1
Introdução

A evolução das técnicas de codificação de mídias contínuas e a crescente popularização das redes de acesso em banda larga vêm tornando cada vez mais difundido o uso de aplicações multimídia distribuídas. Tais aplicações consistem na recuperação, transmissão e recepção de conteúdo audiovisual complexo, que normalmente relaciona tipos e instâncias diferentes de objetos de mídia entre si. Por exemplo, aplicações de vídeo sob demanda permitem ao usuário a escolha e visualização de vídeo com áudio, sincronizados no tempo. Em aplicações mais avançadas, vários objetos de mídia podem estar sincronizados no tempo, no espaço, e ainda ser disparados por eventos, como, por exemplo, as ações de interatividade.

Nesses tipos de aplicações distribuídas, o usuário deseja uma experiência satisfatória com os objetos de mídia reproduzidos, o que subjetivamente se traduz na qualidade de exibição individual desses objetos, no tempo de resposta frente a interações e no sincronismo adequado da apresentação como um todo. Esses aspectos qualitativos de uma aplicação multimídia distribuída representam o resultado final de uma extensa cadeia de serviços de processamento, comunicação e armazenamento oferecidos por diferentes elementos interligados por redes de comunicação.

Preencher, dessa forma, as expectativas dos usuários de aplicações multimídia distribuídas e de outras aplicações de alto desempenho significa atender aos requisitos de uso de todo o conjunto de recursos computacionais distribuídos, de *modo fim-a-fim*. Em outras palavras, deve-se garantir que por todo o caminho de comunicação haja recursos suficientes para o encaminhamento e o processamento adequados dos dados pertinentes às aplicações. Essa funcionalidade é alcançada por meio de *mecanismos de provisão de qualidade de serviço* (QoS – *Quality of Service*) aplicados a todos os subsistemas de participantes do serviço.

Mecanismos de provisão de QoS buscam proporcionar algum controle no compartilhamento dos recursos, de forma a prover aos fluxos de dados dos usuários as garantias por eles solicitadas, ao mesmo tempo em que detectam e tentam corrigir possíveis maus comportamentos e falhas. A forma como os mecanismos atuam sobre essas frentes é ditada pelo *conjunto de estratégias de gerenciamento* aplicado sobre cada recurso. As estratégias de gerenciamento de um recurso podem incluir: a(s) forma(s) como as unidades de informação que compõem os fluxos são escalonadas para uso do recurso; as condições de aceitação de um novo fluxo, entre outras.

Particularmente, o conceito de recurso a ser gerenciado muda de acordo com as necessidades de controle e representação que a infra-estrutura de provisão de QoS deve satisfazer. A complexidade de gerenciamento dos recursos, descrita no parágrafo anterior, é normalmente amenizada por meio de abstrações de alto nível que visam simplificar e agilizar a programação e utilização das aplicações com QoS.

Abstrações de alto nível para o controle da QoS escondem detalhes operacionais de difícil entendimento e expressão pelos usuários de aplicações, e até mesmo por boa parte dos desenvolvedores de software. O uso de tais abstrações pode ser estendido para homogeneizar a forma de interação com recursos e, assim, possibilitar a interoperabilidade entre diferentes plataformas, característica essencial em ambientes distribuídos heterogêneos. Por outro lado, como efeito colateral esperado, quanto mais distante dos recursos reais for o nível de abstração, menor será o poder de controle refinado de QoS pelo seu utilizador.

A definição de quais abstrações de recursos serão disponibilizadas por uma infra-estrutura de QoS é decisão de projeto casada à identificação dos seus casos de uso, quando podem ser determinadas quais representações são relevantes para os atores existentes no cenário em questão. Projetistas e operadores de serviços de telecomunicações, administradores de redes, desenvolvedores de software e usuários finais são atores que podem coexistir em um ambiente distribuído.

