Um Mecanismo Baseado em Ontologias para Processar Informações de Contexto na Computação Pervasiva

João Lopes, Fernando Afonso, Luiz Palazzo, Adenauer Yamin
Universidade Católica de Pelotas, Programa de Pós-Graduação em Informática,
Pelotas, Brasil
{joaolopes, afonso, lpalazzo, adenauer}@ucpel.tche.br

Iara Augustin
Universidade Federal de Santa Maria, Programa de Pós-Graduação em Informática,
Santa Maria, Brasil
august@inf.ufsm.br

Claudio Geyer
Universidade Federal do Rio Grande do Sul, Programa de Pós-Graduação em Computação
Porto Alegre, Brasil
geyer@inf.ufrgs.br

Resumo

Em um ambiente de Computação Pervasiva, a consciência do contexto se mostra um aspecto central, os diferentes componentes de software envolvidos, do ambiente de execução e das aplicações, devem adaptar-se de forma colaborativa às mudanças no contexto. Assim, o objetivo central deste trabalho é a qualificação dos mecanismos para processar informações de contexto propondo, para isso, o uso de ontologias. O EXEHDA-ON tem como premissa de pesquisa que a possibilidade de empregar uma semântica de maior expressividade que a usualmente praticada no processamento dos dados sensorados permite atingir melhores níveis de descrição nas informações que caracterizam o contexto do ambiente computacional. Entende-se como principais contribuições deste trabalho a definição de um modelo ontológico que caracteriza um ambiente pervasivo de computação e a integração deste modelo, através de um mecanismo de sensibilidade ao contexto, à arquitetura de software do middleware EXEHDA.

1 Introdução

Mark Weiser idealizou ambientes físicos com dispositivos computacionais integrados que auxiliariam indivíduos na realização de suas tarefas cotidianas ao fornecer-lhes informações e serviços de forma contínua e transparente [18]. Essa visão resume o que se espera da Computação Pervasiva: acesso do usuário ao seu ambiente computacional independente de localização, tempo e equipamento [5].

Em um ambiente de Computação Pervasiva, os dispositivos, serviços e componentes de software devem ser conscientes de seus contextos e colaborativamente adaptar-se às suas mudanças, caracterizando assim sensibilidade ao contexto [2] [20]. Uma questão relevante na sensibilidade ao contexto é o grau de expressividade que se pode obter na descrição dos possíveis estados do mesmo. Neste sentido, considera-se que o uso de ontologias contribui para qualificar os mecanismos de sensibilidade ao contexto, em função da elevada expressividade que o uso destas pode propiciar. Uma ontologia corresponde a uma especificação formal e explícita de uma conceituação compartilhada [8].

Considerando este cenário, este trabalho apresenta um mecanismo de sensibilidade ao contexto para coletar, processar e disseminar informações de contexto na perspectiva da Computação Pervasiva, considerando o ambiente pervasivo definido no projeto ISAM (Infra-estrutura de Suporte às Aplicações Móveis Distribuídas) e provido pelo *middleware* EXEHDA (*Execution Environment for Highly Distributed Applications*) [1] [19].

O EXEHDA é um *middleware* adaptativo ao contexto e baseado em serviços que visa criar e gerenciar um ambiente pervasivo, bem como promover a execução das aplicações direcionadas à Computação Pervasiva. Estas aplicações são

distribuídas, móveis e adaptativas ao contexto em que seu processamento ocorre. O suporte à adaptação no EXEHDA está associado à operação do subsistema de reconhecimento de contexto e adaptação. Este subsistema inclui serviços que tratam desde a extração das "informações brutas" sobre as características dinâmicas e estáticas dos recursos que compõem o ambiente pervasivo, passando pela identificação em alto nível dos elementos de contexto, até o disparo das ações de adaptação em reação a modificações no estado de tais elementos de contexto.

O EXEHDA-ON utiliza uma abordagem baseada em ontologias para a modelagem do contexto do ambiente pervasivo, bem como para realizar pesquisa e inferência no correspondente modelo ontológico. Este modelo ontológico tem abrangência celular, sendo alimentado por um serviço de monitoramento. Deste modo, o modelo descreve semanticamente o estado atual do ambiente, utilizando um vocabulário comum e interpretável pelos servidores de contexto existentes nas células de execução do EXEHDA.

