7
Considerações Finais

O último capítulo do estudo apresenta algumas considerações finais sobre o projeto desenvolvido. A primeira seção destaca alguns trabalhos relacionados e os compara com o sistema LuaPS. A seção seguinte apresenta os objetivos alcançados e os resultados obtidos, concluindo o estudo desenvolvido.

7.1
Trabalhos Relacionados

A maioria dos sistemas publish-subscribe existentes não modularizam as diferentes decisões de projeto, sendo o sistema final definido pelo conjunto de decisões tomadas. O sistema LuaPS busca se diferenciar destes sistemas monolíticos, definindo diferentes camadas e módulos que agrupam decisões de implementação relacionadas e individualizam as diferentes áreas do sistema. Podemos destacar, entretanto, outros sistemas que também têm como foco o desenvolvimento dividido em módulos.

O YANCEES (Silva & Redmiles, 2005) busca ser um sistema configurável, extensível e reutilizável. Para isto, o sistema se baseia na utilização linguagens como XML e na definição de plug-ins. O sistema LuaPS não aborda as possibilidades de configuração, assumindo que todas as opções do sistema são definidas na camada de aplicação. A diferenciação entre a camada publish-subscribe e a camada de serviços, entretanto, está bastante relacionada com a definição de plug-ins para a extensão do paradigma publish-subscribe. O YANCEES se baseia na utilização de serviços de notificação já existentes, ao contrário do sistema LuaPS, que implementa uma camada publish-subscribe completa.

O ADEES (Vargas-Solar & Collet, 2002) é outro sistema que suporta flexibilidade e extensibilidade. Para tal, o sistema se baseia na utilização de meta-modelos para a definição de diferentes estratégias de publicação e consumo de notificações. Ao contrário do sistema LuaPS, entretanto, a definição

das estratégias não se baseia na divisão em camadas, combinando decisões das camadas publish-subscribe, serviços e aplicação. O sistema ADEES, por outro lado, suporta adaptação dinâmica de forma mais natural do que o sistema LuaPS, onde serviços particulares precisariam ser desenvolvidos com este propósito.

O REDS (Cugola & Picco) é um sistema publish-subscribe cujo principal objetivo é a reconfiguração dinâmica. Para atingir este objetivo o sistema se baseia na modularização e na divisão em camadas. Ao contrário do sistema LuaPS, entretanto, a divisão em camadas proposta não tem como principal a definição de uma camada de serviços. Apesar disto, a análise e as conclusões referentes à modularização são bastante condizentes com o estudo que desenvolvemos.

7.2 Conclusão

O estudo desenvolvido foi baseado na implementação do sistema LuaPS, um sistema publish-subscribe que utiliza uma tabela hash distribuída como substrato. Embora o estudo nos tenha permitido perceber as vantagens da utilização de uma infra-estrutura escalável, confiável e tolerante a falhas, as principais contribuições do estudo são conseqüência da definição e discussão da arquitetura utilizada, que foca na generalidade, flexibilidade e extensibilidade.

A utilização da linguagem Lua e do ambiente ALua foi bastante condizente com os objetivos do projeto. A linguagem e o ambiente disponibilizam inúmeros recursos visando a flexibilidade e a generalidade. Embora o ambiente tenha características bastante particulares, o aprendizado é fácil e, uma vez entendido os conceitos inerentes à forma de interação, muito poderoso. A primeira conclusão do estudo, portanto, foi a adequação do ambiente escolhido para as camadas inferiores.

Uma importante contribuição do projeto é a definição de camadas. Na maioria dos sistemas existentes, as diferentes decisões de projeto não são individualizadas, sendo o sistema final definido pelo conjunto de decisões tomadas. O sistema LuaPS busca se diferenciar destes sistemas monolíticos, definindo diferentes camadas que agrupam decisões de implementação relacionadas e individualizam as diferentes áreas do sistema.

Ao analisarmos a divisão em camadas, podemos perceber a separação entre a camada publish-subscribe e a camada de serviços, baseada no estudo de sistemas publish-subscribe, na percepção das características básicas

do paradigma de comunicação e nas necessidades das aplicações. A camada de serviços tem como objetivo estender a camada publish-subscribe com funcionalidades que não fazem parte o paradigma básico, mas que são necessários para aplicações publish-subscribe complexas. Os módulos de serviço implementados disponibilizam acesso a notificações antigas, gateway para clientes móveis com conexões persistentes e a tolerância a falhas.

Ao analisarmos a camada publish-subscribe e a estrutura de tópicos e inscrições, podemos perceber uma outra contribuição do projeto. No momento da inscrição de um cliente em um tópico, o cliente deve especificar uma função de publicação, responsável por definir o processo de publicação das mensagens. A utilização de uma função genérica possibilita que o sistema seja utilizado por diferentes aplicações. A implementação genérica da função de publicação possibilita uma definição híbrida do filtro sobre as notificações, uma outra contribuição do projeto. Desta forma, apesar do sistema ser baseado em tópicos, a função de publicação permite o filtro por conteúdo.

A última conclusão do estudo é resultado dos testes de carga, realizados no capítulo referente à camada de aplicação. Ainda que os testes sejam preliminares, pudemos verificar bons resultados no que se refere ao desempenho da linguagem de programação, do ambiente escolhido e da arquitetura definida.

Podemos perceber, portanto, que além do sistema LuaPS desenvolvido, o estudo introduz discussões sobre diversos aspectos da arquitetura definida. Desta forma, as idéias propostas podem servir de inspiração para outros sistemas.