De uma forma geral, o controle de QoS em abstrações de mais alto nível deve ser recorrentemente repassado a níveis de abstração inferiores, até que o refinamento seja tal que satisfaça o poder de controle requerido, que também pode variar de cenário a cenário. Por exemplo, em ambientes distribuídos de propósito geral, nos quais as aplicações são construídas a partir de diferentes técnicas de

programação distribuída e são disparadas assincronamente, torna-se interessante que o refinamento do controle de QoS atinja o nível mais baixo possível: o de recursos como CPU, filas de rede, etc. Assim, garante-se que o compartilhamento final dos recursos reais será corretamente realizado, independentemente das cargas de trabalho e formas de interação dessa gama heterogênea de aplicações com a(s) infra-estrutura(s) de QoS disponível(is). Além disso, trata-se de um cenário em que todos os atores citados anteriormente podem estar presentes, exigindo acesso a vários níveis de abstração para o controle de QoS. Por outro lado, no caso de ambientes controlados, como ambientes de tempo real e embutidos (sistemas DRE – *Distributed Real-time and Embedded systems*) (Liu, 2004; Kavimandan, 2007), o controle de QoS pode ser realizado pela camada em comum a todas as aplicações do ambiente, o middleware de tempo real. Por isso, normalmente o refinamento do controle de QoS em DREs não desce ao nível dos recursos reais, o que acaba por exigir em contrapartida um rígido monitoramento de todo o sistema distribuído para a identificação de possíveis violações de QoS (West, 2002).

Observa-se, portanto, um desafio complexo na construção de arquiteturas de provisão de QoS fim-a-fim, em torno do suporte a níveis de abstração de recursos que harmonizem toda a heterogeneidade de recursos reais, plataformas de redes, sistemas operacionais e hardware, e, ainda, tratem a escala de distribuição que pode ser crítica em certos casos. As mesmas abstrações devem permitir que a especificação e a manutenção de subsistemas de gerenciamento de recursos se tornem um trabalho colaborativo entre os atores que diferentes cenários podem apresentar. Sem dúvida, há uma demanda por soluções que permitam que esses atores se tornem colaboradores nos processos de configuração e evolução da infra-estrutura de serviços de comunicação com QoS (Bush, 2001).

Outro desafio, não menos complexo, se torna evidente ao ser considerada a evolução das aplicações e de suas técnicas de codificação de dados, o que resulta no surgimento recorrente de novos requisitos de desempenho. Para acompanhar tal evolução, os serviços devem ser flexíveis o bastante de modo a acomodar novas demandas de QoS por meio de adaptações suaves (Kosmas, 1997) em sua infra-estrutura de QoS. Passa a ser uma habilidade essencial, portanto, a *adaptabilidade da infra-estrutura de QoS*, para permitir a implantação rápida (e de baixo custo) de novos serviços. Isso se traduz no suporte à criação e à

modificação dinâmicas de estratégias de gerenciamento dos recursos, em qualquer fase do ciclo de vida dos serviços.

Torna-se claro, nesse contexto, que o gerenciamento de recursos em si é objeto essencial no ataque a esses desafios, principalmente ao considerar-se recurso uma abstração presente em vários níveis. Um dos motivos para que esses desafios ainda estejam latentes nessa área pode ser atribuído ao fato de que cada infra-estrutura de QoS possui sua própria solução de gerenciamento de recursos, e que raramente trata a uniformização nos subsistemas relevantes, considerando os diversos cenários possíveis. Esse mesmo problema tem mantido operadores e administradores isolados dos usuários e desenvolvedores, pois, nos casos em que há essa coexistência, o primeiro grupo de atores utiliza ferramentas e abstrações totalmente desconexas em relação às usadas pelo segundo grupo. Por fim, a falta de uniformização no gerenciamento de recursos dificulta também o suporte a adaptabilidade, pois as estratégias de gerenciamento a serem criadas ou modificadas estão espalhadas por vários níveis de abstração e precisam tirar proveito de uma interface uniforme para que possam ser desenvolvidas e dinamicamente associadas e desassociadas dos recursos.

1.1. Objetivos e organização da Tese

O presente trabalho propõe uma técnica de gerenciamento de recursos capaz de atender aos requisitos de uniformização, adaptabilidade e interoperabilidade exigidos para a provisão adequada de qualidade de serviço (QoS) fim-a-fim. A solução é genérica o bastante para se especializar em quaisquer cenários distribuídos, deixando a cargo dos desenvolvedores de infra-estruturas de QoS ou projetistas de serviços de telecomunicações a identificação dos casos de uso, atores e níveis de abstração dos recursos necessários em seus projetos.

A abordagem adotada para tal solução é inspirada no paradigma de desenvolvimento de software dirigido por modelos (*model-driven software development* - MDSD) (Stahl, 2006). Diversos conceitos e processos de MDSD são neste trabalho mapeados para o domínio de gerenciamento de recursos, no intuito de formular a técnica de *gerenciamento de recursos dirigido por modelos* (*model-driven resource management* – MDRM). Assim, as grandes virtudes de

MDSD, tais como a formalização do desenvolvimento de software a partir de meta modelos, a independência de plataforma para boa parte do código gerado e a facilidade de manutenção, são herdadas por MDRM em seu domínio específico.

