Sistema Pervasivo de Informação em Saúde Projetado para ser Programado pelo Usuário Clínico

Alencar Machado[1]
Iara Augustin[2]

Resumo: A Computação Pervasiva está inserida no desenvolvimento de ambientes programáveis e interativos, na busca para ajudar o usuário em suas atividades diárias. Considerando que o sistema de saúde do futuro prevê o uso da Computação Pervasiva para otimizar e automatizar atividades clínicas, esse artigo apresenta a arquitetura ClinicSpace, demonstrando através de uma situação-problema a estrutura da arquitetura para o auxilio às tarefas clínicas (aplicações pervasivas que auxiliam o médico a realizar suas atividades), procurando cumprir os requisitos da computação orientada a atividades. Assim, o ClinicSpace pode ser visto como um sistema ciente do contexto orientado às tarefas clinicas.

Palavras-chave: Computação Ubíqua; Middleware; Computação Orientada a Atividades; Atividades Clínicas; Contexto.

Abstract: The Pervasive Computing is embedded in the development of programmable and interactive environments, and it seeks to help the user in their daily activities. Considering that health system of the future requires the use of Pervasive Computing to optimize and automate clinical activities, this paper presents the architecture ClinicSpace, demonstrating through a problem

[1] UNIPAMPA, Avenida Tiarajú, 810, CEP: 97.546-550, Alegrete, RS
{alencar.comp@gmail.com}
[2] PPGI/UFSM, Avenida Roraima, 1000, CEP: 97.105-900,CT , Santa Maria, RS
{ august@inf.ufsm.br }

situation the structure of the aid architecture for the clinical tasks (pervasive applications that help physicians to perform their activities), looking for fulfill the requirements of activity-oriented computing. So, ClinicSpace can be seen as a context-aware system oriented to clinical tasks.

Keywords: Ubiquitous Computing; UbiHealth; Pervasive Electronic Health System, Activity Based Computing; End-User Programming; Context-aware Computing.

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 [1][2].

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* - Prontuário Eletrônico do Paciente), 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 [3] 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 [4]. 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 [5], neste trabalho sistemas pervasivos são vistos como sinônimos de sistemas ubíquos.

O trabalho descrito neste artigo está inserido no escopo do projeto ClinicSpace, o qual propõe a prototipação de uma arquitetura de software para um sistema de informação em saúde ubíquo, baseada no ponto de vista do usuário-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 (programado pelo usuário-final).

Atuais EHSs são desenvolvidos a partir da visão (i) da organização hospitalar (EHR - *Electronic Health Record*) ou (ii) do paciente (*Personal Health System*, Sistemas de Saúde Pessoal, tais como o *GoogleHealth*). Observa-se que 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 para reduzir a intrusão desses sistemas no ambiente hospitalar, dando ao médico o controle e programação do sistema, adequando-o ao seu próprio estilo de trabalho. Dessa forma, disponibiliza-se 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. Este trabalho apresenta uma situação-problema para auxiliar a descrição da arquitetura para o auxilio as tarefas clínicas.

1.1 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. 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, por exemplo: (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. Vários recursos computacionais são utilizados pelo médico na execução de suas atividades, tais como computadores pessoais e dispositivos móveis.

As descrições aqui presentes são atividades cotidianas de médicos, porém poucos trabalhos relevantes de pesquisa procuram sistematizar estes desafios, abaixo segue alguns trabalhos relacionados que contemplam computação ubíqua com atividades humanas em diversos domínios.

1.2 Trabalhos Relacionados

O conceito de Computação Baseada em Tarefas foi introduzido pelo Projeto Aura [6]. 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 [7], 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 [6] 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 e gerenciamento dos serviços. Na área de Saúde, o projeto *Activity-Based Computing* [8] apresenta uma proposta para utilização de Computação baseada em Tarefas destinada a Ambientes de Saúde.

Como se observa, 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 (descrito neste artigo), 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

O artigo apresenta os conceitos e elementos do sistema ClinicSpace (seção 2) a partir de uma situação-problema (seção 1.1), na qual se identificam alguns requisitos a serem modelados para as atividades clínicas (seção 2.1), seção 1.2 uma comparação com os poucos trabalhos relacionados. A seção 3 apresenta os testes realizados. As considerações finais, bem como trabalhos futuros são apresentadas na seção 4.

