7
Utilização do Mobile Social Gateway

Existem três atores envolvidos na arquitetura do Mobile Social Gateway:

- desenvolvedor do framework MoSoGw: é o responsável pelo desenvolvimento de novas *features*, pela integração com novas redes sociais e pela manutenção das integrações já existentes. Em outras palavras, é o mantenedor do repositório de código;
- desenvolvedor da aplicação social: responsável por instalar e configurar uma ou mais instâncias do MoSoGw e por mantê-las atualizadas em relação ao repositório de código do MoSoGw. Ele deve realizar a integração entre sua aplicação social e sua instância do framework;
- usuário final: é a pessoa que irá utilizar a aplicação social desenvolvida.

O desenvolvedor da aplicação social, ao optar pela utilização do Mobile Social Gateway, ganha os módulos de integração com o Facebook, Twitter e Google Plus. Também pode utilizar o RouteExtractFilter e o FriendsSuggestionFilter. O primeiro filtro se conecta ao Google Maps API e extrai a distância e o tempo de percurso entre dois pontos. O segundo, obtém os amigos do usuário no Facebook e os compara com os usuários existentes no MoSoGw para, então, criar uma lista de amigos sugeridos que é disponibilizada através das APIs. Esse processamento ocorre *offline*: periodicamente, um processo que obtém os amigos no Facebook é executado e a lista de amigos sugeridos é armazenada no memcached para posterior utilização. Ele pode, também, escrever seus próprios filtros mas não poderá agrupar filtros pois a interface de filtros do MoSoGw ainda não está pronta.

Nas seções a seguir, será detalhado como o desenvolvedor de uma aplicação social pode utilizar o framework desenvolvido neste trabalho para integrar sua aplicação às redes sociais e diminuir a complexidade de seu desenvolvimento.

7.1 Context Management Service (CMS)

Inferir dados de contexto periodicamente a partir dos sensores disponíveis nos dispositivos móveis pode implicar em uma complexidade considerável no desenvolvimento das aplicações pervasivas. A programação deve ser orientada a eventos e os casos de exceção precisam ser muito bem tratados para que a aplicação não trave ou encerre sua execução.

Nas aplicações desenvolvidas para a plataforma Android, essa tarefa pode ser consideravelmente simplificada pela utilização do CMS (40), um serviço Android que gerencia a coleta, o processamento e a distribuição de qualquer tipo de dado de contexto. No CMS, cada tipo de dado de contexto é obtido ou produzido por um provedor de contexto (CxP: *Context Provider*) específico e que já possui as rotinas e o tratamento necessários para a obtenção desses dados. O desenvolvedor da aplicação social pode utilizar esses provedores e, então, apenas se subscrever aos dados de contexto através de um processo bastante simples. O serviço também permite que o desenvolvedor da aplicação móvel defina a frequência com a qual as informações de contexto são atualizadas, ou seja, de quanto em quanto tempo os provedores de contexto enviarão a atualização do contexto para a aplicação.

Alternativamente, o desenvolvedor pode escrever seus próprios provedores de contexto, um processo bastante simples e facilitado pela característica modular do CMS, sendo apenas necessária a implementação de uma interface definida pelo serviço e o *deployment* desses novos provedores na nuvem para que possam ser encontrados pelo CMS e utilizados na aplicação móvel. Outra característica do CMS é possibilitar a implementação de provedores de contexto de mais alto nível que agregam, combinam ou processam dados mais básicos de contexto de outros CxPs (por exemplo, localização / tempo = velocidade). Assim, é possível estabelecer um grafo de CxPs, onde alguns CxPs consultam ou assinam as atualizações de dados de contexto de outros CxPs. As aplicações podem, ainda, consultar ou assinar as atualizações de dados de contexto a partir de qualquer CxP. Também é possível definir provedores de contexto que obtêm seus dados de uma fonte remota, como informações sociais de um web service externo.

Um dos provedores de contexto nativos, o provedor de nível de bateria, informa o nível de energia do dispositivo. Essa informação não tem relação direta com o propósito do Mobile Social Gateway e não foi usada nas aplicações desenvolvidas como protótipo. Porém, ela é bastante importante quando o desenvolvedor precisa levar em consideração o nível de bateria em aplicações pervasivas que enviam de forma automática e periódica os dados de contexto.