O artigo está organizado nas seguintes seções: a seção 2 descreve a concepção da modelagem do EXEHDA-ON; a seção 3 descreve a implementação do servidor de contexto do EXEHDA-ON; a seção 4 apresenta um estudo de caso; a seção 5 apresenta os trabalhos relacionados e na seção 6 são apresentadas as considerações finais.

2 Aspectos de Modelagem do EXEHDA-ON

A modelagem do EXEHDA-ON contemplou dois principais esforços de concepção: (*i*) modelagem ontológica do ambiente pervasivo, a qual prevê o uso de ontologias implementadas em OWL, sobre as quais o EXEHDA-ON realiza a representação e o processamento das informações de contexto; (*ii*) modelagem da arquitetura de software com a especificação dos diferentes serviços que a integram.

2.1 Modelagem Ontológica

A construção e o processamento do modelo ontológico do EXEHDA-ON são baseados em tecnologias da Web Semântica. A linguagem escolhida para construção do modelo ontológico do EXEHDA-ON foi a OWL (*Web Ontology Language*) [3], padrão para a Web Semântica. Como sub-linguagem foi adotada a OWL-DL, esta sub-linguagem corresponde à lógica de descrição e provê um maior grau de expressividade onde todas as conclusões são computáveis e todas as computações terminam em tempo finito.

A linguagem SPARQL (*SPARQL Protocol And RDF Query Language*) [16], recomendada pelo W3C (*World Wide Web Consortium*), foi escolhida para realização de consultas nas ontologias. Também, foi adotada a API Java do *toolkit* Jena [12], por oferecer: (*i*) mecanismos para manipulação de modelos RDF em memória, bases de dados relacionais e arquivos; (*ii*) suporte à linguagem de consulta de dados RDF SPARQL; (*iii*) um conjunto de APIs para manipulação de ontologias codificadas em OWL; (*iv*) máquinas de inferência baseadas em ontologias e regras.

O modelo ontológico para uso no EXEHDA-ON foi definido considerando aspectos que modelassem o domínio do ambiente pervasivo provido pelo EXEHDA (vide Figura 1). A perspectiva é que este modelo represente o estado atual do ambiente de execução pervasivo provido pelo EXEHDA, gerando deste modo um conhecimento sobre o mesmo, possibilitando assim sua manipulação pelo servidor de contexto, o qual responde às demandas introduzidas pelas aplicações dos usuários.

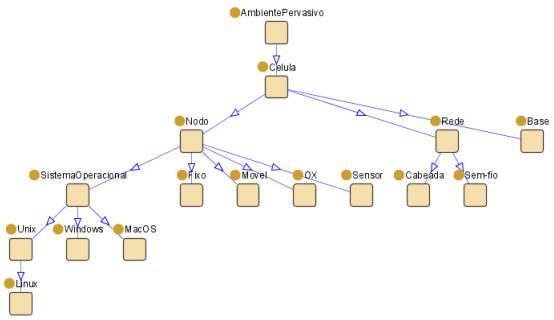

Figura 1. Árvore de conceitos da ontologia

2.2 Modelagem da Arquitetura de Software

Nesta seção é descrita a arquitetura de software do EXEHDA-ON e a forma de adequação da mesma ao Subsistema de Reconhecimento de Contexto e Adaptação do EXEHDA, identificando seus pontos de integração e seus serviços.

De modo geral, uma arquitetura para sistemas sensíveis ao contexto envolve uma série de sensores, de software e/ou de hardware, que monitoram os aspectos de interesse do ambiente computacional (*middleware* e aplicações). As informações colhidas por esses sensores são passadas a um conjunto de serviços de contexto, onde são processadas e/ou modificadas para que possam ser entregues aos consumidores das informações contextualizadas [10].

A arquitetura do EXEHDA-ON prevê a inclusão de serviços e componentes nos "EXEHDANodos" e no "EXEHDABase", os quais são abstrações do ambiente pervasivo provido pelo EXEHDA. A Figura 2 mostra as funcionalidades do EXEHDA-ON integradas ao Subsistema de Reconhecimento de Contexto e Adaptação do EXEHDA.