2 Arquitetura ClinicSpace

Para permitir o suporte computacional à situação descrita e aos requisitos identificados (seção 2.1), está-se prototipando uma arquitetura de software, chamada ClinicSpace, integrando tecnologias e conceitos da Computação Móvel, Pervasiva e Ubíqua, Programação Orientada ao Usuário-final *(End-User Programming)* e Computação Sensível ao Contexto (*Contex-Aware Computing).*

A descrição das camadas dessa arquitetura se dá, neste artigo, a partir da visão do usuário-final (médico) para realizar as atividades descritas na situação-problema, até a camada de execução e gerenciamento do ambiente pervasivo pelo middleware EXEHDA [4][9], que integra a arquitetura. Para que o médico possa utilizar a ferramenta no auxílio à realização de suas atividades diárias, o sistema ClinicSpace disponibiliza para ele a *Interface de Edição de Tarefas e Contexto* (IETC)[10], que lhe permite programar suas atividades.

2.1 Requisitos das Atividades Clínicas

A área clínica possui diversas características dinâmicas que tornam a construção de EHS um desafio. Uma dessas características é a mobilidade. Os 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 EHR deve ter uma semântica 'siga-me' [3][9], desloca-se junto com o médico (migração) e adapta-se às situações onde se encontra (contexto), além de ter acesso a uma base de dados pervasiva (*always on*, *anytime*, *anywhere*, *anydevice*, *anynetwork*).

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. Para atender aos requisitos acima, acredita-se que uma nova forma de projetar EHSs deve ser analisada. A arquitetura que se propõe para atingir esse objetivo é descrita a seguir.

2.2 Programação de Tarefas pelo Médico - Interface de Edição de Tarefas e Contexto (Programming Time)

Para permitir a programação pelo usuário-final, o sistema ClinicSpace modela atividades clínicas através do conceito de composição de tarefas e fluxos de execução, com o intuito de disponibilizar uma interface simples e intuitiva para a utilização pelos médicos. A IETC é a ferramenta de interação do médico com o sistema ClinicSpace. Esta deve estar sempre disponível (*always on*), em todo lugar (*anywhere*), a qualquer tempo (*anytime*) e com qualquer dispositivo (*any device*), ou seja, tem uma funcionalidade pervasiva. Quando o médico inicia o uso da ferramenta é montada automaticamente uma interface com base nas suas tarefas já programadas, como ilustra a Figura 1.

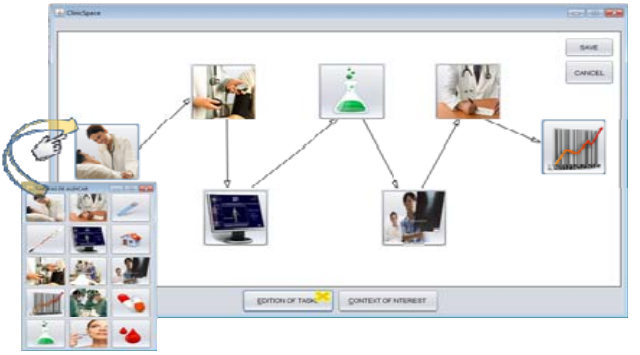

Figure 1: Interface de Edição de Tarefas.

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 a edição de uma tarefa, ele tem à sua disposição, na IETC, um conjunto mínimo, chamado *subtarefas*, as quais podem ser selecionadas e arrastadas para comporem a tarefa, conforme o entendimento do usuário. Por exemplo, para o médico X, a atividade de atendimento ao paciente inicia com o acesso às informações de sinais vitais, enquanto que o médico Y inicia vendo o histórico do paciente. Para programar a funcionalidade desejada, o médico X seleciona o ícone correspondente a *obter informações*, arrasta-o para a área de edição, e associa-lhe as entidades de contexto: *paciente* e *dispositivo* (ver seção 2.3). Para programar os demais procedimentos, a metodologia é a mesma. Esse mecanismo é recursivo, uma tarefa pode ser composta de outras tarefas e não somente de subtarefas. O médico informa também qual é o fluxo de execução, criando associações entre as tarefas, ou seja, ligando uma à outra.

