9
Conclusão e Trabalhos Futuros

O Mobile Social Gateway (MoSoGw) é um protótipo de *framework* que tem como objetivo apoiar o desenvolvimento de aplicações sociais móveis, agindo como um *proxy* entre os clientes móveis e as redes sociais. Ele pode ser usado para filtragem, *caching* e distribuição nas redes sociais dos dados de contexto obtidos a partir dos dispositivos móveis, e vice-versa, para a obtenção de forma otimizada de informações de contexto social e de sistema e e repassá-las aos dispositivos móveis.

O MoSoGw não tem qualquer mecanismo *built-in* para o processamento ou correlação das informações de contexto enviadas pelos usuários e os dados das redes sociais. Inclusive, o módulo de sugestão de amigos se baseia em uma regra bastante simples e pode ser melhorado. No entanto, a arquitetura foi concebida para ser genérica e permitir que serviços web de terceiros desempenhem esse trabalho. Na verdade, diversos tipos de filtragem de dados sociais e de contexto e serviços para correlação dessas informações podem ser implementados por outros aplicativos ou serviços web que necessitam apenas assinar os dados de contexto desejados e gerenciados pelo MoSoGw. Uma aplicação de publicidade pode utilizar dados de posição do usuário para entregar propagandas de estabelecimentos próximos ao usuário.

Uma outra limitação da arquitetura proposta neste trabalho refere-se à privacidade das informações e a obtenção automática das informações de contexto no cliente móvel, duas questões importantes que não foram abordadas. Sobre a primeira limitação, a interface de administração provida pelo MoSoGw permite que o administrador do sistema altere as informações de contexto enviadas pelo usuário e, dessa forma, um serviço pode consumir informações falsas e manipuladas. Já a automação se refere à obtenção das informações de contexto de forma transparente, sem que o usuário precise clicar em um botão e enviar ativamente a informação. Neste caso, o provedor de contexto de bateria poderia ser utilizado para controlar a periodicidade da atualização.

O uso inicial do Mobile Social Gateway pelas aplicações de protótipo e os testes de desempenho não são conclusivos e é necessário o desenvolvimento de mais aplicações móveis sociais baseadas no *framework* e explorar mais as

possibilidades de acoplamento de módulos de processamento. O uso combinado do CMS no lado das aplicações móveis e do MoSoGw como ponto de troca de informações com as redes sociais possui um enorme potencial que não foi totalmente explorado pelas aplicações apresentadas na seção 7.4. Além disso, devem ser realizados testes de performance em cenários com uma carga mais elevada e com uma configuração mais robusta em que conjuntos de *boxes* são configurados em dezenas de máquinas para, então, alcançar melhores resultados. Estes testes não foram realizados porque o objetivo da dissertação era ter uma primeira prova de conceito da arquitetura com foco em escalabilidade sem necessariamente ser a solução definitiva para atender as requisições do usuários em larga escala.

9.1 Trabalhos Futuros

Com a finalidade de tornar a aplicação mais genérica e facilitar a adição de novas funcionalidades, algumas partes do *framework* podem ser rescritas para funcionarem como *plugins*: uma nova rede social ou um novo filtro podem ser adicionados sem que haja a necessidade de alteração do código fonte em si mas apenas da incorporação em tempo de execução dos novos componentes que seriam, então, carregados pelo *framework* sem que seja necessária a reinicialização da aplicação ou um trabalho de rescrever o código para incorporar essas funcionalidades. Essa característica pode ser interessante no caso discutido na seção 7.2: a atualização de uma rede social, caso a integração com ela fosse feita através de um *plugin*, demandaria apenas a atualização desse *plugin* e não de toda a instância do framework permitindo ao desenvolvedor da aplicação social realizar a atualização quando desejar.

Algumas questões sobre a escalabilidade podem ser estudadas em pesquisas futuras. O memcached se comporta como uma tabela *hash*, o que significa que também pode ser utilizado de forma distribuída e, assim, também ser escalado. Cabe à biblioteca Python utilizada pelo MoSoGw a tarefa de controlar da distribuição da carga entre as instâncias de memcached de forma transparente para o desenvolvedor.

A figura 9.1 demonstra essa arquitetura distribuída. Nela, o usuário acessa o Mobile Social Gateway através de um balanceador de cargas que o redireciona para o servidor mais adequado. Cada um dos servidores é composto por um nginx que realiza o balanceamento de carga entre diversas instâncias do Tornado (*Python Applicaton Server*) e por uma instância de memcached acessível por todos os demais servidores.