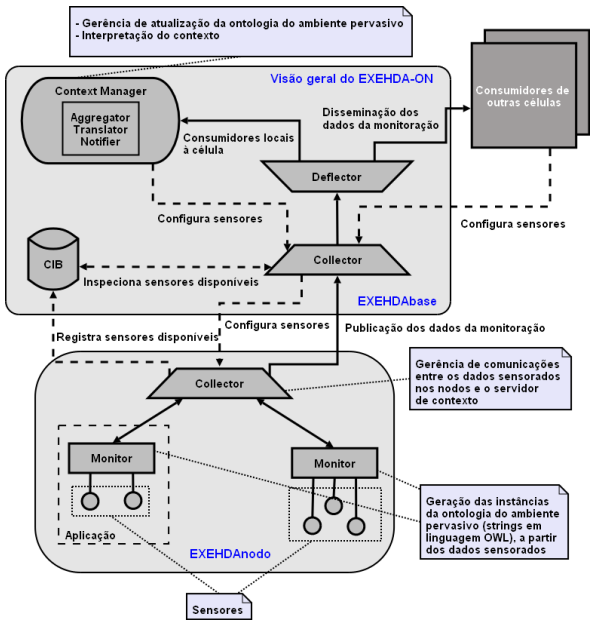

Figura 2. Integração das funcionalidades

Em cada EXEHDANodo, junto ao componente "Monitor", o serviço do EXEHDA-ON "Gerenciador de monitoramento do contexto" fica com a responsabilidade de receber os dados sensorados e gerar as instâncias da ontologia do ambiente pervasivo, representadas por *strings* em linguagem OWL. Por sua vez, o serviço "Collector" do EXEHDA é responsável pela gerência das comunicações entre os dados sensorados nos nodos e o servidor de contexto que aglutina os dados da célula como um todo.

A localização dos serviços do EXEHDA-ON dentro do EXEHDAbase é junto ao "ContextManager", que é serviço chave na construção de informações globais de contexto, mais especificamente dentro das estruturas de cadeias de detecção de contexto do "ContextManager" (*aggregator* e *translator*), que são empregadas, respectivamente, para composição dos dados de um ou mais sensores e para abstração da informação de contexto.

Assim, no EXEHDAbase atua o serviço responsável pela aglutinação das instâncias na ontologia do ambiente pervasivo, denominado "Gerenciador de atualização da base ontológica", e o serviço responsável pelas consultas e inferências sobre a base ontológica do EXEHDA-ON, denominado "Interpretador de contexto".

O serviço "Interpretador de contexto"(vide Figura 3) é central para consecução dos objetivos propostos para o EXEHDA-ON, realizando o processamento das informações de contexto com o intuito de identificar na base ontológica a existência das condições de contexto de interesse dos consumidores registrados. Este serviço recebe as solicitações dos consumidores a partir do "Gerenciador de subscrições" e realiza consultas e inferências sobre a base ontológica, utilizando o componente "Gerenciador de consultas e inferências".

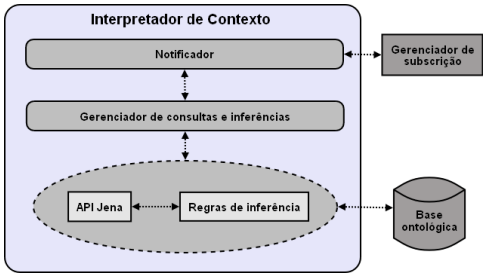

Figura 3. Interpretador de contexto

3 Implementação do Servidor de Contexto

O servidor de contexto para o ambiente pervasivo foi modelado e desenvolvido com o uso da API Jena e a linguagem de programação Java. Os dados sensorados são compostos em instâncias da ontologia do ambiente pervasivo no nodo (processamento de *strings*), as quais são enviadas ao servidor de contexto, localizado no EXEHDABase. Estas instâncias contêm as informações de identificação do nodo e os dados sensorados. No servidor de contexto é feita a aglutinação destas instâncias na ontologia do ambiente pervasivo na célula com a utilização da API Jena.

Com o intuito de abstrair aspectos de baixo nível relativos ao tratamento das comunicações em rede, na implementação do servidor de contexto foram utilizados os serviços de comunicação do EXEHDA. A classe principal do servidor de contexto executa uma *thread* responsável por utilizar esses serviços para aguardar as atualizações de contexto dos nodos.

