

7

Conclusão

O propósito desta tese foi facilitar o desenvolvimento de aplicações para disseminação de conteúdo para clientes móveis, com adaptação de conteúdo sensível a contexto para comunicação publish/subscribe, tendo como foco questões de eficiência e escalabilidade das adaptações.

Primeiramente, foram identificados vários requisitos que devem ser atendidos por aplicações de disseminação de conteúdo para clientes móveis. A partir destes requisitos, foi projetada uma arquitetura de referência flexível, com uma estrutura modular que permite a separação das diferentes funcionalidades necessárias para esse tipo de aplicação. Foi também proposto um algoritmo (*Context-Aware Client Grouping Algorithm*) para lidar com a questão de eficiência e escalabilidade do sistema. Posteriormente, esse algoritmo, que otimiza a execução das adaptações de conteúdo identificando clientes com contextos comuns, teve seu desempenho avaliado. Para facilitar o desenvolvimento das aplicações de disseminação de conteúdo, foi desenvolvido o *ProxyFramework* para criação de proxies responsáveis pela adaptação de conteúdo, e que implementa as principais funcionalidades da camada de adaptação da arquitetura de referência. Finalmente, para avaliar a usabilidade do *ProxyFramework*, foi feita uma avaliação qualitativa, através de questionários com desenvolvedores de protótipos de aplicação de disseminação de conteúdo, no qual pode-se constatar os benefícios do framework como forma de reduzir o esforço de implementação deste tipo de aplicação.

7.1

Contribuições

As principais contribuições desta tese são: i) o projeto de uma arquitetura de referência para aplicações de disseminação de conteúdo para clientes móveis; ii) algoritmos para otimização da execução de adaptações de conteúdo; e iii) um framework para facilitar o desenvolvimento de aplicações sensíveis ao contexto.

Arquitetura para aplicações de disseminação de conteúdo sensíveis

a contexto: No projeto da arquitetura definiu-se camadas para as diferentes funcionalidades necessárias para o desenvolvimento de aplicações para clientes móveis cujo conteúdo disseminado deve ser adaptado segundo características dos dispositivos e do canal sem fio. Esta arquitetura satisfaz os principais requisitos de aplicações de disseminação de conteúdo para clientes móveis, que incluem uma comunicação *push* (do tipo publish/subscribe), customização do conteúdo, e transparência aos clientes da realização de adaptações.

Na arquitetura, a divisão em camadas de comunicação, de inferência de contexto e de adaptação de conteúdo permite o desacoplamento dessas funcionalidades e a separação de responsabilidades, podendo ser desenvolvidas independentemente. Dessa forma, também permite, pelo menos em teoria, que diferentes middlewares de comunicação e de provisão de contexto sejam integrados à arquitetura para prover esses serviços a aplicações de disseminação de conteúdo. Além disso, a camada de adaptação é estruturada em módulos que podem ser instanciados conforme a necessidade da aplicação e de acordo com a capacidade do nó de processamento.

Segundo as pesquisas realizadas, esta é a primeira arquitetura projetada e implementada que provê suporte a adaptação de conteúdo para aplicações baseadas no paradigma de comunicação publish/subscribe.

Algoritmo para agrupamento de clientes baseado em contexto: Como as adaptações de conteúdo são operações que em geral utilizam muitos recursos computacionais (processamento e memória), sua execução representa o ponto crítico de desempenho em sistemas *push*. Foi elaborado um algoritmo para otimização da execução das adaptações, chamado *Context-Aware Client Grouping Algorithm*, em duas versões: uma para adaptações não-parametrizadas e outra para adaptações parametrizadas por contexto. Para lidar com a questão de eficiência da execução das adaptações para um grande número de clientes, esses algoritmos utilizam as informações de contexto para agrupar clientes com contextos semelhantes e efetuar as adaptações para os grupos ao invés de executá-las individualmente. Deve-se notar que a formação de grupos é dinâmica e re-calculada a cada envio de mensagem.

Diversos experimentos foram realizados para avaliar o desempenho desses algoritmos, levando em consideração o tempo de serviço em relação ao aumento do número de clientes e de adaptações. Esses experimentos demonstraram o bom desempenho e escalabilidade de sistemas que utilizam os algoritmos com grupos, nos quais o aumento do número de clientes não implica necessariamente em um aumento do tempo de serviço. Pôde-se observar que o desempenho do

algoritmo com grupos depende do grau de compartilhamento de adaptações, que é definida pelas situações de contexto comuns dos clientes. Quanto maior o número de clientes com contextos semelhantes, maior o compartilhamento de adaptações, e melhor o desempenho. Por outro lado, quanto maior o número de regras de adaptação, provavelmente, menor será o ganho obtido pelo algoritmos com grupos. Por isso, é importante que haja uma solução de compromisso entre a maior especificidade das regras que definem as situações de contexto e o desempenho do sistema.

