dados do paciente para o banco de contexto. Assim, o contexto é específico do domínio onde o médico atua. Diferente da Base de Dados Pervasiva que deve ser acessível em qualquer lugar e a qualquer momento, independente de domínio.

O Banco de Contexto está sendo objeto de estudo do mestrando Vinicius Maran.

Considerações Finais

A modelagem apresentada é baseada nos resultados coletados no capítulo 2; desta forma, foram adicionadas as melhores formas de modelagem de contexto apresentada, buscando uma melhor representação e processamento do contexto associado às tarefas clínicas, sempre levando em consideração a personalização do usuário.

5 RESULTADOS E DISCUSSÕES

Este capítulo apresenta uma avaliação comparativa com os trabalhos relacionados e o estudo de caso gerado para testes na arquitetura ClinicSpace.

5.1 Avaliação Comparativa dos Trabalhos Relacionados

O conceito de Computação Baseada em Tarefas foi introduzido pelo Projeto Aura (SOUSA; GARLAN, 2002). Esse *middleware* é pró-ativo, ou seja, não há a programação de aplicações envolvidas por parte do usuário, o que leva ao aumento da interferência do sistema no ambiente. Considera-se que os usuários tendem a rejeitar sistemas totalmente automatizados em ambientes hospitalares. Já no projeto Gaia (ROMAN, 2002) visualiza-se um futuro onde o espaço habitado pelas pessoas é interativo e programável. Assim, os usuários podem interagir com seus escritórios, casas e carros, para requisitar informações, beneficiarem-se dos recursos disponíveis, e configurar o comportamento de seu habitat. Porém, esse sistema visa elementos integrantes de um ambiente de sala-de-aula e campus (educação). O projeto Task Computing (SONG; LABROU; MASUOKA, 2004) disponibiliza um *framework* que visa capacitar o usuário a executar tarefas em aplicações, dispositivos e serviços em ambientes pervasivos, o foco principal está na infra-estrutra de suporte à execução o gerenciamento dos serviços. Na área de Saúde, o projeto *Activity-Based Computing* (BARDRAM; CHRISTENSEN, 2007) apresenta uma proposta para utilização de Computação baseada em Tarefas destinada ao Ambientes de Saúde.

	Suporte às Tarefas			Ambiente Pervasivo		
	Controle	**Automação**	**Personalização**	**Mobilidade**	**Contexto**	**Contexto Interesse**
AURA		**X**		**X**	**X**	
GAIA				**X**	**X**	
ABC	**X**	**X**		**X**	**X**	
Task Computing	**X**	**X**			**X**	
ClinicSpace	**X**	**X**	**X**	**X**	**X**	**X**

Tabela 4: Comparativo entre Projetos Relacionados

Como se observa na tabela 4, nenhum desses projetos apresenta a visão centrada no usuário final (médico) defendida pelo projeto ClinicSpace. Este, além de permitir a modelagem das atividades diárias realizadas pelo próprio usuário e a possibilidade de associação do contexto de interesse do médico (proposto neste trabalho), também utiliza sensibilidade ao contexto de uma forma robusta, utilizando informações estáticas (pEHS) e dinâmicas (sensores) para melhor detectar as mudanças de contextos durante a realização de uma atividade. Espera-se, desta forma, reduzir a rejeição dos sistemas computacionais em atividades clínicas hospitalares.

5.2 Estudo de Caso

O estudo de caso realizado neste trabalho partiu da descrição de uma situação-problema que descreve um caso de uso da IETC, desta forma procurou-se avaliar a solução desenvolvida. Para tal, foi utilizado o protótipo disponível do Subsistema de Gerenciamento Distribuído de Tarefas, (SGDT) contendo o Serviço de Gerenciamento de Tarefas (SGT), o Serviço de Acesso a Tarefas (SAT) e o Serviço de Contexto para Tarefas (SCT). Esses serviços foram integrados ao EXEHDA e instanciados em dois nodos: base e cliente. Foram feitos testes a partir do carregamento da ontologia pelo SAT. A partir da análise dos dados foram identificados gargalos de desempenho e, com isso, foram realizadas as correções necessárias para que o processamento do sistema interfira o mínimo possível no desempenho das atividades pelos usuários.

5.2.1 Situação Problema

Considere-se a seguinte situação: um médico trabalha em um hospital que tem um sistema de gerenciamento computacional atuando de forma pervasiva e ubíqua. O médico chega ao hospital e tem agendado uma rotina diária de atividades para realizar, como o atendimento a pacientes em seu consultório, visita (ronda) de rotina a seus pacientes baixados para acompanhamento, cirurgia em um determinado paciente. Cada atividade tem uma rotina específica para ser realizada, a qual pode ser decomposta em um fluxo de subatividades.

A atividade de atendimento a pacientes pode ser decomposta em subatividades, como: (i) verificação dos sinais vitais, tais como pressão, batimentos cardíacos e temperatura; (ii) verificação do histórico de saúde, identificando alergias e enfermidades anteriores; (iii) análise de exames laboratoriais; (iv) diagnóstico; (v) prescrição de receita.

O paciente, quando dá entrada no hospital, é identificado pela recepção e sensores físicos são acoplados ao seu corpo, para que suas funções vitais sejam monitoradas e sua localização e identificação sejam descobertas dinamicamente pelo sistema de informação do hospital. Recursos computacionais são utilizados pelo médico na execução de suas atividades, tais como computadores pessoais e dispositivos móveis.

5.2.2 Ambientes de Testes

O Ambiente de testes foi configurado gerando um nodo base do EXEHDA onde são mantidas as descrições ontológicas das tarefas (acessadas pelo SAT) e Banco de Contexto. O nodo cliente é usado pelo usuário para executar suas tarefas.

Para simular o funcionamento dos serviços foram implementadas as tarefas mencionadas na situação-problema. Usando-se essas tarefas, foi modelada uma tarefa de atendimento a um paciente usando a IETC. Tal tarefa recebeu um contexto contendo um dispositivo e um paciente com os sensores de pressão, batimentos cardíacos e temperatura. Esta tarefa recebeu um dono (médico).

Sendo assim, os testes de carga foram realizados na medição do tempo que o SAT utilizou para realizar o carregamento da ontologia. Iniciou-se com a ontologia contendo somente uma estrutura de indivíduos (médico, tarefa e contexto). O arquivo OWL atingiu o

tamanho de 13KB. Buscando-se gerar uma ontologia com um tamanho significativo para verificar o impacto no ClinicSpace, foi gerado outros arquivos até chegar em um arquivo OWL com média de 5MB, resultando em 640 médicos e suas relações (tarefas e contexto). Assim, podem ser medido o tempo de carregamento do novo SAT em um hospital contendo uma média de 640 médicos.

