Explorando Sensibilidade ao Contexto com MoCA

Vilnei Marins de Freitas das Neves, Adenauer Corrêa Yamin, Nelsi Warken

Centro Politécnico – Universidade Católica de Pelotas (UCPEL)

vilneineves@gmail.com, {adenauer, nelsi}@ucpel.tche.br,

Resumo. *Este trabalho tem por objetivo central explorar as funcionalidades e tecnologias relativas a Sensibilidade ao Contexto, disponibilizadas pelo middleware MoCA (Mobile Collaboration Architecture) e relacion -las as utilizadas pelo middleware EXEHDA.*

1. Introdução

Este trabalho está inserido nos esforços de pesquisa do G3PD/UCPEL em Computação Pervasiva, a qual prevê que uma aplicação e/ou o ambiente de execução, proativamente monitoram e controlam as condições do contexto. O sistema computacional (middleware e aplicação) reage às alterações no contexto através de um processo de adaptação. A proposta da Computação Pervasiva pode ser construída pela integração da computação móvel, computação em grade e computação sensível ao contexto. Em um ambiente de computação pervasiva, os dispositivos, serviços e agentes devem ser conscientes de seus contextos e automaticamente adaptar-se às suas mudanças, isso caracteriza a sensibilidade ao contexto (YAMIN, 2004).

2. O middleware EXEHDA

O middleware EXEHDA faz parte dos esforços de pesquisa do Projeto ISAM. O ISAM vem sendo desenvolvido por um consórcio de universidades gaúchas, e foi iniciado na UFRGS sob a coordenação do Prof. Cláudio Geyer (ISAM, 2007).

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, sob este ambiente, das aplicações que expressam a semântica siga-me. Estas aplicações são distribuídas, móveis e adaptativas ao contexto em que seu processamento ocorre, estando disponíveis a partir de qualquer lugar, todo o tempo (LOPES, 2007).

3. O middleware MoCA

A contribuição desse trabalho dá-se através do estudo do *middleware* MoCA (Mobile Collaboration Architecture), onde foram analisadas as suas tecnologias e funcionalidades pertinentes a Sensibilidade ao Contexto. O MoCA (Rubinsztejn et al, 2004, Sacramento et al. 2004) é um *middleware* baseado em serviços, flexível e extensível, constituído de um conjunto de serviços e API's, que visam dar suporte ao desenvolvimento de aplicativos distribuídos sensíveis ao contexto. Seus aplicativos são direcionados a operarem em uma infra-estrutura de rede local, com suporte inclusive a conexões sem fio (IEEE 802.11b/g). Ele fornece meios para recolhimento, armazenamento e tratamento de dados dos diferentes contextos aos quais os dispositivos estão submetidos (VITERBO et al., 2006).

3.1. Componentes do MoCA

Ao longo do estudo desenvolvido, foi possível identificar que o middleware possui em sua estrutura principal um conjunto de componentes, os quais são descritos a seguir (Rubinsztejn et al, 2004, Sacramento et al. 2004):

Context Information Service (CIS): capaz de receber, armazenar e processar informações de contexto enviadas pelos monitores em execução nos dispositivos móveis, que podem ser acessadas de forma síncrona ou assíncrona.

Location Inference Service (LIS): responsável por inferir a localização aproximada de um dispositivo móvel, através da comparação do padrão de radiofrequência atual de um dispositivo, com padrão de sinais mensurados em pontos de referência pré-definidos, sendo possível definir regiões simbólicas, feitas por uma associação de nomes a regiões físicas definidas (ex: prédios, salas).

Symbolic Region Manager (SRM): Permite estabelecer relações entre as regiões atômicas definidas pelo LIS, caracterizando uma hierarquia entre as mesmas.

Configuration Service (CS): é responsável por armazenar e gerenciar informações de configuração de cada dispositivo utilizadas pelo CIS, armazenando-as em tabelas hash, e indexadas pelo endereço MAC (*media access control address*) do dispositivo, onde cada entrada na tabela guarda os endereços do servidor CIS e de um DS, bem como a periodicidade do envio das informações do dispositivo.

Discovery Service (DS): Armazena informações como nome, propriedade e endereço de qualquer aplicação e seus componentes ou serviço registrado e disponibiliza a descoberta de seu endereço por parte de seus clientes.

Monitor: É executado em cada um dos dispositivos móveis. É responsável por armazenar dados relativos ao estado e ambiente de execução do dispositivo e enviá-los para o CIS. Dentre os dados coletados estão incluídos dados relativos a qualidade da conexão, energia, uso de CPU, uso de memória, etc. É o único desenvolvido em C++.

3.2 Configuração do middleware MoCA

Após a instalação dos componentes básicos do MoCA, que deve ser feita considerando as orientações disponíveis no site do projeto, pode ser realizada a configuração dos serviços do middleware respeitando uma série de dependências funcionais, que são necessárias antes de se executar uma aplicação propriamente dita (MoCA, 2008).

Configuração do serviço CIS: Antes do serviço CIS ser iniciado, é necessário que seja criado um arquivo com o nome de cis.properties, salvo no diretório "CIS_HOME\conf", contendo o endereço do servidor externo e as portas de comunicação que serão usadas para atender as requisições dos clientes, como é mostrado na Figura 1.

```
cis.monitor.port=55010
cis.event_server.port=55000
cis.server.port=55001
cis.server.address=0.0.0.0
```

Figura 1: Exemplo de Arquivo cis.properties.