Framework para gerência de adaptações de conteúdo: O ProxyFramework foi desenvolvido para facilitar a implementação da camada de adaptação proposta na arquitetura de referência. Com esse framework, pode-se desenvolver um proxy responsável por todas as funcionalidades necessárias à adaptação. As funções de aquisição e gerenciamento de contexto dos clientes, seleção das adaptações adequadas e execução destas, e o armazenamento temporário de mensagens devido a desconexão de clientes correspondem às partes concretas do framework. O ProxyFramework pode ser utilizado para aplicações com comunicação do tipo publish/subscribe, bem como, request/reply. A definição das regras de adaptação e a implementação de adaptadores de conteúdo e de políticas de armazenamento (*buffering*) constituem os principais pontos de configuração e extensão do framework.

O ProxyFramework fornece mecanismos simples para incorporação de adaptadores de conteúdo e de políticas de armazenamento temporário (*buffering*), através de interfaces bem definidas de classes base que devem ser estendidas. Adicionalmente, a definição das regras de adaptação é feita através de uma arquivo de configuração XML. Essas características tornam simples sua utilização, e permite que o desenvolvedor se dedique à lógica de negócios da aplicação, tornando transparente ao desenvolvedor a gerência de contextos e execução das adaptações.

A estrutura do framework é modular, permitindo que apenas as funções necessárias à aplicação sejam instanciadas. Além disso, a separação entre gerentes independentes que interagem através de filas de mensagens permite uma execução concorrente eficiente em arquiteturas multi-core (cada vez mais comuns).

Nos últimos semestres, o ProxyFramework têm sido utilizado pelos alunos da disciplina de computação móvel ministrada no DI/PUC-Rio para desenvolver protótipos de aplicações sensíveis a contexto com adaptação de textos e imagens. A partir de questionários respondidos pelos alunos foi possível evidenciar o quanto o framework pode facilitar e agilizar o desenvolvimento de

proxies de aplicações cujo conteúdo precisa ser adaptado de acordo com as características da rede e do dispositivo.

Além destas contribuições, alguns trabalhos, listados a seguir, foram feitos como parte da implementação da arquitetura.

Um serviço de aquisição de informações estáticas de dispositivos foi desenvolvido e incorporado ao CIS da MoCA. As características dos dispositivos móveis, que incluem informações de hardware (tamanho de tela, número de cores, tipo teclado, etc) e software (SO, formatos de dados e imagens suportados, etc), são obtidas de perfis descritos com base no esquema CC/PP e no vocabulário UAProf. Estas informações são de fundamental importância para definição de regras de adaptação, ou para serem utilizadas como parâmetros das adaptações de conteúdo.

Outro componente MoCA, parcialmente projetado e desenvolvido, foi a interface de comunicação Pub/Sub ECI, usada tanto para distribuição de eventos de contexto, quanto para a implementação da infra-estrutura de comunicação da arquitetura para serviços de disseminação de conteúdo. Além disso, houve participação no projeto da arquitetura MoCA como um todo.

7.2

Trabalhos Futuros

A versão atual do ProxyFramework implementa boa parte dos requisitos e componentes da camada de adaptação da arquitetura discutida, tais como aquisição de contexto, ativação de adaptações de acordo com as regras e o contexto do usuário, algoritmos para melhorar a eficiência do sistema, dentre outros. No entanto, pretende-se implantar melhorias ao projeto inicial do ProxyFramework a partir das seguintes atividades futuras:

- Desenvolver o componente de atribuição de prioridades de adaptação, segundo critérios programáveis pelo desenvolvedor do proxy.
- Avaliar os requisitos para adaptação de fluxos de dados (*streams*), para implementar possíveis alterações no framework a fim de adequá-lo a esse tipo de dado.
- Substituir a linguagem proprietária utilizada para definição de regras de adaptação por uma linguagem de regras padronizada, como RuleML, por exemplo.
- Oferecer mecanismos para atualização de regras de adaptação em tempo de execução, de maneira que o administrador possa inserir, remover ou

alterar regras com o proxy em funcionamento, sem a necessidade de reiniciá-lo.

- Desenvolver uma interface gráfica para gerenciamento das regras de adaptação, para facilitar a definição das mesmas no arquivo de configuração em XML. Essa interface deve informar os atributos de contexto disponíveis e suas possíveis combinações (para formação das expressões de contexto), bem como os adaptadores e suas características. Além disso, deve efetuar verificações de inconsistências e conflitos.
- Implementar a interface de interação com outros middlewares publish/subscribe, tais como JMS ou Siena.