Os equipamentos usados nas simulações foram:

- EXEHDAbase:
  - Processador: Intel Core 2 Duo;
  - Memória: 2 GB;
  - Java versão 1.6.0_18.
- EXEHDAnodo:
  - Processador: AMD 2.20 GHz;
  - Memória: 2 GB;
  - Java versão 1.6.0_18.
- Rede Ethernet 10/100 Mbps.

5.2.3 Programação de Tarefas pelo Médico

Quando o médico inicia o uso do ClinicSpace, via Interface de Edição de Tarefas e Contexto (IETC), é montada a interface com base nas suas tarefas já programadas, como ilustra a Figura 28.

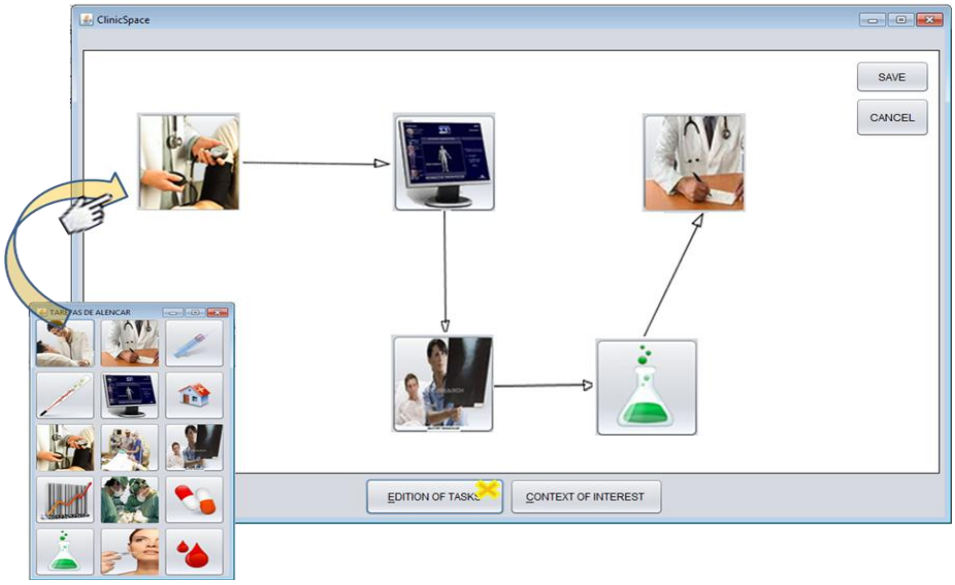

Figura 27: Modelagem da Tarefa

Como apresenta a figura 27 na IETC, foi modelado a tarefa de testes contendo as cinco subtarefas descritas na situação-problema: (i) verificação dos sinais vitais; (ii) verificação do histórico de saúde; (iii) análise de exames laboratoriais; (iv) diagnóstico; (v) prescrição de receita.

Nesta, para a atividade de A*tendimento ao Paciente*, o médico modela o fluxo de tarefas necessárias, conforme a sua forma particular de realizá-la. Para programar a funcionalidade desejada, o médico X seleciona o ícone correspondente à subtarefa desejada, arrasta-o para a área de edição, e associa-lhe as entidades de contexto, por exemplo *paciente* e *dispositivo*. Para programar os demais procedimentos, a metodologia é a mesma. O médico informa também qual é o fluxo de execução, criando associações entre as tarefas, ou seja, ligando uma à outra.

5.2.4 Associando Contexto às Tarefas

Na Interface de Edição de Tarefas e Contexto (IETC) é disponibilizada, para o médico, a opção de associação de contexto de interesse a uma tarefa.

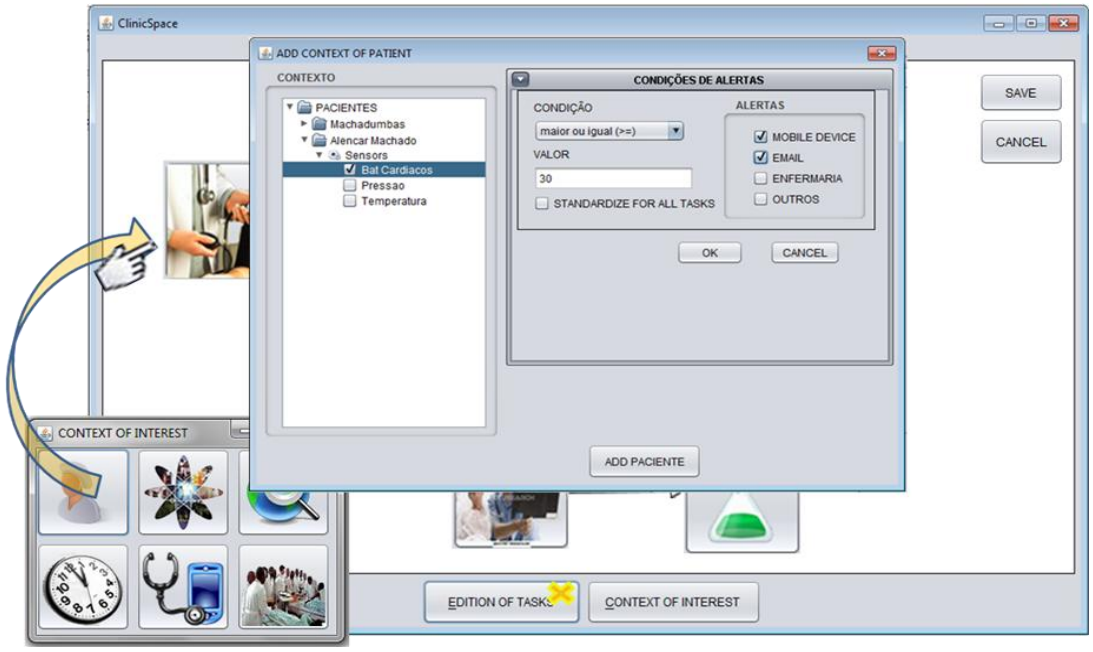

Figura 28: Modelagem do Contexto de Interesse

Na situação-problema, conforme ilustrado na Figura 28, o médico associa a entidade de contexto *paciente* à tarefa de *Atendimento ao Paciente*.

A IETC, então, apresenta os pacientes vinculados ao perfil do médico. Para cada paciente é apresentado um elemento de contexto específico, modelado anteriormente, com os sensores que estão realizando o monitoramento de funções vitais.

Para o ambiente de testes foi adicionado um elemento de contexto `Paciente` contendo os sensores de pressão, batimentos cardíacos e temperatura. Também foi adicionado o elemento de contexto dispositivo contendo o indivíduo `celularE71`, assim informando o dispositivo ao qual a tarefa deverá se adaptar em tempo de execução.