Configuração do LIS: Para a configuração do LIS, devem ser criados três arquivos, salvos no diretório "LIS_HOME\conf". O primeiro, chamado cis.properties, contém informações sobre o servidor externo CIS, como endereço IP e portas de comunicação, de modo que o LIS possa se registrar (vide Figura 2).

```
cis.monitor.port=55010
```

```
cis.event_server.port=55000
cis.server.port=55001
cis.server.address=0.0.0.0
```

Figura 2: Exemplo de Arquivo *cis. ro erties* utilizado pelo LIS.

O segundo arquivo, chamado *srm.properties*, descreve os parâmetros do servidor SRM, permitindo que o servidor LIS possa conectar-se e obter informações sobre regiões simbólicas e a hierarquia entre as mesmas. A Figura 3 apresenta o conteúdo do arquivo *srm.properties*.

```
srm.server.host=192.168.1.104
srm.query.port=55030
srm.event.port=55031
```

Figura 3: Exemplo de Arquivo srm.properties.

O terceiro e último arquivo, chamado lis.properties, possui em seu conteúdo informações sobre os parâmetros do servidor LIS, como endereço IP e portas de comunicação que devem ser usadas para atendimento das requisições dos clientes (vide Figura 4).

```
lis.server.host= 192.168.1.102
lis.server.port=55021
lis.publisher.port=55020
```

Figura 4: Exemplo de Arquivo lis.properties.

A partir desse momento, para executar o LIS, basta invocar o comando "lis" através do terminal.

Configuração do CS: O procedimento de configuração do CS se dá através da criação e edição do arquivo cs.properties. Esse arquivo deve ser salvo no diretório indicado pelo variável de ambiente MOCA_HOME e deve conter o endereço IP do servidor onde o CS será executado, como indicado na Figura 5.

```
cs.ip_address=192.168.1.104
```

Figura 5: Exemplo de Arquivo lis.properties.

O CS será iniciado através do comando "java -jar cs-rc2.jar", executado através do terminal. Nesse momento, o CS está pronto para adicionar ou remover dispositivos de sua base de dados, bem como requisitar informações sobre um dispositivo.

Configuração do DS: De forma semelhante a configuração do CS, a configuração do DS é feita através da criação de um arquivo chamado *server.properties* contendo o endereço do servidor onde o DS está executando e a porta usada para comunicação (vide Figura 6).

```
ds_ip = 192.168.1.104
ds_port=55017
```

Figura 6: Exemplo de Arquivo server.properties.

Para iniciar a execução do serviço DS, o comando "java -cp ds-rc1c.jar moca.service.ds.slp.Da " deve ser invocado no terminal.

Configuração do Monitor: Para configurar o Monitor, no diretório "C:\Windows\MonitorXP" deve ser criado o arquivo monitor.properties, que contém os endereços IP e número das portas dos servidores CS, CIS e DS, além do intervalo de

tempo entre as requisições e o número da porta do próprio Monitor. A Figura 7 apresenta o conteúdo original do arquivo monitor.properties.

```
CIS_ADDRESS = cis.lac.inf.puc-rio.br
CIS_PORT = 55010
CIS_PERIODICITY = 3
CS_ADDRESS = cs.lac.inf.puc-rio.br
CS_PORT = 55016
DS_ADDRESS = ds.lac.inf.puc-rio.br
DS_PORT = 55017
DEFAULT_CARD_QUERY_INTERVAL
=250 MONITOR_PORT = 61555
```

Figura 7: Exemplo de Arquivo monitor.properties.

4. Considerações Finais e Trabalhos Futuros

Os estudos realizados até agora caracterizaram o middleware MoCA como uma estrutura vocacionada para dispositivos móveis, e para o desenvolvimento de aplicações com foco em localização. Os estudos também mostraram que as tecnologias empregadas no MoCA, particularmente o sensoramento da localização mostram-se interessantes para uso no EXEHDA.

Como trabalhos futuros espera-se implementar aplicações com a mesma funcionalidade nos middlewares EXEHDA e MoCA, tendo como objetivo uma comparação mais profunda dos diferentes aspectos de programação e execução das soluções em ambas plataformas. Além disso, buscar-se-á sistematizar uma comparação entre os middlewares EXEHDA e MoCA, contemplando aspectos de modelagem, implementação e perfil operacional, com o intuito de avaliar os aspectos do MoCA enquanto middleware para suporte a Computação Sensível ao Contexto, passíveis de serem empregados na qualificação das funcionalidades do EXEHDA.

5. Referências

Lopes, J. L. B. EXEHDA-ON: Uma Abordagem Baseada em Ontologias para Sensibilidade ao Contexto na Computação Pervasiva. [S.l.]: Universidade Católica de Pelotas - Programa de Pós-Graduação em Informática, 2007. Technical Report 2007-01.

MoCA. Mobile Collaboration Architecture Home Page. Disponível em http://www.lac.inf.puc-rio.br/moca/.

Sacramento, V., Endler, M., Rubinsztejn, H. K., Lima, L. S., Gonçalves, K., Nascimento,F. N., And Bueno, G. A. (2004). MoCA: A middleware for developing collaborative applications for mobile users. IEEE Distributed Systems Online, 5(10).

Viterbo Filho, J.; Sacramento, V.; Rocha, R. C. A.; Endler, M. (2006). MoCA: Uma Arquitetura para o Desenvolvimento de Aplicações Sensíveis ao Contexto para Dispositivos Móveis. In: Simpósio Brasileiro de Redes de Computadores, Sessão de Ferramentas. 5, 2006. Anais do SBRC 2006.

Yamin, A. (2004). "Arquitetura para um Ambiente de Grade Computacional Direcionado as Aplicações Distribuídas Móveis e Conscientes do Contexto da Computação Pervasiva". Tese de Doutorado, Universidade Federal do Rio Grande do Sul, Porto Alegre, Brasil.