O banco de dados também pode ser escalado através da configuração de

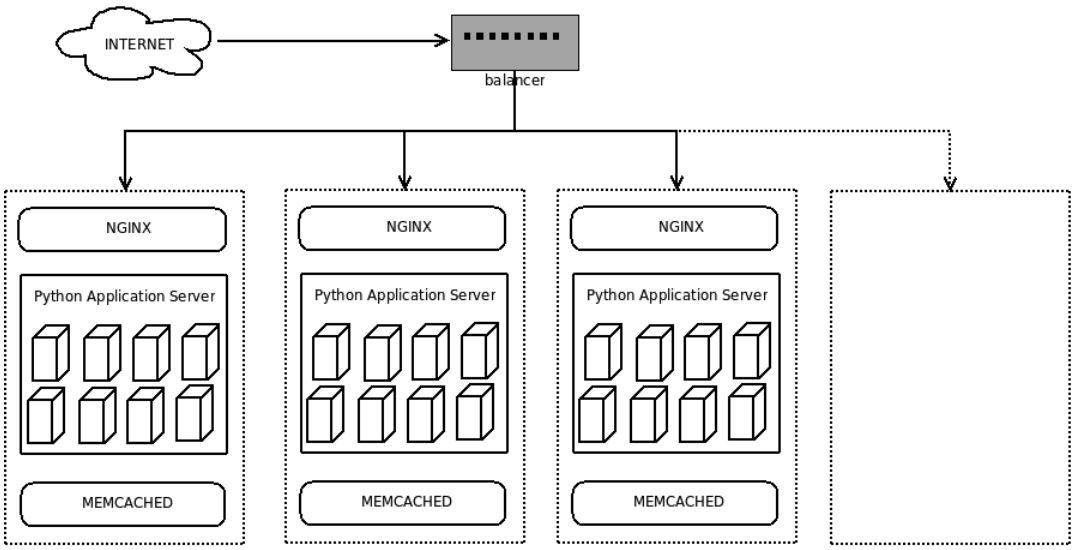

Figura 9.1: Escalando Verticalmente o MoSoGw

múltiplas instâncias de MySQL. O processo de escrita tende a ser mais caro do que o de leitura, uma vez que os dados podem ser armazenados em cache e a leitura pode ser feita a partir deste cache. Para manter a integridade das chaves no MySQL, a escrita só pode ser feita sequencialmente. Já o processo de leitura pode ser paralelizado através da configuração de vários *slaves* de MySQL (supondo uma configuração *master-slave*), a partir dos quais será feita a leitura dos dados, e de um balanceador de carga na frente destes vários servidores (figura 9.2). Isso significa que a aplicação Tornado deverá conhecer apenas o endereço deste balanceador, que distribuirá a carga de leitura entre as instâncias do banco de dados.

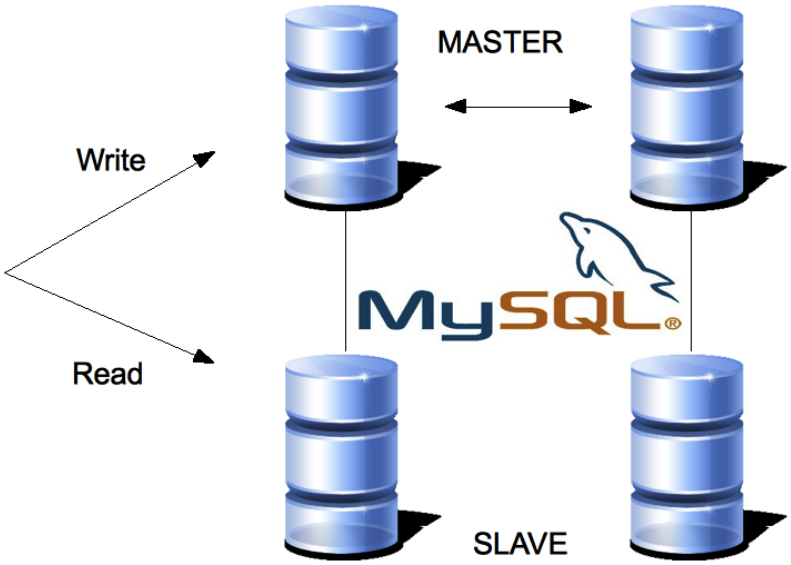

Figura 9.2: Escalando o Acesso ao Banco de Dados