5.2.5 Descrição Ontológica da Tarefa e Contexto

A relação de indivíduos gerados, após a modelagem da tarefa pela IETC, é ilustrada na figura 30, onde constam as relações entre os indivíduos e seu respectivo dono.

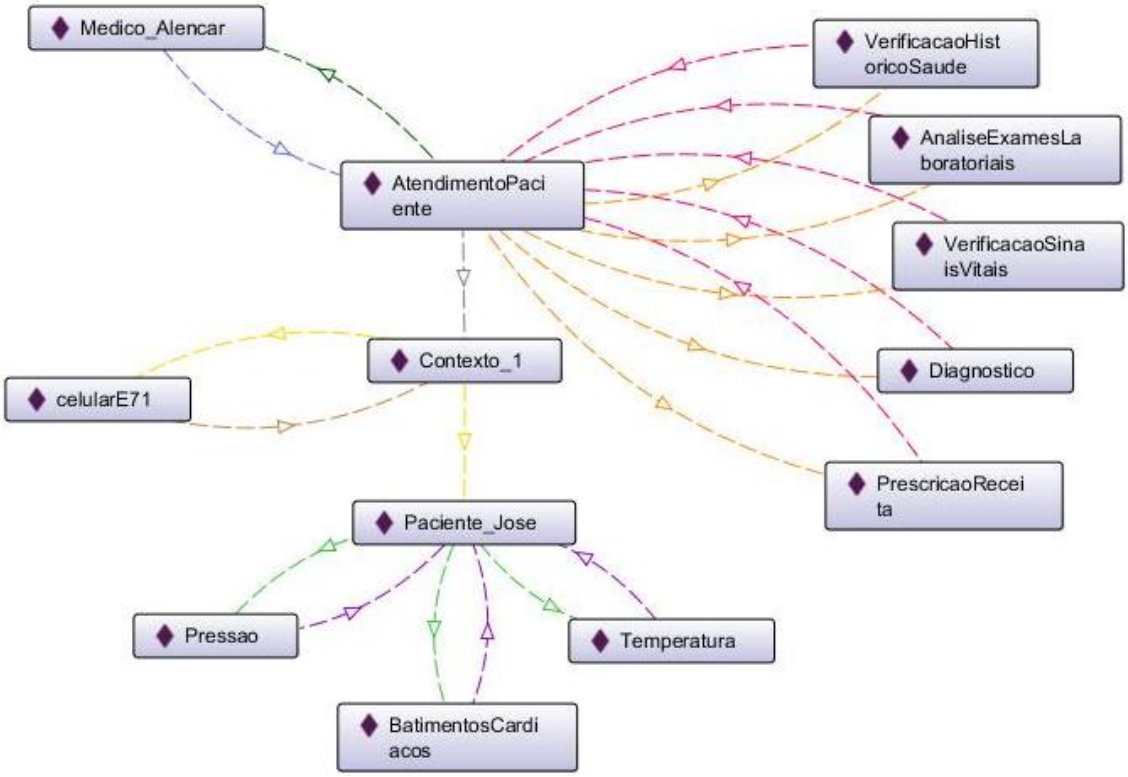

Figura 29: Representação gráfica das instâncias e relacionamentos contidos em uma Tarefa.

A figura 29 ilustra, de forma gráfica, uma instância de indivíduos da ontologia onde o indivíduo *Medico_Alencar* é dono de um tarefa que contém cinco subtarefas, as quais, respectivamente, contém um contexto vinculado.

Esta estrutura de indivíduos foi replicada 640 vezes para verificação do impacto desta ontologia no carregamento do SAT.

5.2.6 Suporte Computacional à Realização das Atividades

Como visto anteriormente, para simular o funcionamento do Subsistema de Gerenciamento Distribuído de Tarefas (SGDT), o EXEHDA (com os novos serviços integrados) foi configurado e inicializado em dois nodos: um nodo base, responsável pelo gerenciamento da célula, e um nodo comum (cliente), representando um dispositivo de usuário, o qual faz parte da célula.

Dessa forma, a aplicação do usuário (IETC) é iniciada e, através da API do SGDT, busca as tarefas disponíveis ao usuário. Como essas informações estão localizadas no nodo

base, a instância local do serviço SAT gera requisições à respectiva instância remota.

Por isso, a operação desse serviço foi monitorada na simulação para determinar seu impacto no sistema. Assim, foi analisado o tempo para inicialização das tarefas. Esse tempo foi avaliado de forma absoluta, uma vez que relacioná-lo com o tempo total de execução da tarefa se torna impreciso, já que cada usuário realiza suas tarefas de forma diferente e em tempos variados.

Quando uma tarefa é iniciada, por vontade do médico ou por gatilho de variação em um elemento de contexto, são obtidas as informações de perfil do médico e sua localização. Na realização da tarefa *Atendimento ao Paciente*, telas específicas vinculadas com cada subtarefa são apresentadas de forma pró-ativa para simplificar a execução, exigindo menos interferência e atenção do médico em termos de controle do sistema computacional. Assim, uma definição ontológica da tarefa é carregada e as subtarefas são executadas.