O servidor de contexto mantém uma base de dados ontológica com informações sobre o estado do ambiente pervasivo. Todos os procedimentos relativos à manipulação da ontologia são realizados com o uso da API Jena. Quando a *thread* responsável pela comunicação recebe a ontologia do ambiente pervasivo atualizada de algum nodo ela a repassa para a classe principal "*ContextServer*", a qual possui o método "*mergeOntologies*" responsável por verificar a ontologia recebida e realizar as atualizações necessárias na base de dados ontológica. O servidor é executado na base da célula, gravando as alterações ocorridas na ontologia dentro do arquivo *log* do EXEHDA.

Também, um serviço de consultas foi desenvolvido junto ao servidor de contexto. Este serviço possibilita a realização de pesquisas na ontologia do ambiente pervasivo utilizando a linguagem SPARQL. A linguagem de consulta SPARQL é utilizada através da biblioteca ARQ da API Jena. Os

métodos da classe *QueryConnection* implementam as consultas em SPARQL que pesquisam propriedades dos nodos, tais como: temperatura do processador, memória total, tamanho do disco e número de processadores.

4 Estudo de Caso: Suporte à Decisão para o Escalonamento de Recursos

Na perspectiva deste estudo de caso, os nodos existentes em uma célula, em função do seu estado, são selecionados para computações paralelas numericamente intensivas. Nesta seção é caracterizado o uso do EXEHDA-ON enquanto provedor de informações para a tomada de decisões por parte de um escalonador de recursos quando da gerência de uma execução paralela.

No EXEHDA, a escolha do nodo onde é instalado um componente de software denomina-se adaptação não-funcional. Neste tipo de adaptação a natureza da computação a ser realizada é mantida, independentemente do equipamento aonde o processamento será disparado [19].

Para as avaliações neste estudo de caso, foi utilizado o "CpuSteal" [14], um gerador de cargas baseado em uma distribuição de probabilidades exponencial. O "CpuSteal" opera em ciclos de ativação e desativação. Quando ativo, ele gera operações de ponto flutuante ocupando o processador, quando inativo ele entra em repouso. O controle do nível de ocupação média do processador é feito através da determinação de quanto tempo ele permanece ativo e quanto tempo ele fica em repouso em cada um dos ciclos.

Na avaliação do suporte do EXEHDA-ON à tomada de decisão para o escalonamento de recursos foi utilizado o *cluster* H3P do Centro Politécnico da UCPEL (http://h3p.g3pd.ucpel.tche.br), composto de 10 nodos com sistema operacional Linux, freqüência de 2020MHz e memória de 185MB. Estes computadores são interligados por uma rede *FastEthernet*. As cargas computacionais ficaram distribuídas conforme Tabela 1 e foram produzidas pelo gerador de cargas "CPUSteal". O nodo "H3P" é o gerenciador do sistema, assim não foi utilizado para o processamento. Nos nodos "H1", "H2" e "H3" não foi imposta carga gerada pelo "CPUSteal".

4.1 Aplicação de Teste Desenvolvida

A aplicação prevista instala computações remotas em função da disponibilidade de recursos computacionais na célula de execução. A medida que surjam recursos que atendam os requisitos mínimos estabelecidos pelo desenvolvedor no âmbito do contexto celular o componente da aplicação responsável pela gerência da execução é notificado para que execute os procedimentos pertinentes.

Tabela 1. Carga média dos nodos

Nodo	Carga (%)
H4	18.48
H5	18.82
H6	39.30
H7	40.26
H8	59.95
H9	61.45

A aplicação desenvolvida é do tipo sintético, e tem por objetivo central permitir a exploração das funcionalidades do EXEHDA-ON. Sua finalidade matemática é calcular o número π utilizando o método de Monte Carlo [13], do ponto de vista computacional se trata de uma computação paralela do tipo *bag-of-tasks* cujo número de computações paralelas é determinado pelo programador. Considerando que o processamento de cada uma das iterações pode ocorrer de forma independente, uma das maneiras de particionar o problema é dividir o número total de iterações desejadas entre vários nodos.

4.2 Resultados Obtidos

O suporte à decisão para o escalonamento de recursos com o EXEHDA-ON foi avaliado através de quatro execuções da aplicação nos equipamentos do *cluster* H3P. Parte dos nodos foi submetido a diferentes cargas sintéticas, através de um gerador programado de contexto de carga de processador, conforme mostra a Tabela 1.

