Middleware Reconfigurável para Telefones Celulares

Aluno: Vitor Paulo Villarino Pinto

Orientador: Prof. Dr. Edmundo R. Mauro Madeira

IC – UNICAMP

email: vepevaronil@gmail.com, edmundo@ic.unicamp.br

CNPq/PIBIC/SAE/UNICAMP

middleware - telefones celulares

Introdução

O desempenho dos aplicativos de telefonia móvel é afetado diretamente pelas mudanças do meio, como memória, CPU e disponibilidade de rede. Mas, devido à falta de suporte dos middlewares, a maioria das aplicações nem reage a essas mudanças.

Este projeto visa mudar esse panorama, através de dois estudos: a utilização de um middleware baseado no modelo de componentes OpenCom [**Coulson,08; Ueyama,05**] em ambientes J2ME (Java Micro Edition) e o desenvolvimento de aplicativos reconfiguráveis para este ambiente.

OpenCOM

OpenCom é um modelo de componentes genéricos capaz de construir sistemas de softwares para uma variedade de domínios como middlewares, sistemas operacionais e ambientes de redes de sensores.

Este modelo baseia-se em instanciações e conexão de componentes. Basicamente existem cápsulas, que são repositórios onde os componentes são carregados (*loading*), instanciados e contidos. Os desenvolvedores constroem sistemas instanciando componentes e estabelecendo conexões (*binding*).

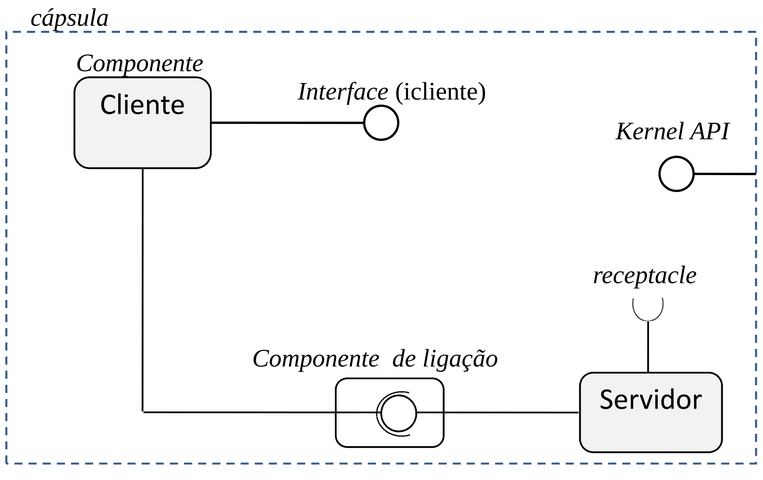

Figura 1. Modelo de Programação

Os componentes possuem *interfaces* que expressam suas funcionalidades e *receptacles* que são as interfaces requeridas de outros componentes. Quando uma *interface* e um *recptacle* estabelecem uma conexão (*binding*), um novo componente é criado. (Figura1)

Migração do OpenCOM para J2ME

Para este trabalho, utilizamos uma versão “light” do OpenCom (desenvolvido pela Universidade de Lancaster) . Esta versão utiliza um *overhead* bem menor que a versão normal, tornando-o perfeito para a utilização em ambientes altamente limitados como o de telefonia móvel.

A etapa mais trabalhosa desta migração foi a falta dos métodos reflexivos (métodos que retornam informações sobre objetos e classes como atributos públicos, interfaces, permissões, etc, presentes majoritariamente em ***java.lang.reflect***) não estão presentes na configuração J2ME-CLDC.