2.3 Execução das Tarefas Programadas (Run Time)

O médico interage com o ClinicSpace sempre através da IETC. Quando uma tarefa é iniciada, por vontade do médico ou por gatilho de um elemento de contexto, são obtidas automaticamente as informações de perfil do médico e sua localização. Na realização da tarefa *Atendimento ao Paciente*, telas específicas vinculadas com cada subtarefa programada (figura 2) são apresentadas de forma pró-ativas para simplificar a execução, exigindo menos interferência e atenção do médico em termos de controle do sistema computacional. Assim, uma definição ontológica da tarefa (ver seção 2.6) é carregada e as subtarefas são executadas.

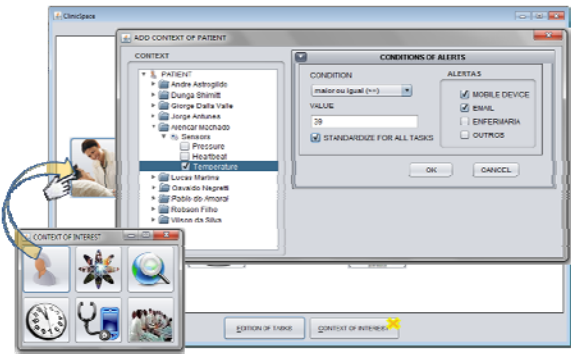

Figure 2: Modelagem do Contexto de Interesse.

A Figura 3 ilustra o diagrama de sequência para a tarefa de *verificação do histórico de saúde do paciente*. Quando esta é iniciada, através da IETC, é chamada a subtarefa de *identificação do paciente (subTarefa1).* Baseando-se na localização do médico, é apresentada uma tela para a escolha de qual paciente o médico irá atender, caso mais de um

de seus pacientes seja detectado no raio de detecção, no momento da realização da tarefa. Conhecendo qual paciente está sendo atendida, a tarefa inicia a subtarefa *busca de dados do paciente (subTarefa2)*. Esta realiza uma chamada a uma aplicação específica do pEHS – *pervasive Electronic Health System* [11], que retorna dados contextualizados do histórico do paciente. Por exemplo, o médico associou a essa subtarefa os elementos de contexto *perfil* e *dispositivo*. Assim, as informações retornadas são somente as de seu interesse e cuja visualização será adaptada à capacidade de visualização do dispositivo, em uso pelo médico, no momento. As demais subtarefas do cenário proposto se relacionam com os mesmos subsistemas e serviços da arquitetura, que, por motivos de espaço, não serão detalhados.

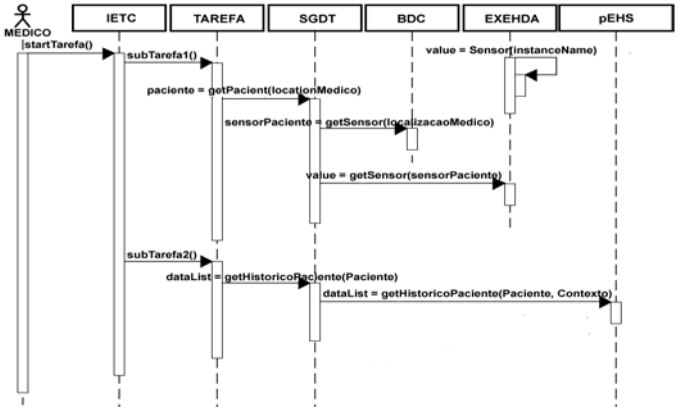

Figure 3: Diagrama de Sequência da execução da tarefa de Atendimento ao Paciente.

2.4 O modelo de Tarefa (Infra-Estrutura Computacional de Suporte)

O termo *tarefa* é sinônimo de *atividade* que se refere 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 [12]. Aplicando ao ambiente clínico o modelo genérico proposto pela Teoria da Atividade, tem-se o médico atuando como o 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, as quais consistem em registros, procedimentos, equipamentos e recursos. Essa mediação segue regras especificadas através de guias clínicos [13].

Ao modelar essas noções no sistema ClinicSpace, a atividade clínica foi decomposta em um conjunto de tarefas – definidas como atividades que tem o auxílio de recursos computacionais. As tarefas podem ser modeladas usando o conceito de composição, tendo como ponto de partida as subtarefas – conjunto básico de aplicações projetadas e disponibilizadas pelo sistema ClinicSpace. 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.