Embora essa informação não seja necessariamente enviada como dado de contexto para a rede social, com ela é possível regular a periodicidade da atualização das informações de contexto de acordo com o nível da bateria. Outro provedor de contexto que já vem integrado ao CMS é o de localização. Ele obtém a posição geográfica do dispositivo utilizando tanto as redes 3G quanto o sinal do GPS.

Além dos CxPs básicos já disponíveis no CMS os seguintes provedores de contexto de alto nível foram implementados para suportar as aplicações de protótipo:

- provedores de contexto de localização e de velocidade: esses CxPs informam a localização atual do usuário (GPS ou baseados em células) e velocidade (em km/h), respectivamente;
- provedor de contexto de distância: este CxP calcula a distância (e tempo de deslocamento) entre dois pontos identificados por suas coordenadas geográficas (padrão WSG84). A distância entre os pontos é obtida através da API do Google Maps;
- provedor de contexto de reunião: este CxP acessa o Google Calendar para recuperar informações sobre os eventos programados, tais como localização, lista de participantes e o tempo estimado que cada um levará para chegar ao local do evento.

7.2 Configuração do Mobile Social Gateway

Conforme já dito nos capítulos anteriores, para a aplicação social móvel enviar dados de contexto para diversas redes sociais, ela precisa não só conhecer bem o protocolo de comunicação de cada uma delas mas também implementar seus fluxos de autorização e autenticação, aumentando o custo e o tempo de desenvolvimento. Soma-se a isso a necessidade de uma atualização da aplicação a cada nova rede social criada, o que acontece com bastante frequência atualmente. Nesse sentido, utilizando o Mobile Social Gateway, o desenvolvedor da aplicação social pode focar apenas no desenvolvimento de sua aplicação deixando a tarefa de integração com as redes sociais a cargo do desenvolvedor do *framework*.

A proposta do MoSoGw não é fornecer um serviço de compartilhamento com redes sociais e nem a infraestrutura necessária para tal mas ser um framework a partir do qual aplicações sociais podem ser desenvolvidas. Cada desenvolvedor deve possuir sua própria instância do framework, configurada por ele, através dos seguintes passos:

1. Instalar o *framework* Mobile Social Gateway, incluindo a interface de administração, na configuração desejada, escolhendo a quantidade de servidores que lhe atenda. A instalação também envolve a criação do banco de dados e a carga inicial das informações sobre as redes sociais suportadas por padrão (Facebook, Twitter e Google Plus).

2. Criar aplicações em cada uma das redes sociais: nesse passo, o desenvolvedor irá obter a chave secreta necessária para a implementação do protocolo OAuth e deverá configurar sua instância do MoSoGw com essas chaves. A aplicação descrita na seção 7.4.1, por exemplo, possui suas próprias credencias para cada rede social. Se a aplicação ActiveCal (7.4.2) enviasse dados para as redes sociais, ela também deveria ter suas credencias e, por isso, deveria possuir uma instância diferente do MoSoGw.

3. Decidir pela utilização da API Java (recomendada para aplicações Android) ou pela API REST (recomendada para *web services*).

4. Disponibilizar uma interface de criação e login de usuários no MoSoGw.

5. Consumir o serviço que informa quais as redes sociais e serviços estão disponíveis.

6. Com as informações obtidas no passo anterior, ele é capaz de formatar os dados para correto envio ao MoSoGw, conforme descrito na seção 7.3.

As informações de contexto que serão utilizadas e a forma através da qual elas serão obtidas fica a critério de cada aplicação social que for desenvolvida. O MoSoGw, por padrão, suporta *status* e posição e aplica um tratamento padrão para estes tipos de contexto, criando uma URL do Google Maps e a encurtando com o TinyUrl.

O esforço para enviar dados de contexto apenas para uma rede é o mesmo que para mais de uma pois é o MoSoGw que se encarrega de distribuí-los. O único passo a mais do usuário final é acessar a interface de administração e conceder as permissões de acesso para essas novas redes, disponibilizando o *token* de autenticação para o MoSoGw. Na prática, isso significa que o usuário poderá selecionar essa nova rede sem a necessidade de uma atualização da aplicação social. Recomenda-se a criação de uma interface na qual o usuário pode decidir para quais redes sociais ou serviços deseja enviar os dados de contexto que forem obtidos.