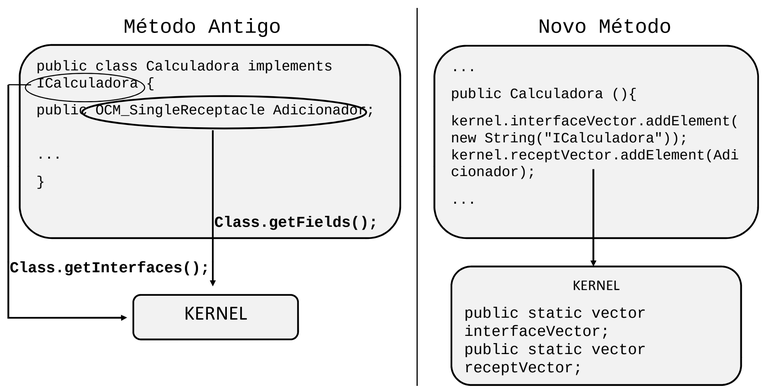

Figura 2. Diferenças dos Métodos

No OpenCom, estes métodos eram chamados durante o processo de instanciação para descobrir quais eram os receptáculos e interfaces de um componente. Sem estes métodos, a maior questão a ser resolvida era como conseguir essas informações.

A melhor solução encontrada foi a de passar esta tarefa para o desenvolvedor do componente (não o desenvolvedor final), que terá que transferir as informações durante o construtor do componente (Figura 2).

Cenário de Aplicações

Durante este trabalho foram produzidos alguns aplicativos que confirmassem a nossa plataforma. Dentre eles, o mais interessante é um aplicativo reconfigurador entre Bluetooth (utilizando a API JSR-82 da Sun)e Wi-Fi (IEEE 802.11).

Cada uma das tecnologias possui um módulo principal (BT e Wi-fi), que possui duas interfaces (*IClient* e *IServer*), onde os componentes de cliente e servidor se ligam. O módulo principal, além de coordenar as mensagens, se reconfigura através da disponibilidade da rede e da quantidade de bateria disponível. (Figura 3)

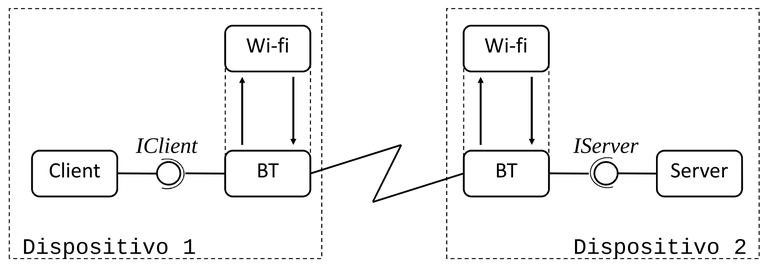

Figura 3. Reconfigurador

Testes com a Plataforma

Para confirmar a tecnologia OpenCom, realizamos diversos testes tanto em ambientes simulados (emuladores virtuais) quanto em ambientes reais. Utilizamos nos testes virtuais os emuladores Su Sun Wireless Toolkit 2.5 para CLDC e Sony Ericsson CDC Platform, e para os testes reais um celular Samsung SGH – G600 e um sensor Gumstix Connex 400xm-bt. Os resultados estão nas tabelas abaixo.

Tabela 1. Teste Virtual

	OpenCom – RUNES	OpenCom – Light - CLDC	Java
Overhead Mínimo	3Kb 120 bytes	132 bytes	-
Overhead para um Comp. Nulo	992 bytes	609 bytes	-
Tempo para Instanciar o Kernel	85ms	20ms	-
Tempo de Load de Comp. Nulo	20ms	1µs	97µs
Instanciar Comp. Nulo	160µs	140µs	2.3µs

Tabela 2. Teste Real

	OpenCom – Light – CLDC		Java	
TESTE	**SGH-G600**	**Gumstix**	**SGH-G600**	**Gumstix**
Instanciar o Kernel	35ms	475ms	-	-
Load de Comp. Nulo	135µs	433µs	400µs	2,33ms
Instanciar Comp. Nulo	380µs	4,73ms	15µs	133µs

Conclusão

Através deste trabalho demonstramos que a reconfiguração é útil em telefones celulares e que o OpenCom se demonstrou uma plataforma flexível e eficiente na produção de aplicativos para este ambiente.

Referências

[Ueyama,05] – Ueyama J., “A Runtime Component Model for System Software”. PhD thesis, Lancaster University, December 2005

[Coulson,08] – Coulson G., Blair G., Grace P., Taiani F., Joolia A., Lee K., Ueyama J.,Sivaharan T., “A Generic Component Model for Building Systems Software”, ACM Transactions on Computer Systems (TOCS). Volume 26 , Issue 1, February 2008