Como visto, o usuário interage com o sistema, personaliza suas tarefas, as quais são armazenadas vinculadas 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 disponibilizadas na IETC [10]. Essas subtarefas foram determinadas a partir do estudo de Laerum and Faxvaag que definiram as tarefas mais comuns no ambiente hospitalar [13]. As tarefas ClinicSpace 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.

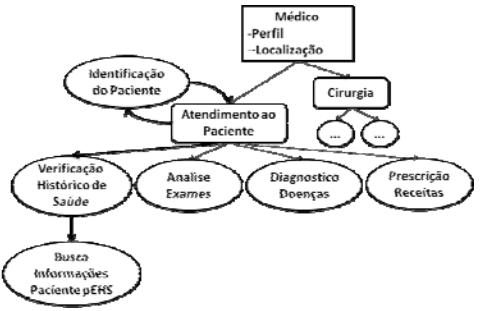

Figure 4: Representação-exemplo do relacionamento entre as tarefas e sub-tarefas.

Quanto mais abrangente for a tarefa, menos recursos físicos ela utiliza, pois somente as subtarefas tem código computacional a ser executado. As subtarefas são os nós finais do grafo. Por exemplo, na figura 4, a tarefa *Atendimento ao Paciente* faz o sincronismo necessário entre as subtarefas *Identificação do Paciente* e *Verificação de Histórico de Saúde*. A tarefa controla as informações necessárias para iniciar suas subtarefas; considerando a figura 4, 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*).

2.5 O modelo de Contexto

A modelo de contexto proposto neste trabalho foi baseada nos estudos de Henricksen, Indulska, e Rakotonirainy [14][15], 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 5, onde o diagrama não expressa o tipo de informações trocadas entre os elementos, devido a este não

serem contemplados pela UML.

Na figura 5, 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 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 [16].

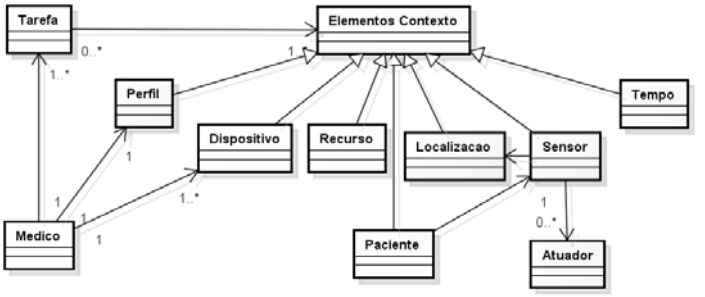

Figure 5: Modelagem do contexto clínico

A modelagem do contexto onde a aplicação pervasiva (tarefa) 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) alavanca a integração de elementos de contexto em uma aplicação pervasiva, tal contribuição 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 está 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 monitorando uma atividade em um ambiente específico.

2.6 Descrição Ontológica da tarefa e contexto

Como visto, no sistema ClinicSpace, as atividades são decompostas em tarefas e subtarefas, programada pelo usuário de acordo com 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 na arquitetura de software que suporta o sistema ClinicSpace. Na ontologia que descreve cada tarefa é especificado o seu contexto, o qual é responsável por manter os elementos de contexto associados às tarefas e o vínculo com seu dono (médico). Esta ontologia de tarefas e contexto foi modelada usando a ferramenta Protégé (http://protege.stanford.edu) que gera um arquivo OWL-DL, com a estrutura semântica original dos conceitos.

A figura 6 apresenta a taxonomia ontológica que representa os conceitos Médico, Tarefa e Contexto. Este gráfico foi gerado através do *plugin OntoGraf*. Na figura, 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*; criando, assim, o conceito que uma tarefa em determinado momento pode ser subtarefa de outra tarefa. 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.