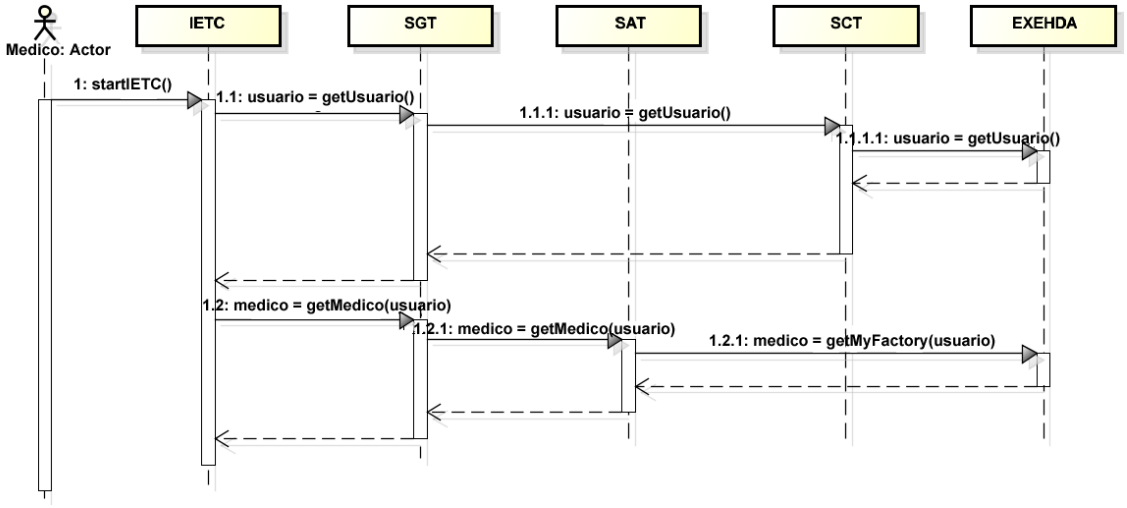

Figura 30: Diagrama de Seqüencia do carregamento da ontologia pelo SAT

Inicialmente, como apresenta o diagrama de sequencia (figura 30), quando o médico vai utilizar a IETC, a mesma faz uma chamada ao SGT para verificar qual usuário está iniciando a execução. Obtido o usuário, é feito uma chamada ao método *getMedico*(*usuario*) no SGT que se comunica com o respectivo serviço do SAT para tal carregamento, passando o nome do usuário para carga da respectiva ontologia dentro do *Ambiente Virtual do Usuário* (AVU) no EXEHDA (*getMyFactory(usuario)*). Sendo assim, após o carregamento, uma instância do médico contendo toda estrutura de tarefas, subtarefas e contexto é retornada para a IETC possibilitando sua utilização pelo médico.

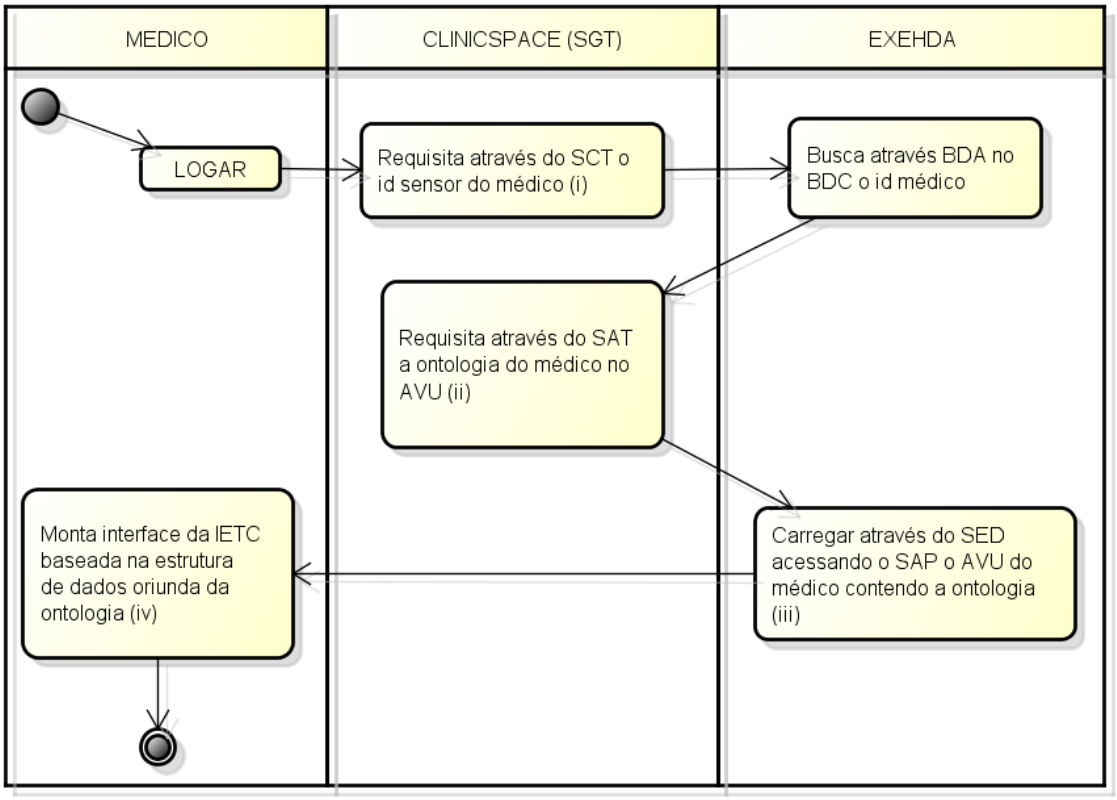

Figura 31: Diagrama de Atividades do carregamento das tarefas

O diagrama de Atividades (Figura 31) ilustra a integração da IETC, serviços da arquitetura ClinicSpace e os subsistemas de serviços EXEHDA para a realização do carregamento da ontologia pelo médico.

O Diagrama de Atividade está separado em MEDICO (nó cliente), CLINICSPACE e EXEHDA (nó base) para demonstrar a interação entre as partes na atividade realizada.

Quando o médico chega perto de um terminal, o ClinicSpace identifica automaticamente ou formalmente através de login e senha no IETC. O (i) SCT faz uma chamada na *Base de Dados da Aplicação* (BDA) onde está o *Banco de dados de Contexto* (BDC) requisitando os dados do usuário. O (ii) SAT contendo a informação de quem é o médico logado, faz uma chamada ao *Subsistema de Execução Distribuída* (SED), o qual acessa os serviços do *Subsistema de Acesso Pervasivo* (SAP) para realizar o carregamento do *Ambiente Virtual* do *Usuário* (AVU) contendo a ontologia (arquivo OWL).

Após o carregamento do arquivo OWL pelo SAT, o qual fez a leitura das instâncias e suas relações transformando-os em objetos Java para manipulação pela IETC, (iii) toda estrutura das tarefas do médico está na memória do nodo base do EXEHDA e é transportado, através do *Subsistema de Execução Distribuída* (SED), até o nodo cliente onde está o médico. Desta forma, (iv) a IETC faz a leitura da estrutura da tarefa e seu contexto, remontando os

elementos na interface gráfica, sendo que cada subtarefa é modelada na tela como um retângulo e associa-se a imagem salva na sua criação.

A partir deste momento, estão disponíveis todas as tarefas ativas e modeladas pelo médico, levando em consideração o contexto de interesse informado.

5.2.7 Resultados e Discussões da Avaliação

Para avaliação do Serviço de Acesso a Tarefas foram realizados testes do carregamento da ontologia, iniciando com um indivíduo médico contendo as relações descritas na seção 5.2.5. Após a primeira medição foi replicado em dez (10) vezes a estrutura existente, seguindo-se medições no número de réplicas: vinte (20), quarenta (40), oitenta (80), cento e sessenta (160), trezentas e vinte (320) e seiscentas e quarenta (640). Assim é possível calcular o impacto que uma ontologia contendo a estrutura de modelagem de tarefas e contexto para uma média de 640 médicos pode impactar no desempenho da arquitetura ClinicSpace.

A primeira fase de testes foi realizada com o EXEHDA sempre partindo do zero, ou seja, sempre iniciando uma nova instância do *middleware* a cada arquivo OWL carregado pelo SAT.