Qualquer nova rede social criada será disponibilizada pelo desenvolvedor do framework para ser utilizada pelas aplicações clientes através das APIs do

MoSoGw. O esforço do desenvolvedor da aplicação social será apenas esperar por uma atualização do framework ou, se desejar, implementar ele mesmo a interface para a nova rede. Porém, caso haja uma mudança no protocolo de alguma rede social que não mantenha compatibilidade com a versão anterior, ou seja, que "quebre"a comunicação e gere um erro no envio ou recebimento de informações, essa rede ficará automaticamente indisponível no MoSoGw e as aplicações sociais serão notificadas através do serviço de descoberta de novas redes. Caberá ao desenvolvedor da aplicação social móvel verificar esse serviço periodicamente.

7.3 Padrão das Informações

O MoSoGw espera que as informações de contexto do usuário sejam enviadas pela aplicação pervasiva seguindo um padrão bastante simples:

```
POST /context?auth=188f0a3e365bf54f67f615b1ca199c59 HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
body: {"application": ["facebook", "twitter"],
"context": {"location": "-22,956, -43,245"},
 {"status": "Testando contexto"}}
```

O corpo da requisição deve conter uma lista de redes sociais ou outros serviços para os quais o usuário deseja enviar os dados de contexto e uma listagem dos contextos em si. No exemplo acima, o usuário selecionou o Facebook e o Twitter. Cabe à aplicação móvel disponibilizar uma interface para que o usuário possa escolher para quais redes sociais ou serviços externos enviar os dados.

7.4 Protótipos

Para demonstrar a utilidade da arquitetura MoSoGw, alguns protótipos foram desenvolvidos e serão apresentados nas seções 7.4.1 e 7.4.2. Ambos os aplicativos utilizam o CMS (40), descrito na seção 7.1, para obter as informações do dispositivo e devem enviar as informações de contexto seguindo um padrão apresentado na seção 7.3.

7.4.1 Mobile Social Share

O Mobile Social Share é um aplicativo que permite aos usuários móveis compartilharem informações de contexto sobre sua posição atual. Ele utiliza o provedores de contexto de posição para obter dados de contexto dos sensores dos dispositivos móveis de diversos usuários e informar as condições de tráfego em diferentes regiões de uma cidade.

Após realizar o cadastro no MoSoGw (figura 7.1), o usuário tem à sua disposição a tela inicial da aplicação é exibida na figura 7.2. Nesta imagem é possível observar que o usuário selecionou o Facebook e o Twitter.

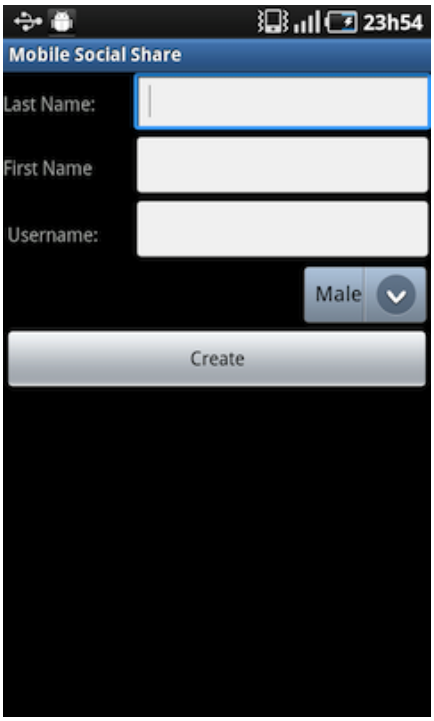

Figura 7.1: Telas de Criação de Usuário

Além das informações de contexto, os usuários podem utilizar um *select box* da GUI (*Graphical User Interface*) para inserir manualmente as informações de contexto (tráfego lento, normal, ou bom) ou escrever algum texto arbitrário sobre a condição de tráfego atual (figura 7.3). A informação é, então, enviada para as redes sociais ou para algum outro serviço externo selecionado pelo usuário. No caso da rede social, todos os seus contatos receberão a informação que ele enviou sobre o trânsito e, assim, poderão seguir um outro caminho mais livre.