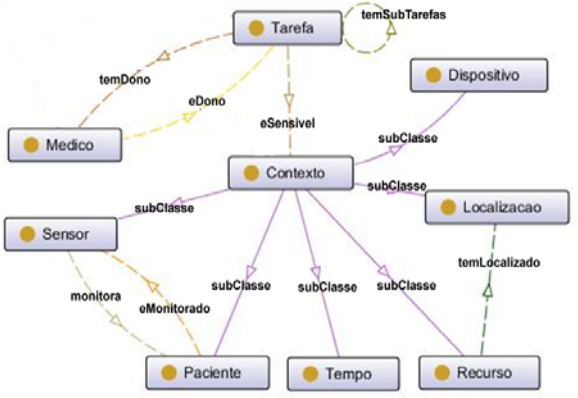

Figure 6: Representação gráfica do modelo ontológico de tarefa e contexto

Para o médico programar o contexto de uma tarefa, 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, sendo o gerenciamento desse contexto realizado de forma pró-ativa pelo ClinicSpace.

O ClinicSpace, quando acionado de forma explícita (usando a IETC) ou implícita (alteração detectada de elementos de contexto associado à tarefas com disparo), carrega as instâncias das tarefas contendo as relações existentes entre seus elementos. Desta forma, quando é adicionado mais de uma subtarefa ou elemento de contexto na tarefa, estas são

instâncias desses elementos que, após a modelagem ser salva pelo médico, são armazenadas no arquivo OWL de tarefas personalizadas pelo médico.

A figura 7 ilustra, de forma gráfica, uma instância de indivíduos da ontologia correspondentes à situação-problema (seção 1.1), onde o indivíduo *Medico_Alencar* é dono de um tarefa que contém cinco subtarefas, as quais, respectivamente, contém um contexto vinculado.

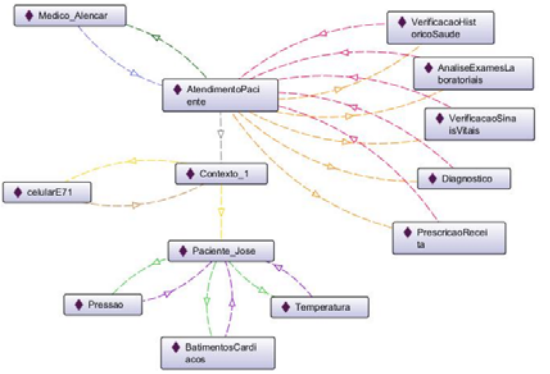

Figure 7: Representação gráfica das instâncias e relacionamentos contidos em uma Tarefa.

2.7 Componentes da Arquitetura

A arquitetura de software ClinicSpace (Figura 8), é composta por (i) Interface de Edição de Tarefas e Contexto (IETC), (ii) Sistema Gerenciador Distribuído de Tarefas (SGDT)[14], (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, (v) *Middleware* EXEHDA[9] de gerenciamento do ambiente pervasivo.

Conforme ilustra a figura 8, a arquitetura ClinicSpace provê ferramentas para que o médico possa, através de uma interface simples, programar os suportes computacionais às suas atividades e seu fluxo de trabalho. Para tal, o Sistema Gerenciador Distribuído de Tarefas (SGDT)[19] faz a mediação entre o *Middleware* de controle de ambientes pervasivos (EXEHDA), os Sistemas Eletrônicos de Saúde Pervasivos (pEHS), as bases de dados e a Interface de Edição de Tarefas e Contexto, provendo o acesso às informações de forma transparente, de acordo com as tarefas programadas pelo usuário. O SGDT é implementado seguindo a linha de modularização existente no EXEHDA; assim, implementa módulos que disponibilizam serviços, que são gerenciados pelo Serviço de Execução Distribuição do EXEHDA [18].

O SGDT [19] é constituído pelos serviços: (i) Serviço de Acesso as Tarefas (SAT) - carrega as tarefas já modeladas pelo médico (ontologia); (ii) Serviço de Contexto para Tarefas (SCT) - acessa o banco de dados de contexto e constrói um link para busca de informações, em tempo de execução, providas pelos sensores administrados pelo EXEHDA [6], bem como as chamadas de funções relativas ao pEHS [8]; (iii) Serviço de Acesso a Tarefas Ativas (SATA) - provê a possibilidade de interrupção e retomada de execução de tarefas pelo usuário, bem como migração de tarefas entre os dispositivos utilizados pelo usuário; (iv) Serviço de Gerenciamento de Tarefas (SGT) - atua como uma camada mediadora entre a aplicação e os demais serviços, sendo construído com base no *design patterns Facede*[20], onde realiza a abstração de todos os serviços para a interface gráfica.