MEDICO (qtd)	**TAM (KB)**	**TEMPO (ms)**
1	13	2654
10	82	2924
20	161	3194
40	316	3505
80	613	4875
160	1248	4890
320	2431	6361
640	4971	7981

Tabela 5: Testes no SAT para carregamento da Ontologia sempre em uma "nova" instância do EXEHDA

A tabela 5 descreve em detalhes a quantidade de médicos, o tamanho da ontologia em KB e o tempo em milissegundos utilizado para o SAT carregar a ontologia.

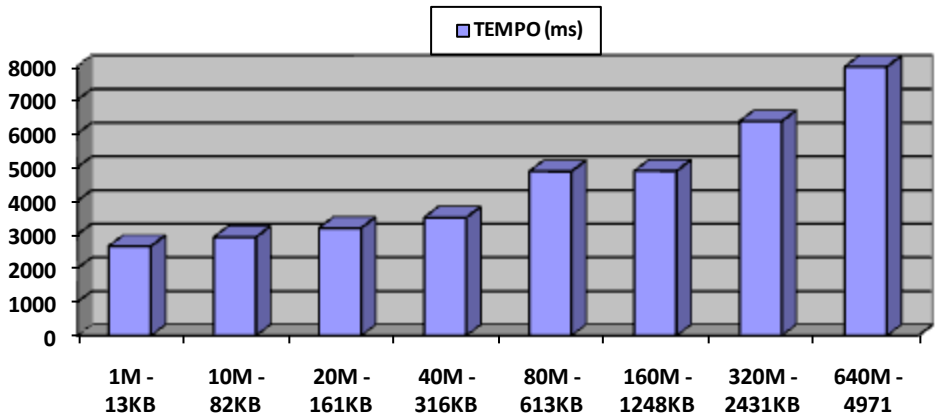

Figura 32: Gráfico do tempo médio para inicialização do SAT:

O gráfico (figura 32) apresenta o tempo gasto para o carregamento pelo SAT e o conjunto de oito (8) diferentes tamanhos de arquivos OWL. Por exemplo, na primeira medição, o arquivo contendo uma instância da ontologia, que corresponde a um médico (1M – 13KB), descrita na 5.2.5 expressa num arquivo OWL de 13 KB demorou para ser carregada 2654 ms.

Nota-se que o tempo inicial para o carregamento da ontologia pelo SAT é em torno de 2,5 segundos - um custo inicial. O gráfico mostra que, mesmo aumentando 383 vezes o tamanho da ontologia, o tempo para carga ficou em 7,9 segundos, mostrando que a ontologia pode ser extremamente grande e mesmo assim ter um custo inicial e final aceitável, porém perceptível ao usuário final.

Considerando que os tempos apresentados não foram satisfatórios, iniciou-se outra coleta de dados, mantendo o EXEHDA sempre em execução. Desta forma, foram carregadas todas as ontologias na mesma instância do EXEHDA. A tabela 6 apresenta os dados obtidos.

MEDICO (qtd)	TAM (Kb)	TEMPO (ms)
1	13	2764
10	82	559
20	161	448
40	316	742
80	613	763
160	1248	812
320	2431	1499
640	4971	2427

Tabela 6: Testes no SAT para carregamento da ontologia

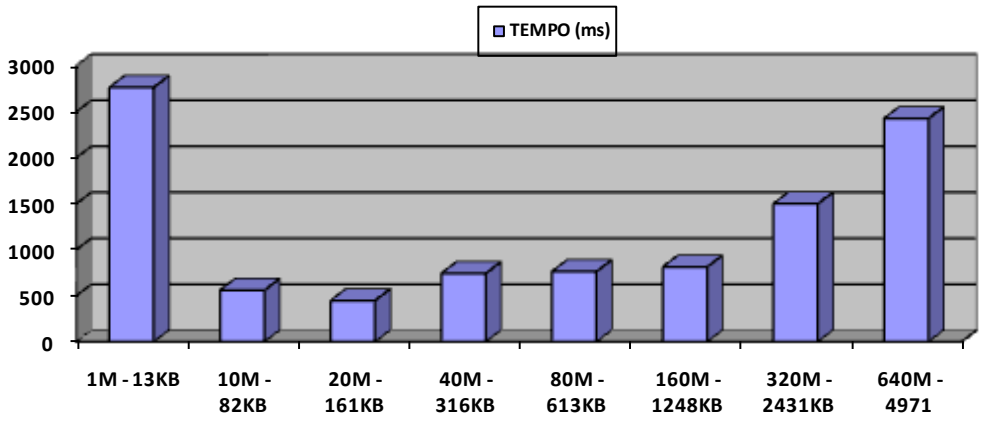

Figura 33: Gráfico do tempo médio para inicialização do SAT na mesma instância do EXEHDA

O gráfico (figura 33) apresenta os resultados obtidos com a mesma instância do EXEHDA, ou seja, foi inicializado o EXEHDA uma única vez, depois as ontologias foram carregadas sob demanda dos nós clientes, onde cada índice do gráfico foi uma solicitação de um nó cliente à mesma instância do EXEHDA. Nota-se que o custo inicial de carregamento da ontologia permanece. No entanto, o próximo carregamento diminui em cinco (5) vezes o valor inicial, apesar de a ontologia carregada ser seis (6) vezes maior que a inicial. O último carregamento, mesmo sendo realizado por um arquivo trezentas e oitenta e três (383) vezes maior que o arquivo inicial, ainda ficou com 337ms abaixo do carregamento inicial da ontologia. Esses dados levam à conclusão que a melhor abordagem é sempre manter o EXEHDA em execução.

Melhorando a Solução

Procurando melhorar o tempo de carregamento da ontologia, buscou-se a separação de cada instância do indivíduo médico com suas tarefas e contextos correspondentes, portanto, gerando uma ontologia para cada médico. Esta ontologia é copiada para o Ambiente Virtual do Usuário (AVU) quando um médico passa a existir como usuário do ClinicSpace. Desta forma, sempre que o médico vai modelar ou executar suas tarefas, somente neste momento o ClinicSpace faz o carregamento da ontologia correspondente ao médico, evitando o desperdício de tempo com a carga de partes de uma ontologia que talvez não seja utilizada.

Para testar essa solução, foram geradas 10 ontologias do mesmo tamanho, simulando 10 usuários utilizando a ferramenta. Levando-se em conta o último resultado, foi mantido o EXEHDA em execução e instanciado pelos nós clientes dez (10) aplicações, carregando dez (10) ontologias semelhantes, porém sendo arquivos OWL diferentes. Os dados coletados estão

descritos na tabela 7.