É importante salientar que a proposta MDRM melhor se classifica como a adoção de um processo análogo à MDSD para o desenvolvimento de modelos de gerenciamento de recursos, uma vez que certas especificidades desse domínio exigem algumas adaptações no paradigma. MDRM inclui as especificações de um meta modelo próprio exclusivamente voltado para o gerenciamento de recursos e de uma linguagem de domínio específico (DSL – Domain-specific language) capaz de expressar o formalismo do meta modelo em código independente de plataforma.

Tal qual MDSD, MDRM possibilita a construção de ferramentas em total conformidade com o meta modelo, para facilitar os processos de validação, simulação e implantação dos modelos. MDRM vai além de MDSD ao permitir que os mesmos mecanismos usados para a descrição dos modelos de gerenciamento de recursos possam ser usados para modificá-los depois de implantados, ou seja, em tempo de operação. O Capítulo 2 descreve de forma resumida o paradigma MDSD e introduz os conceitos e processos mapeados para MDRM.

Os demais objetivos desta tese compreendem cada um dos requisitos e etapas que definem a especificação completa da abordagem MDRM. O meta modelo proposto é denominado “Árvores de Recursos Virtuais” (Virtual Resource Trees – VRT). A partir do conceito de recurso virtual, que representa uma parcela da capacidade de um recurso, o meta modelo formaliza uma estrutura hierárquica (VRT) que expressa as sucessivas divisões da capacidade de um recurso e, ainda, prevê a associação 1:1 de conjuntos, possivelmente disjuntos, de estratégias de gerenciamento a cada recurso virtual. Dessa forma, o meta modelo VRT possibilita a coexistência de diferentes políticas no gerenciamento de um mesmo recurso, dando suporte a diferentes tipos de aplicações e seus requisitos. As estratégias de gerenciamento são mantidas de forma desacoplada dos recursos virtuais, o que permite adaptações em tempo de projeto e de operação. A funcionalidade fim-a-fim do meta modelo é viabilizada por meio de VRTs compostas, que podem ser usadas para representar em níveis mais altos a conjunção de recursos virtuais advindos de VRTs de níveis mais baixos.

Constroem-se, assim, abstrações de alto nível mais simples e que ocultam, ainda que com a mesma interface, a distribuição e complexidade de outros recursos virtuais. O Capítulo 3 é dedicado ao detalhamento do meta modelo VRT.

Baseado no meta modelo, este trabalho apresenta uma DSL denominada *Pan*, dedicada à descrição de modelos de gerenciamento de recursos baseados em VRTs. Pan é uma linguagem de aplicação XML (Bray, 2006) que recorre ao formalismo de *XML Schema* (Thompson, 2004) e *ISO Schematron* (ISO/IEC, 2006) para cercar-se de ferramentas de validação estrutural, sintática e semântica no processo de especificação de VRTs. Pan permite, sempre que possível, o reuso e a alteração de descrições já criadas e implantadas. A especificação completa da linguagem Pan é assunto do Capítulo 4.

Devido à utilização de técnicas análogas a MDSD, MDRM permite que ambientes de desenvolvimento que ofereçam as ferramentas necessárias para agilizar a criação e manutenção de modelos baseados em VRT sejam construídas. Tais ferramentas incluem editores gráficos, compiladores, transformadores de VRTs e extratores de dados em tempo real. Os aspectos relacionados à construção de um ambiente de desenvolvimento em MDRM são discutidos no Capítulo 5.

A descrição em Pan de um modelo de gerenciamento baseado em VRT deve ser encaminhado a módulos de software responsáveis por realizar as transformações de código Pan para código específico das plataformas de gerenciamento que instanciarão o modelo. Para isso, a infra-estrutura de gerenciamento deve suportar a criação de VRTs. O Capítulo 6 ilustra, por meio de uma solução para sistemas operacionais de propósito geral, como uma infra-estrutura de gerenciamento de recursos reais pode ser adaptada para abrigar VRTs. Particularmente, a solução apresentada, denominada VRT-FS, é capaz de representar todos os conceitos do meta modelo, e manter estreito mapeamento destes com sua estrutura de gerenciamento, baseada em sistemas de arquivos especiais. Assim, as transformações modelo-para-código se tornam mais simples, exatas e eficazes.

Finalmente, o Capítulo 7 é reservado para as considerações finais.