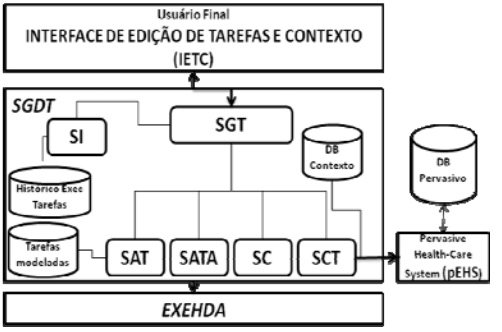

Figure 8: Arquitetura ClinicSpace

2.8 Middleware de Gerenciamento do Ambiente Pervasivo – EXEHDA

O *Middleware* EXEHDA [9] 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 conectividade no tratamento de conexões e desconexões realizadas por dispositivos de acesso sem fio. EXEHDA disponibiliza serviços para o gerenciamento pervasivo de ambientes, destacando-se: (i) Acesso Pervasivo - gerencia a *base de dados da aplicação (BDA)*, para disponibilização de dados e código de cunho mais abrangente, não pertencente a usuários específicos (ex: banco de dados do contexto e 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; (ii) Execução Distribuída - provê a possibilidade de execução distribuída dos serviços e aplicações; (iii) Comunicação - gerencia 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 infraestrutura de hardware sem fio; (iv) Serviço de Contexto e Adaptação - provê a infraestrutura para o gerenciamento dos elementos do contexto,

disponibilizando através do processo de subscrição (inscrição para recebimento de informação desejada) os dados coletados pelos sensores.

3 Testes e relato do trabalho

Os testes realizados constituíram-se na geração de um ambiente sintético e controlado para busca de resultados de escalabilidade. O Ambiente de testes foi configurado gerando um nodo base do EXEHDA, responsável pelo gerenciamento da célula (Processador: Intel Core 2 Duo; memória: 2 GB; Java versão 1.6.0_18) onde são mantidas as descrições ontológicas das tarefas (acessadas pelo SAT) e Banco de Contexto. O nodo cliente (Processador: AMD 2.20 GHz; memória: 2 GB; Java versão 1.6.0_18) é usado pelo usuário para executar suas tarefas. Para simular o funcionamento dos serviços foram implementadas as tarefas mencionadas na situação-problema. 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).

Na figura 9(a), o arquivo contendo uma instância da ontologia, que corresponde a um médico (1M – 13KB), expressa num arquivo OWL de 13 KB, demorou para ser carregada em 2654 ms. Nota-se que o tempo inicial para o carregamento da ontologia pelo SAT é em torno de 2,5 segundos - 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.

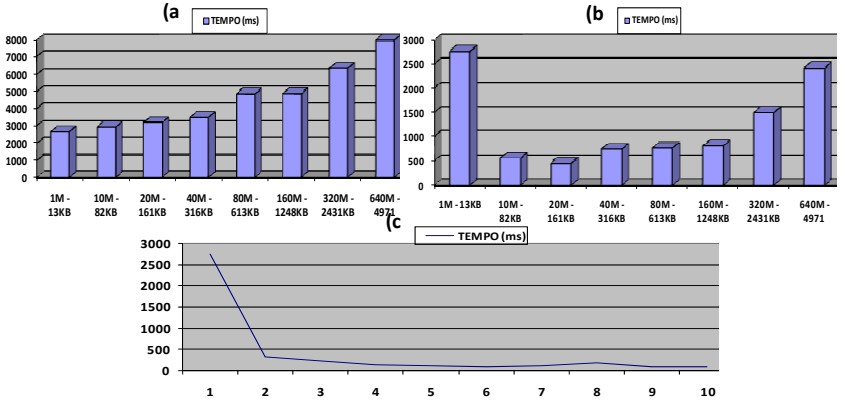

Figura 9: Gráfico do tempo médio para inicialização do SAT.

Na figura 9(b) são apresentados 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 maiores 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 – essa situação é natural para um middleware de gerenciamento de ambiente pervasivo.

Procurando melhorar o tempo de carga 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. Está 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.

Na figura 9(c) são apresentados 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 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 [21], 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, porém a solução aqui proposta buscando utilizar a ontologia juntamente com objetos Java se mostrou bastante satisfatória, como demonstra os gráficos.