Em cada uma das quatro execuções foram selecionados cinco nodos, sendo que em três destas execuções os nodos foram escolhidos de forma aleatória. Em uma das execuções os nodos foram selecionados a partir de consulta SPARQL construída pelo EXEHDA-ON, considerando um critério de carga da CPU inferior a 70%, bem como uma ordem dos nodos crescente de carga da CPU e decrescente de poder computacional (vide Listagem 1).

Observa-se que em função do resultado ordenado da consulta torna-se possível a seleção por parte do módulo Escalonador do número necessário para o processamento do cálculo, considerando as tarefas que faltam.

Cabe ressaltar que caso o número de nodos atendendo ao critério fosse inferior a cinco, estes seriam selecionados e, o escalonador aguardaria novos nodos para dar continuidade ao restante do processamento. Quando o número de nodos necessário para o processamento é atingido o nodo mestre cancela a subscrição do contexto de interesse.

O cálculo do π pelo método de Monte Carlo foi realizado com o lançamento de 10 bilhões de pontos. Este número total de iterações foi dividido entre os cinco nodos selecionados em cada uma das execuções. Na Listagem 2 são

mostrados os resultados obtidos. A forma de apresentação é a mesma registrada no *log* do EXEHDA.

Listagem 1. Nodos selecionados

```
Consulta SPARQL executada:

PREFIX exehda-on: <http://exehda.ucpel.tche.br/exehda-on#>
SELECT ?NODO ?CARGACPU ?PODERCOMP
WHERE { ?NODO exehda-on:CargaCPU ?CARGACPU;
        exehda-on:PoderComp ?PODERCOMP
FILTER (?CARGACPU <= 70.00)}
ORDER BY ?CARGACPU DESC(?PODERCOMP)

Resultado da consulta:

          Nodo   | CargaCPU   | PoderComp
          ==============================
          H1     |   0.00     | 3997.69
          H3     |   0.00     | 3997.69
          H2     |   0.00     | 3325.95
          H4     |  18.48     | 3997.69
          H5     |  18.82     | 3997.69
          H6     |  39.30     | 3997.69
          H7     |  40.26     | 3997.69
          H8     |  59.95     | 3325.95
          H9     |  61.45     | 3325.95
          ==============================

Nodos selecionados: h1, h2, h3, h4, h5
```

Observando os resultados na Listagem 2 pode-se verificar a precisão do impacto do gerador de carga no tempo de execução do módulo de cálculo. Por exemplo, os nodos h1 e h5 possuem o mesmo poder computacional, tendo sido imposto ao nodo h5 uma carga média de 18.82%. Analisando os tempos destes nodos, obtidos nas diferentes execuções apresentadas na Listagem 2, pode-se observar diferenças em torno de 20%.

Listagem 2. Resultados obtidos

```
Execução com nodos selecionados pelo EXEHDA-ON
...
O resultado de hostid:3.exehda-h3p foi 1570799271 em 812.632 segundos
O resultado de hostid:1.exehda-h3p foi 1570789020 em 813.025 segundos
O resultado de hostid:2.exehda-h3p foi 1570817787 em 983.396 segundos
O resultado de hostid:4.exehda-h3p foi 1570801460 em 991.103 segundos
O resultado de hostid:5.exehda-h3p foi 1570780264 em 995.219 segundos
Resultado do calculo do PI: 3.1415951208
...

Execução 1: nodos aleatórios
...
O resultado de hostid:1.exehda-h3p foi 1570801127 em 811.151 segundos
O resultado de hostid:4.exehda-h3p foi 1570781893 em 935.631 segundos
O resultado de hostid:2.exehda-h3p foi 1570806011 em 976.104 segundos
O resultado de hostid:5.exehda-h3p foi 1570804099 em 997.109 segundos
O resultado de hostid:7.exehda-h3p foi 1570797361 em 1221.303 segundos
Resultado do calculo do PI: 3.1415961964
...

Execução 2: nodos aleatórios
...
O resultado de hostid:3.exehda-h3p foi 1570827840 em 813.745 segundos
O resultado de hostid:5.exehda-h3p foi 1570820120 em 957.303 segundos
O resultado de hostid:7.exehda-h3p foi 1570771285 em 1192.303 segundos
O resultado de hostid:9.exehda-h3p foi 1570764426 em 1818.579 segundos
O resultado de hostid:8.exehda-h3p foi 1570813229 em 1850.741 segundos
Resultado do calculo do PI: 3.14159876
...

Execução 3: nodos aleatórios
...
O resultado de hostid:1.exehda-h3p foi 1570800575 em 812.077 segundos
O resultado de hostid:3.exehda-h3p foi 1570802723 em 814.551 segundos
O resultado de hostid:2.exehda-h3p foi 1570791423 em 971.851 segundos
O resultado de hostid:6.exehda-h3p foi 1570793860 em 1084.87 segundos
O resultado de hostid:8.exehda-h3p foi 1570801037 em 1830.695 segundos
Resultado do calculo do PI: 3.1415958472
...
```