O Mobile Social Share exercita o fluxo de informações de contexto do cliente móvel para as redes sociais e também valida o funcionamento da rede social criada pelo MoSoGw. Na tela apresentada na figura 7.4, é possível ver

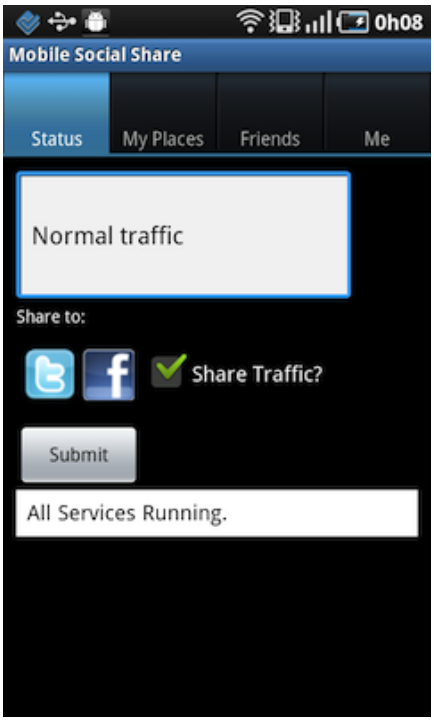

Figura 7.2: Telas de Compartilhamento

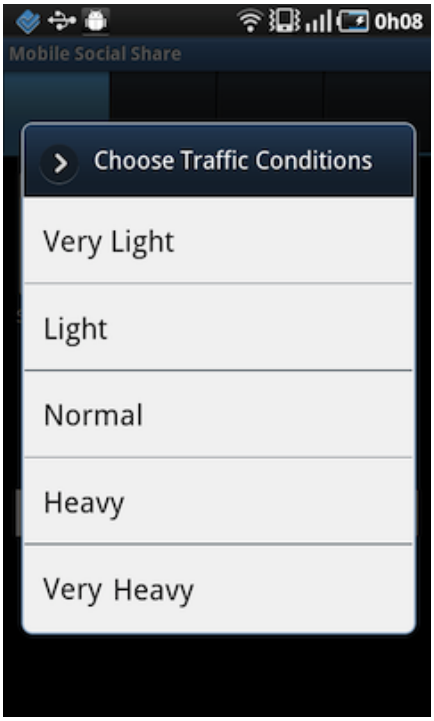

Figura 7.3: Combo de Seleção da Condição do Tráfego

uma das conexões existentes do usuário. Ao selecionar um amigo nesta lista, o usuário obtém informações mais detalhadas (figura 7.5).

A aplicação também utiliza o filtro FriendsSuggestionFilter implementado no MoSoGw para sugerir novos amigos ao usuário.