MEDICO (qtd)	TEMPO (ms)
1	2751
2	320
3	223
4	129
5	117
6	96
7	110
8	192
9	91
10	102

Tabela 7: Testes no SAT para carregamento da Ontologia sempre na "mesma" instância do EXEHDA em arquivos OWL de 13KB.

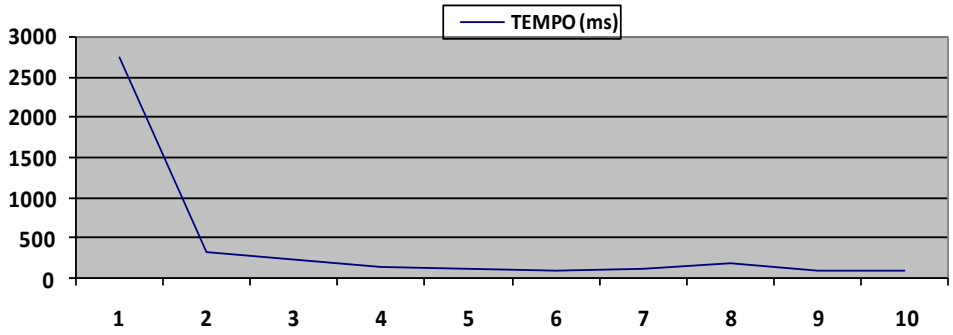

Figura 34: Gráfico do tempo médio para o carregamento de 10 arquivos OWL com 13KB

O gráfico (figura 34) apresenta os resultados obtidos após a separação e inicialização de cada ontologia de acordo com cada usuário. Nota-se que o custo inicial de carregamento na primeira vez da ontologia é sempre o mesmo. Neste momento, foi constatado que as bibliotecas do Protégé utilizadas inserem um custo inicial e, após sua primeira execução, mantém os objetos estáticos na memória. Após o primeiro carregamento, os demais se mantiveram oscilando entre 320ms e 91ms, mostrando que a abordagem de gerar uma ontologia por usuário é a mais satisfatória.

Segundo Bettinia et al. (2010), a modelagem ontológica de contexto provê claras vantagens em termos de expressividade e interoperabilidade, porém problemas de desempenho quanto ao processamento com OWL-DL já foram confirmados por avaliações

experimentais de utilização do contexto em diferentes ontologias baseadas em arquiteturas.

Assim, verifica-se que a execução on-line da ontológica apresenta problemas de escalabilidade, especialmente quando a ontologia é povoada por um grande número de indivíduos. Os testes realizados confirmam esse problema.

6 CONCLUSÕES

A Computação Ubíqua/Pervasiva aplicada a hospitais torna esses ambientes mais inteligentes e centrados no usuário final. Para tanto, é necessário pesquisas em diferentes áreas, como modelagem de contexto, sensibilidade ao contexto e programação orientada a atividades. O projeto ClinicSpace constrói uma infra-estrutura para que o médico possa tornar o sistema o mais próximo possível da sua realidade e necessidade (personalização) e, assim, espera-se diminuir a rejeição desses profissionais em relação à utilização de sistemas computacionais para o auxílio nas suas tarefas diárias. Visando a construção de uma arquitetura para utilização em um hospital do futuro, são adicionados os conceitos de Computação Ubíqua procurando reduzir a interferência direta da computação nas atividades diárias dos médicos.

A modelagem de contexto está listada entre os principais desafios vinculados à Computação Ubíqua, a qual está ainda caminhando para a construção de padrões que a viabilizem em sua ampla utilização. A definição de como modelar sistemas ubíquos se torna necessário para esse fim. Transformar sistemas cientes dos elementos integrantes do ambiente, que influenciam as decisões do usuário e da aplicação, tornando-os mais inteligentes e pró-ativos, auxilia a tomadas de decisões de forma mais coerente e precisa.

A construção e disponibilização de sistemas sensíveis ao contexto envolvem diversas camadas de software, estruturas e conceitos, contribuindo para aumentar a complexidade desses sistemas.

A identificação dos elementos de contexto que influenciam em uma atividade clínica se tornou necessário no projeto da arquitetura ClinicSpace, pois não é viável que sistemas sensíveis ao contexto gerenciem tudo que se encontra em um ambiente (contexto). Assim, identificou-se alguns elementos de contexto (paciente, dispositivo, recursos, sensor, tempo, localização) relevantes para a arquitetura ClinicSpace. Porém, não se esgota esta identificação, pois o modelo de contexto genérico proposto para ambientes clínicos tem o objetivo de ser extensível.

A modelagem de Tarefas Clínicas e associação de Elementos de Contexto à Interface de Edição de Tarefa e Contexto (IETC) visam criar abstrações de alto nível que melhor

representem as atividades executadas diariamente pelos profissionais, no cuidado de seus pacientes. Assim, os médicos podem adicionar suas próprias preferências, experiência e conhecimentos para a criação, definição e personalização das tarefas.

A utilização de ontologias para estruturação e descrição das tarefas e contexto mostrou-se satisfatória. Para buscar melhores resultados no processamento da ontologia, tomou-se a decisão de gerar uma ontologia por médico. Foi necessário uma re-implementação do Serviço de Acesso a Tarefas para adicionar o suporte ao carregamento das ontologias. Com a utilização das bibliotecas do WebProtégé foi possível disponibilizar para a IETC toda estrutura de processamento ontológico necessário para sua modificação. Este novo serviço apresentou resultados aceitáveis e permite que trabalhos futuros de inferências na execução das tarefas e nos estados contextuais possam ser criados.

6.1 Contribuições

As contribuições do trabalho para a arquitetura ClinicSpace foram:

- Modelagem do contexto - identificação dos elementos de contexto clínico que a arquitetura deve ser sensível;
- Contexto Ontológico - criação da ontologia das tarefas e do contexto clínico, pois tal parte da arquitetura estava sendo realizada por *parses* XML. Sendo assim, desenvolveu-se o modelo ontológico proposto e a implementação das alterações necessárias no Serviço de Acesso a Tarefas, Serviço de Contexto para Tarefas e Serviço Gerenciador de Tarefas, assim permitindo a associação dos elementos de contexto modelados;
- Interface gráfica de associação dos elementos de contexto de interesse - objetiva permitir ao médico definir a informação de quais elementos de contexto são de seu interesse na realização de uma determinada tarefa, diminuindo a carga existente nos gerenciadores de contexto e na implementação da tarefa.