Os valores registrados no *log* tem a precedência definida pela ordem que ocorrem, o último a concluir a computação determina o tempo total de execução.

5 Trabalhos Relacionados

Nesta seção é feita uma comparação entre os principais projetos em Computação Sensível ao Contexto utilizados como referência para a definição deste trabalho, e a solução adotada no EXEHDA-ON. Esta comparação está organizada em três grandes categorias: arquitetura, modelagem de contexto e processamento de contexto.

Arquitetura: *CASS* [7] tem uma arquitetura baseada em *middleware* centralizado. *CoBra* [4] possui arquitetura baseada em agentes. Considerando que o EXEHDA-ON foi concebido como extensão do *middleware* EXEHDA, ele mantém os aspectos arquiteturais do mesmo. Assim, sua arquitetura é baseada em serviços, cuja integração visa fornecer a infra-estrutura necessária para suporte à sensibilidade ao contexto em um ambiente pervasivo.

Modelagem de contexto: *Context Toolkit* [6] manipula o contexto através de tuplas com atributos e valores que são codificados usando XML. *Hydrogen* [11] usa uma abordagem orientada a objetos para modelagem do contexto. A estrutura e o vocabulário da ontologia aplicada no *Context Managing Toolkit* são descritos em RDF. No Gaia [15] o contexto é representado através de predicados escritos em DAML+OIL. Abordagens baseadas em ontologias escritas em OWL para modelagem de contexto são encontradas nos projetos *SOCAM* [9] e *CoBrA*. A solução adotada no EXEHDA-ON para modelagem do contexto consiste no uso de ontologias próprias desenvolvidas em OWL, linguagem recomendada pelo W3C, visto que, revisa e incorpora melhoramentos às demais linguagens, tais como: RDF e RDF *Schema* e DAML-OIL. O modelo baseado em ontologias é o mais promissor para a modelagem de contexto em ambientes pervasivos [17].

Processamento de contexto: o processamento do contexto nos projetos *CASS*, *CoBrA*, *Context Toolkit* e *SOCAM* é baseado em motores de inferência que realizam a interpretação do contexto. Modelos de contexto não baseados em ontologias apresentam menor expressividade o que reduz a possibilidade de realização de inferências, sendo esta uma das limitações dos projetos CASS e *Context Toolkit*. No EXEHDA-ON o processamento do contexto é realizado através de serviços e componentes de software que manipulam a base ontológica, utilizando a linguagem de consulta SPARQL e a API Jena.

6 Considerações Finais

Uma questão relevante na sensibilidade ao contexto é o grau de expressividade que se pode obter na descrição dos

possíveis estados do mesmo. Quanto maior a expressividade do modelo de informação do contexto, maior é a capacidade de representar a estrutura e a semântica dos conceitos. Neste sentido, o uso de ontologias contribui para qualificar os mecanismos de sensibilidade ao contexto, em função da elevada expressividade que o uso destas pode propiciar.

O processo de tradução dos dados sensorados para contextualizados no EXEHDA vinha sendo feito por algoritmos e estruturas de dados particulares para cada tipo de aplicação. Assim, a contribuição central da pesquisa EXEHDA-ON é minimizar a gerência desta personalização por parte do programador, tendo proposto a construção de um modelo ontológico que descreva semanticamente o estado atual do ambiente, utilizando um vocabulário comum e interpretável pelo servidor de contexto. Além disso, o fato de ser utilizado um modelo ontológico também torna possível a realização de pesquisas e inferências sobre o estado do ambiente pervasivo, com a utilização de uma linguagem de alto nível.