Os componentes da arquitetura do Mobile Social Gateway utilizados

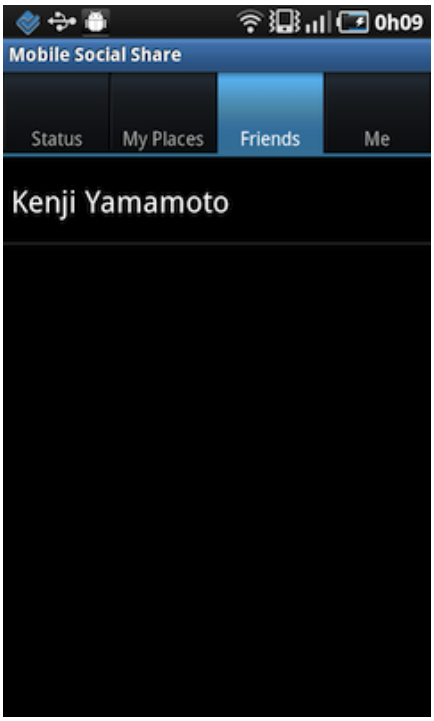

Figura 7.4: Lista de Amigos no Mobile Social Share

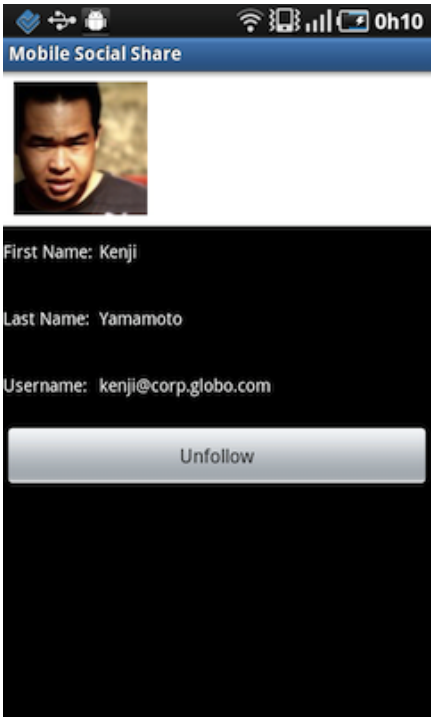

Figura 7.5: Detalhe de um Amigo

estão representados na figura 7.6. A legenda de cada uma das setas está de acordo com a figura 5.3. No fluxo A os relacionamentos do Facebook são obtidos pelo FriendsSuggestionFilter, tratado e repassados para o Mobile Social Share (fluxo B). A posição é obtida utilizado o LocationProvider, disponível no CMS (fluxo H). Essa informação é enviada para o MoSoGw (fluxo C) e distribuído

nas redes sociais (fluxo D).

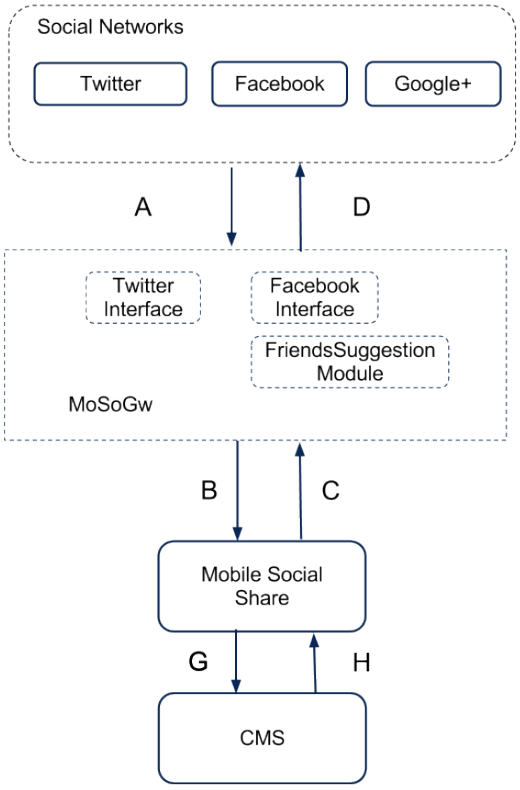

Figura 7.6: Arquitetura do Mobile Social Share

7.4.2 ActiveCal

O ActiveCal é um aplicativo móvel utilizado para monitorar os participantes de um evento. Para cada entrada de calendário representando um evento, todo usuário do ActiveCal é capaz de saber a posição de todos os demais participantes, a distância que falta para o ponto de encontro e a hora estimada de chegada. O ActiveCal se integra ao Google Maps API e ao Google Calendar API. A primeira API fornece a distância entre dois pontos geográficos e as possíveis rotas para o destino. A segunda fornece informações sobre a reunião tais como data, hora e local.

A autenticação dos usuários nos serviços do Google é gerenciada pelo próprio ActiveCal através de um interface de login (figura 7.7 em que o usuário utiliza sua conta do gmail.

Após se autenticar, o usuário pode escolher visualizar todo o seu calendário ou os eventos mais próximos (figura 7.8), que são apresentados em uma lista (figura 7.9 e, para cada evento, são exibidas suas informações (figura 7.10).