6.2 Trabalhos Futuros

Os trabalhos futuros propostos na arquitetura abrangem:

- compartilhamento de tarefas entre outros médicos, para que seja possível a interrupção de uma tarefa e delegação da mesma para outro médico até o seu término (procurando resolver as questões de troca de médicos no atendimento de um paciente);
- pesquisa de mais entidades/elementos de contexto para que o poder de sensibilidade da tarefa ao contexto aumente gradativamente com cada elemento adicionado;
- elaboração de regras de inferências na ontologia, assim permitindo melhorar a usabilidade da IETC, bem como o processamento das tarefas;
- Testes de usabilidade embasados nos conceitos de *Quality of Experience* (QoE) em um ambiente real, buscando detectar um grau de qualidade na experiência do usuário com o uso da arquitetura ClinicSpace, identificando melhorias necessárias na interface de modelagem de tarefas e contexto, procurando refiná-la para que se torne intuitiva para o público-alvo (médico).

6.3 Publicações

As publicações até o momento relativas ao trabalho são:

UBICOMM 2010: *"Applying a middleware for pervasive environment to manage end-user programming of clinical activities"*. In: The Fourth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies UBICOMM, 2010, Florence - Italia. UBICOMM, 2010.

CLEI 2010: *"Ciência do Contexto para Tarefas Clínicas em um Sistema de Saúde Pervasivo"*. In: XXXVI Conferência Latino-americana de Informática - CLEI, Assuncion – Paraguai, 2010.

SBCUP 2010: *"Ferramenta para Definição de Contexto pelo Usuário-Final na Programação de Tarefas Clínicas em um Sistema de Saúde Pervasivo".* In: Congresso da SBC - Simpósio Brasileiro de Computação Ubíqua e Pervasiva, 2010, Belo Horizonte - MG.

RBCA 2010: *“EHS Arquitetura de um Sistema de Informação Pervasivo para Auxílio às Atividades Clínicas”.* In. Revista Brasileira de Computação Aplicada, v. 2, p. 2, Série: 2; ISSN/ISBN: 21766649; 2010.

RITA xxx: *“Sistema Pervasivo de Informação em Saúde Projetado para ser Programado pelo Usuário Clínico”.* In: Revista de Informática Teórica e Aplicada, 2010. *(submetido para publicação)*

SIRC 2009: *“Requisitos de um Registro Eletrônico de Saúde Ubíquo”*. In: VIII Simpósio de Informática da Região Centro do Rio Grande do Sul, 2009, Santa Maria. Requisitos de um Registro Eletrônico de Saúde Ubíquo, 2009.

REFERÊNCIAS

AUGUSTIN, I., YAMIN, A., NASCIMENTO, E., BARBOSA, J.L.V, Cavalheiro, G., e Geyer, C. **ISAM: um Middleware para Aplicações Móveis Distribuídas**. *RITA – Revista de Informática Teórica e Aplicada*. VIII, num 2, 2001.

AUGUSTIN, I.; LIMA, JOÃO CARLOS D. and YAMIN, A. C. 2006. **Computação Pervasiva: como Programar Aplicações**. In: X SIMPOSIO BRASILEIRO DE LINGUAGENS DE PROGRAMAÇÃO (SBLP), 2006, Itatiaia, RJ. **Anais...** [S.l.]: SBLP, 2006.

AUGUSTIN, I.; YAMIN, A.; GEYER, C. F. R. ***Managing the follow-me semantics to build large-scale pervasive applications***. In: Proceedings of the 3rd International Workshop on Middleware for Pervasive and Ad-Hoc Computing, 2005, France, p. 1-8.

BARDRAM, J. E. ***Hospitals of the Future***: Ubiquitous Computing Support for Medical Work in Hospitals. In: Proceedings of the 5th International Conference in Ubiquitous Computing, 2003, USA.

BARDRAM, JAKOB E. and CHRISTENSEN, Henrik B. 2007**. Pervasive Computing Support for Hospitals: An overview of the Activity-Based Computing Project**. IEEE Pervasive Computing, vol. 6, issue 1, p. 44-51, 2007.

BETTINIA, C., BRDICZKAB, O., HENRICKSEN, K., INDULSKAD, J., NICKLASE, D., RANGANATHANF, A., RIBONI, D.: **A survey of Context Modelling and Reasoning Techniques. Pervasive and Mobile Computing**, 6(2), 161-180 (2010)

BOUQUET, P., GIUNCHIGLIA, F., HARMELEN, V, L. SERAFIINI, H. STUCKENSCHMIDT**, Contextualizing Ontologies**, Journal of Web Semantics 1 (4) (2004)

CHEN, H., PERICH, F., FININ, T. W., JOSHI, A., SOUPA: **Standard Ontology for Ubiquitous and Pervasive Applications,** in: 1st Annual International Conference on Mobile and Ubiquitous Systems (MobiQuitous 2004), IEEE Computer Society, 2004.

DEY, A., and ABOWD, G., **The Context Toolkit: Aiding the Development of Context-Aware Applications**, In Proceedings of Human Factors in Computing Systems: CHI 99, Pittsburgh, PA: ACM Press, pp.434-441. (2006)

FERREIRA, Guiliano G. L. **Adicionando ao Middleware Exehda o Suporte a Aplicações Orientadas a Atividades Humanas**. 2009a. 103f. Monografia (Mestrado em Computação) Universidade Federal de Santa Maria, Santa Maria, RS, 2009.

FERREIRA, Giuliano L.; SILVA, Fábio L.; LIBRELOTTO, Giovani R. and AUGUSTIN,Iara. 2009b. **Middleware for Management of End-user Programming of Clinical Activities in a Pervasive Environment**. In: In: 2009 Workshop on Middleware for Ubiquitous andPervasive Systems, Fourth International Conference on Communication System Software and Middleware. WMUPS 2009.

FERREIRA, Giuliano L.; SILVA, Fábio L.; LIBRELOTTO, Giovani R. and AUGUSTIN, Iara. 2009c. **Adaptando o Middleware EXEHDA para o Tratamento de Atividades Clínicas**. In: XXXV Conferencia Latinoamericana de Informática (CLEI 2009). CLEI –.

GAMMA, E. et al. Design Patterns: **Elements of Reusable Object-Orient ed Software**. *Addison Wesley, USA*, 1995.

GARLAN, D.; STEENKISTE, P.; SCHMERL, B. Project **Aura: Toward Distraction-free Pervasive Computing**. IEEE *Pervasive Computing***,** New York, v.1, n.3, sep. 2002.

HENRICKSEN, K., INDULSKA, J.: Developing **context-aware pervasive computing applications: Models and approach**. Journal of Pervasive and Mobile Computing 2(1), 37–64 (2006)

HENRICKSEN, K., INDULSKA, J. and RAKOTONIRAINY A. **Modeling Context Information in Pervasive Computing Systems**. 1st International Conference on Pervasive Computing. In: Lecture Notes in Computer Science, Volume 2414, pp. 167-180. Springer, 2002.