Na perspectiva de continuidade da pesquisa desenvolvida durante a concepção do EXEHDA-ON os seguintes aspectos poderão ser explorados em trabalhos futuros: (i) analisar o desempenho do EXEHDA-ON considerando as diferentes possibilidades de modelagem ontológica do contexto; (ii) construir diferentes modelos ontológicos em função do domínio a ser tratado; (iii) expandir o mecanismo para construção de contextos que englobem várias células.

Referências

[1] I. Augustin. *Abstrações para uma Linguagem de Programação Visando Aplicações Móveis Conscientes do Contexto em um Ambiente de Pervasive Computing*. Tese (doutorado em ciência da computação), Instituto de Informática, UFRGS, Porto Alegre, RS, 2004.

[2] I. Augustin, A. C. Yamin, L. C. da Silva, R. A. Real, G. Frainer, and C. F. R. Geyer. Isamadapt: abstractions and tools for designing general-purpose pervasive applications: Experiences with auto-adaptive and reconfigurable systems. *Softw. Pract. Exper.*, 36(11-12):1231–1256, 2006.

[3] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. Owl web ontology language reference, 2004. Disponível em: <http://www.w3.org/TR/owl-ref/>. Acesso em maio de 2008.

[4] H. Chen. *An Intelligent Broker Architecture for Pervasive Context-Aware Systems*. Dissertation (doctor of philosophy), University of Maryland, Baltimore, 2004.

[5] C. A. da Costa, A. C. Yamin, and C. F. R. Geyer. Toward a general software infrastructure for ubiquitous computing. *IEEE Pervasive Computing*, 7(1):64–73, 2008.

[6] A. Dey, D. Salber, and G. Abowd. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction*, 16, 2001.

[7] P. Fahy and S. Clarke. Cass - middleware for mobile context-aware applications. *MobiSys - International Conference on Mobile Systems, Applications, and Services*, 2004.

[8] D. Fensel, W. Wahlster, H. Lieberman, and J. Hendler. Spinning the semantic web: Bringing the world wide web to its full potential. *The MIT Press*, 2005.

[9] T. Gu, H. Pung, and D. Zhang. A middleware for building context-aware mobile services. In *Proceedings of IEEE Vehicular Technology Conference*, Milão, Itália, Maio 2004. IEEE Press.

[10] K. Henricksen and J. Indulska. Developing context-aware pervasive computing applications: Models and approach. *Pervasive and Mobile Computing*, 2(2):37–64, 2006.

[11] T. Hofer, W. Schwinger, M. Pichler, G. Leonhartsberger, and J. Altmann. Context-awareness on mobile devices - the hydrogen approach. 2002.

[12] B. McBride. Jena api - a semantic web framework for java, 2007. Disponível em: <http://jena.sourceforge.net/ontology/>. Acesso em julho de 2008.

[13] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing. 2nd ed*. Cambridge University Press, Cambridge, UK, 1992.

[14] R. A. Real. Tips, uma proposta de escalonamento direcionada à computação pervasiva, 2004.

[15] M. Roman, C. Hess, R. Cerqueira, A. Ranganat, R. Campbell, and K. Nahrstedt. Gaia: A middleware infrastructure to enable active spaces. *IEEE Pervasive Computing*, Outubro 2002.

[16] A. Seaborne. Sparql - a query language for rdf, 2007. Disponível em: <http://www.w3.org/TR/rdf-sparql-query/>. Acesso em julho de 2008.

[17] T. Strang and C. Linnhoff-Popien. A context modeling survey. In *Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing*, Nottingham, England, 2004. UbiComp.

[18] M. Weiser. The computer for the 21st century. *Scientific American*, 3(265):94–104, Setembro 1991.

[19] A. Yamin. *Arquitetura para um Ambiente de Grade Computacional Direcionado às Aplicações Distribuídas, Móveis e Conscientes do Contexto da Computação Pervasiva*. Tese (doutorado em ciência da computação), Instituto de Informática, UFRGS, Porto Alegre, RS, 2004.

[20] A. C. Yamin, I. Augustin, J. Barbosa, L. C. da Silva, R. A. Real, A. S. Filho, and C. F. R. Geyer. Exehda: Adaptive middleware for building a pervasive grid environment. *Frontiers in Artificial Intelligence and Applications - Self-Organization and Autonomic Informatics*, 135:203–219, 2005.