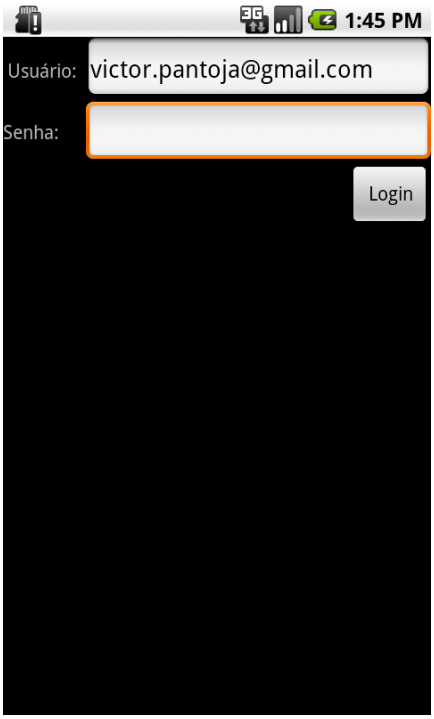

Figura 7.7: Autenticação no ActiveCal

Figura 7.8: Tela Inicial do ActiveCal

A aplicação utiliza o provedor de contexto de posição já integrado ao CMS para obter a posição de cada usuário e o provedor de contexto de reunião para obter o evento do Google Calendar. O provedor de distância recebe os dados obtidos através do dois primeiros para obter a distância e o tempo estimado para o usuário da aplicação chegar ao local do evento. A partir dessa

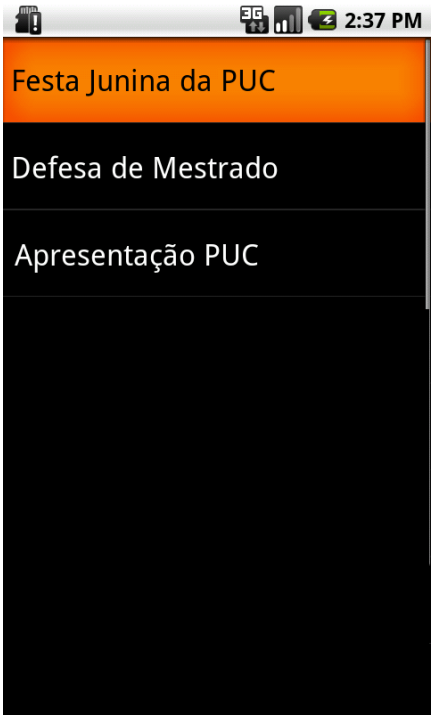

Figura 7.9: Listagem dos Eventos

Figura 7.10: Detalhe de um Evento

informação, o ActiveCal alerta o usuário caso ele esteja atrasado.

Ao contrário da aplicação anterior, o ActiveCal não distribui informações de contexto através de redes sociais, mas exercita a capacidade do MoSoGw em otimizar a transferência de dados dos web services do Google aos clientes móveis. A API do Google Maps, por si só oferece uma grande quantidade de

informações em grande parte sem relevância para o ActiveCal que são filtradas pelo módulo RouteExtractFilter do MoSoGw, de modo a encaminhar apenas as informações relevantes (distância e tempo do percurso).

Conforme visto na figura 7.11, apenas o CMS e o módulo de filtragem são utilizados na arquitetura do ActiveCal. No fluxo E, a rota e a distância são obtidas do Google Maps API e filtradas pelo RouteExtractFilter e repassadas para o ActiveCal (fluxo B). A posição corrente do usuário é obtida pelo LocationProvider, disponível no CMS (fluxo H). A API do Google Calendar não faz parte da arquitetura MoSoGw uma vez que o provedor MeetingContextProvider, presente no CMS, se comunica diretamente com essa API.

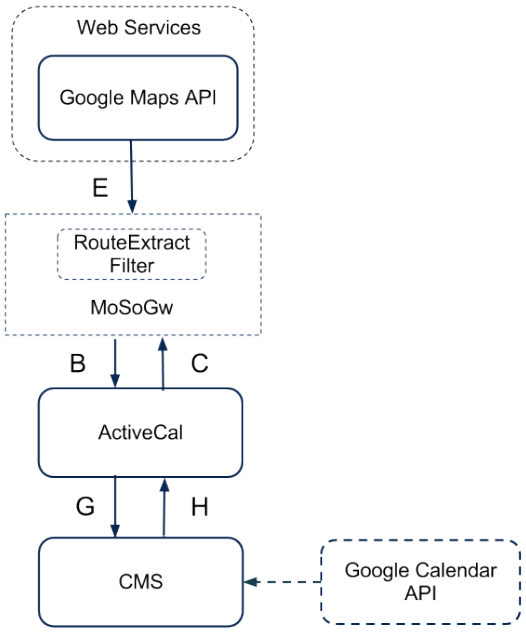

Figura 7.11: Arquitetura do ActiveCal