JHA, A.; DESROCHES, C.; CAMPBELL, E.; DONELAN, K.; RAO, S.; FERRIS, T; SHIELDS, A.; ROSENBAUM, S.; BLUMENTHAL, *D.* **The Use of Electronic Health Records in U.S. Hospitals**. *In:* New England Journal of Medicine. *2009, 360:1628-1638*

LAERUM H, and FAXVAAG, A**. Task-oriented evaluation of electronic medical records systems: development and validation of a questionnaire for physicians**. In: BMC Medical Informatics and Decision Making,1. p. 1-16.3, 2004.

MARKLS, D., ALUR, D., CRUPI, J., **Core J2EE Patterns: Best Practices and Design Strategies** ISBN: 8535212728, 2004.

ROMAN, M.; et al. Gaia: a *Middleware* Infrastructure to Enable Active Spaces. **IEEE *Pervasive Computing***, New York, v.1, n. 4, dec. 2002.

ROMAN M, HESS C., CERQUEIRA R., CAMPBELL and NAHRSTEDT K. **Gaia: a Middleware Infrastructure to Enable Active Spaces**. *In IEEE Pervasive Computing*. New York. 2003. pp. 74-83.

RANGANATHAN, ANAND and CAMPBELL, Roy H. 2005. **Supporting Tasks in a Programmable Smart Home**. In: From Smart Homes to Smart Care, vol. 15. Amsterdam: IOS Press, 2005. p. 3-10.

RIZZETTI, T. ***Framework* para gerenciamento e personalização de contexto orientado a tarefas.** Dissertação de mestrado. Universidade Federal da Santa Maria, Santa Maria, RS, Brasil, 2009.

SIMONS C. CMP**: A UML Context Modeling Profile for Mobile Distributed Systems**. In *Proceedings of the 40th Hawaii International Conference on System Sciences (HICSS'07)*, page 289b, 2007.

SILVA, Fábio L. ; FERREIRA, Giuliano L. ; RIZZETTI, Tiago A. ; AUGUSTIN, Iara ; LIBRELOTTO, Giovani R. and YANIN Adenauer C. 2008 . **Introduzindo a orientação a tarefas clínicas em um middleware de gerenciamento do espaço pervasivo**. In: II Workshop on Pervasive and Ubiquitous Computing. WPUC 2008.

SILVA, Fábio L. **ClinicSpace: Modelagem de uma Ferramenta-Piloto para Definição de Tarefas Clínicas em dicionando ao Middleware Exehda o Suporte a Aplicações Orientadas a Atividades Humanas**. 2009a. 98f. Monografia (Mestrado em Computação) - Universidade Federal de Santa Maria, Santa Maria, RS, 2009.

SILVA, Fábio L.; FERREIRA, Giuliano L.; RIZZETTI, Tiago A.; LIBRELOTTO, Giovani R. and AUGUSTIN, Iara. 2009b. **Ferramenta para a Programação pelo Usuário-Final de Tarefas Clínicas em um Ambiente de Saúde Ubíquo**. In: XXXV Conferencia Latinoamericana de Informática (CLEI 2009). CLEI -

SONG, Z.; LABROU, Y., MASUOKA, R. **Dynamic Service Discovery and Management in Task Computing**. First Annual International Conference Mobile and Ubiquitous Systems: Networking and Services, MobiQuitous'04, p. 310-318,2004.

SOUSA, J. and GARLAN, D. **Aura: An architectural framework for user mobility in ubiquitous computing environments**, The Working IEEE/IFIP Conference on Software Architecture (WICSA) 2002.

SOUZA, Marcos V. B. **Inferência de Atividades Clínicas na Arquitetura ClinicSpace a partir de Propriedades do Contexto**. 2010a. 103f. Monografia (Mestrado em Computação) Universidade Federal de Santa Maria, Santa Maria, RS, 2010.

SOUZA, Marcos V. B., AUGUSTIN, I. **Inferência de Atividades Clínicas na Arquitetura ClinicSpace a partir de Propriedades do Contexto**. In X Workshop de Informática Médica (WIM). Belo Horizonte, MG, Brasil, 2010b.

STRANG, T. and POPIEN, C**. A context modeling survey** In: Proc. of the Workshop on Advanced Context Modeling, Reasoning and Management as Part of UbiComp, pp.33–40. (2005)

TALEB T, BOTTAZZI D, NASSER N (2010) **A novel middleware solution to improve ubiquitous healthcare systems aided by affective information**. IEEE Trans Inf Technol Biomed 14(2):335–349

TURNER, K. J., DOCHERTY, L. S., WANG, F., and CAMPBELL, G. A. 2009**. Managing Home Care Networks**. In Proceedings of the 2009 Eighth international Conference on Networks (March 01 - 06, 2009). ICN. IEEE Computer Society, Washington, DC, 354-359. DOI= http://dx.doi.org/10.1109/ICN.2009.16

UBIHEALTH: 5th International Workshop on Ubiquitous Health and Wellness part os UbiComp 2010 (Setember 26th 2010, Copenhagen, Denmark) http://www.create-net.org/ubint/ubihealth/index.htm. Acesso em 17/11/2010

VARSHNEY, U., **Pervasive Healthcare** In: *IEEE Computer*, vol. 36(12), p. 138-140, 2003.

VICENTINE, C. **PEHS**: Arquitetura de um Sistema Pervasivo de Informação em Saúde Orientado às Atividades Personalizadas pelo Usuário Clínico. Dissertação de mestrado. Santa Maria, RS, Brasil, 2010.

ZHANG, D,. GU T., WANG, X., ENABLING **Context-aware Smart Home with Semantic Technology CONON**, International Journal of Human-friendly Welfare Robotic Systems 6 (4) (2005) 12–20.

YAMIN, A. C. **Arquitetura para um Ambiente de Grade Computacional Direcionado às Aplicações Distribuídas, Móveis e Conscientes do Contexto da Computação Pervasiva.** Tese de Doutorado. Porto Alegre, RS, Brasil, 2004.

WEISER, M. **The Computer of 21st Century**, In: Scientific American, 1991, p 3-11.

WARTENA, F., MUSKENS, J., and SCHMITT, L. 2009. **Continua: The Impact of a Personal Telehealth Ecosystem**. In Proceedings of the 2009 international Conference on Ehealth, Telemedicine, and Social Medicine (DOI= http://dx.doi.org/10.1109/eTELEMED.2009.8

W3C **Composite Capability/Preference Profiles (CC/PP)**: Structure and Vocabularies 2.0 W3C Working Draft 30 April 2007; Acesso em 02/11/2010: http://www.w3.org/TR/2007/WD-CCPP-struct-vocab2-20070430/