4 Conclusão

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, gerenciamento do 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 prototipação de uma arquitetura para utilização em um hospital do futuro, adicionam-se os conceitos e tecnologias da Computação Ubíqua de forma a reduzir a interferência direta da computação nas atividades cotidianas dos médicos. A modelagem de Tarefas Clínicas e Associação de Elementos de Contexto à Interface de Edição de Tarefas 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.

Os trabalhos futuros propostos no projeto ClinicSpace abrangem: (i) testes de usabilidade em um ambiente real, (ii) 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, (iii) pesquisa de mais entidades/elementos de contexto para que o poder de sensibilidade da tarefa ao contexto aumente gradativamente com cada elemento adicionado.

Referências

[1] 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

[2] Miller, R.; Sim, I. Physicians Use Of Electronic Medical Records: Barriers And Solutions. In: *Health Affairs*, 2004, 23, no. 2. p 116-126.

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

[4] 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.

[5] Varshney, U., "Pervasive Healthcare" In: *IEEE Computer*, vol. 36, p. 138-140, 2003.

[6] D. Garlan, P. Steenkiste, and B. Schmerl, Project Aura: Toward Distraction-free Pervasive Computing. In *IEEE Pervasive Computing*. New York, NY, 2002. pp. 22-31.

[7] M. Roman, C. Hess, R. Cerqueira, R.H. Campbell, and K. Nahrstedt, Gaia: a Middleware Infrastructure to Enable Active Spaces. *In IEEE Pervasive Computing*. New York. 2002. pp. 74-83.

[8] J.E. Bardram and Christensen, Pervasive Computing Support for Hospitals: An Activity-Based Computing Project. In *IEEE Pervasive Computing*, v. 6, issue 1, 2007.

[9] Augustin, I., Yamin, A., *and* Geyer, EXEHDA: Adaptive middleware for building a pervasive grid environment. In *Frontiers in Artificial Intelligence and Applications: Self-Organization and Autonomic Informatics* (I), vol. 135, pp. 203-219. IOS Press, 2005.

[10] Lorenzi, F., Ferreira, G., Rizzeti, T., Librelotto, G., Augustin, I., "Ferramenta para a Programação pelo Usuário-Final de Tarefas Clínicas em um Ambiente de Saúde Ubíquo" *Conferencia Latino Americana de Informática*, 2009.

[11] Caroline, V., Machado, A., Augustin, I. PEHS Arquitetura de um Sistema de Informação Pervasivo para Auxílio às Atividades Clínicas. In *Revista Brasileira de Computação Aplicada*. ISSN 2176-6649. Volume 2, Número 2. 2010

[12] Ranganathan, A., Campbell, R.: Supporting Tasks in a Programmable Smart Home. *3rd International Conference on Smart Homes and Health*, Nothern Irland, 2005.

[13] 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.

[14] Henricksen, K., Indulska, J.: Developing context-aware pervasive computing applications: Models and approach. *Journal of Pervasive Mobile Computing*, 2006

[15] 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.

[16] Alencar, M; Augustin, I. Ferramenta para Definição de Contexto pelo Usuário-Final na Programação de Tarefas Clínicas em um Sistema de Saúde Pervasivo. *In: SBCUP 2010*

[17] G. Ferreira, Augustin, I; Librelotto, G; Lorenzi, F; Alencar, M ; Extending a Middleware for Pervasive Computing to Programmable Task Management in an Environment of Personalized Clinical; In: *UBICOMM*, 2010.

[18] Alencar, M ; Augustin, I. ; Librelotto, G . Ciência do Contexto para Tarefas Clínicas em um Sistema de Saúde Pervasivo. In: *CLEI*, 2010.

[19] G. Ferreira, I. Augustin, G.R. Librelotto, F.L. Silva, and A.C. Yamin, Middleware for management of end-user programming of clinical activities in a pervasive environment. In *Proceedings of the 2009 Workshop on Middleware for Ubiquitous and Pervasive Systems*. ACM New York, USA. 2009. pp 7-12.

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

[21] Bettinia, C., Henricksen, K., Indulska, J., A survey of Context Modelling and Reasoning Techniques. *Pervasive and Mobile Computing*, 6(2), 161-180 2010.