Alguns trabalhos futuros relacionados ao serviço de provisão de contexto CIS da MoCA e ao mecanismo de comunicação publish/subscribe (ECI) são listadas a seguir:

- Identificação única do cliente, permitindo mobilidade pessoal;
- Definição de preferências dos usuários quanto à atividades, horários, dispositivo usado;
- Aquisição de informações de contexto de sensores, como temperatura, luminosidade, etc.
- Gerenciamento de mobilidade (*handoff*), no ECI.

Além das questões de aperfeiçoamento do ProxyFramework e da MoCA, existem ainda tópicos de pesquisa a serem investigados para aprimorar a arquitetura e seus componentes.

Primeiramente, formalizar uma descrição das adaptações selecionadas, e dos adaptadores que as realizam. Essa descrição deve conter informações sobre a adaptação realizada, o tipo (formato) de conteúdo a que se aplica, bem como o formato resultante após a adaptação, os possíveis parâmetros de adaptação, custo de execução, dentre outros. Essas informações podem então ser utilizadas para compor as seqüências de adaptadores, escolhendo os adaptadores segundo algum critério (qualidade, custo, tempo de execução, etc).

Os adaptadores de conteúdo são em grande parte restritivos, ou seja, provocam algum tipo de perda de informação, e conseqüentemente, redução do tamanho da mensagem. Assim, para aumentar a eficiência na execução de uma cadeia de adaptadores, seria interessante adotar um mecanismo para melhor escalonar esses adaptadores, executando primeiro aqueles que provocam uma maior redução do dado, tornando, assim, as adaptações seguintes mais ágeis (por atuar sobre um dado menor). Entretanto, esse mecanismo não pode

alterar o resultado final, e assim, é necessário identificar em cada cadeia as adaptações que podem ser aplicadas em uma ordem diferente da sequência original, sem comprometer o resultado final.

Um interessante tópico de pesquisa seria investigar formas de cooperação entre proxies distribuídos para realizar os serviços de adaptação [125]. A política escolha do nó de para adaptação pode então se basear em informações globais do sistema, como carga dos proxies, disponibilidade de largura de banda da rede e custo da comunicação. Além disso, em uma abordagem ortogonal a apresentada, pode-se utilizar de caches com conteúdo adaptado (em múltiplas versões) [126] em conjunto com a escolha dos adaptadores, na definição da sequência de adaptação. Entretanto, esta abordagem só é viável e útil para alguns tipos de aplicações, cujo conteúdo não seja dinâmico (com freqüentes alterações).

Um outro tema interessante a ser investigado é a utilização de análises do comportamento dos clientes para prover mecanismos de reconfiguração automática das regras de adaptação, permitindo ajustes nos parâmetros de adaptação e adequação de QoS, ao longo da utilização do sistema. Alguns sistemas, como [88, 127], baseiam-se em *feedbacks* dos clientes para aprender como melhor adaptar o conteúdo. Trabalhos em inteligência artificial, e técnicas de aprendizado computacional (*machine learning*) podem ser utilizados para este fim.

Por fim, sugere-se que sejam feitos estudos sobre os impactos da instabilidade de contexto nos sistemas com adaptação sensível a contexto, e sobre mecanismos para identificação dessa instabilidade. A oscilação do contexto pode tornar-se um grave problema, pois uma variação no contexto pode ativar uma cadeia de adaptadores, que pode ser executada desnecessariamente, levando a um aumento no uso de recursos (processamento, memória), e até execução inapropriada de adaptações. A solução para este problema provavelmente será obtida pela combinação de mecanismos implementados próximo ao provedor de contexto e ao **ProxyFramework**. Flutuações específicas ao tipo de dado de contexto e forma de sua coleta, deverão ser tratadas no provedor de contexto, por exemplo, com o uso de filtros para suavizar as variações, ou definindo pequenas margens de erro. Enquanto que no **ProxyFramework**, a informação de instabilidade pode ser utilizada na decisão de quando ativar as adaptações. Por exemplo, pode-se adotar formas diferentes de reação a mudanças de contexto, podendo ser reação imediata (*eager*) ou reação atrasada (*lazy*).

Esta tese representa um passo inicial para o desenvolvimento de um sistema mais completo para disseminação de conteúdo customizado para clientes móveis. Ela abordou questões fundamentais de suporte à adaptação sensível ao contexto para aplicações baseadas em comunicação publish/subscribe, mas existem ainda vários aspectos a serem investigados e implementados, alguns dos quais foram discutidos nesta seção.