HyraxMsg: uma aplicação de "messaging" para "mobile edge clouds"

Miguel Silva (up201404978@fc.up.pt)

Departamento de Ciências de Computadores
Faculdade de Ciências da Universidade do Porto

22 de Junho de 2017

Resumo

Em cenários de emergência (ex. acidentes, desastres naturais) ou de multidão (ex. festivais de música, eventos desportivos), existe frequentemente uma dificuldade de comunicação e localização entre pessoas usando dispositivos móveis como "smartphones" e "tablets", devido à ausência de acesso fiável à Internet via Wifi ou 3G/4G. Para cenários deste tipo, é atrativo o uso de "mobile edge clouds", onde dispositivos próximos formam uma rede sem suporte infraestrutural por forma a habilitar comunicação. Com esta motivação, desenvolvemos uma aplicação Android chamada HyraxMsg, que permite a troca de mensagens ("messaging") e o anúncio de localizações entre pessoas usando apenas ligações ponto-a-ponto Wifi-direct e Bluetooth. A aplicação usa internamente o middleware Hyrax para "mobile edge clouds" e GPS para obtenção de dados de localização.

Palavras-chave: "mobile edge clouds", Android, Hyrax

1 Introdução

As "mobile edge clouds" são redes formadas por dispositivos móveis próximos espacialmente. A capacidade conjunta dos vários dispositivos numa rede deste tipo pode ser explorada em termos de processamento, armazenamento e conectividade de rede para habilitar aplicações adaptativas à localização espacial e composição em rede numa lógica de "crowd sourcing" [5]. O seu uso é especialmente relevante quando não há suporte de infraestrutura Wifi/3G para comunicação entre dispositivos, ou para aliviar a carga sobre a infraestrutura existente enquanto providenciando melhor qualidade de serviço em termos de comunicação. Neste contexto podemos ter várias aplicações, tais como a disseminação de vídeos em eventos desportivos [10] ou processamento computacional em paralelo usando a capacidade conjunta de vários dispositivos [9].

Para habilitar "mobile edge clouds", as várias tecnologias de comunicação disponíveis em dispositivos como Wifi, Wifi-Direct ou Bluetooth podem ser utilizadas [8, 7]. A Figura 1 representa de forma esquemática o uso potencial dessas tecnologias em simultâneo por forma a habilitar uma "mobile edge cloud". No contexto do projecto Hyrax (`http://hyrax.dcc.fc.up.pt`) tem vindo a ser desenvolvido um middleware para esse efeito [7], em conjunto com aplicações que o têm vindo progressivamente a integrar. O middleware funciona em dispositivos Android ("tablets" e "smarpthones") e permite de forma automizada a formação de uma rede e posterior comunicação entre dispositivos.

Neste contexto, desenvolvemos uma aplicação que usa o middleware Hyrax para a troca de mensagens ("messaging") e o anúncio de localizações entre pessoas usando apenas ligações ponto-a-ponto Wifi-Direct e Bluetooth, chamada HyraxMsg. Aplicações deste tipo têm tido uso no mundo real, ex. como é o caso da FireChat [2] ou da ZombieChat [12] para cenários de emergência (ex. acidentes, desastres naturais) ou de multidão (ex. festivais de música, eventos desportivos). Nestes cenários onde é frequente a dificuldade de comunicação e localização en-

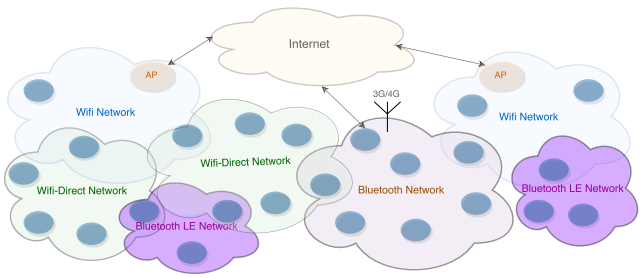


Figura 1: Formação de "mobile edge clouds" usando várias tecnologias (imagem retirada de [7]).

tre pessoas, devido à ausência de acesso fiável à Internet por sobrecarga ou simplesmente inexistência de infraestrutura Wifi ou 3G/4G.

O resto do relatório está estruturado como se segue. Apresentamos primeiro a funcionalidade da aplicação HyraxMsg (Secção 2) e descrevemos a sua implementação (Secção 3). Fazemos de seguida uma discussão de trabalho relacionado (Secção 4), e finalizamos com uma discussão de trabalho futuro (Secção 5).

2 A aplicação HyraxMsg

A aplicação HyraxMsg permite a comunicação entre vários utilizadores e obtenção da respetiva localização e distância relativa num ambiente de "mobile edge clouds". Descrevemos agora a funcionalidade da aplicação em termos da interface com o utilizador, com os vários ecrãs agrupados na Figura 2.

A funcionalidade compreende um ecrã de "login" que dá acesso a uma lista de contactos preenchida à medida que vão sendo descobertos utilizadores na rede. A partir da lista de contactos é possível iniciar uma sessão de conversa com um utilizador específico ou com todos os utilizadores. Foi ainda implementada uma janela de "debug" para efeitos de desenvolvimento. A seguir detalhamos a interação em cada uma destas interfaces gráficos.

2.1 Login

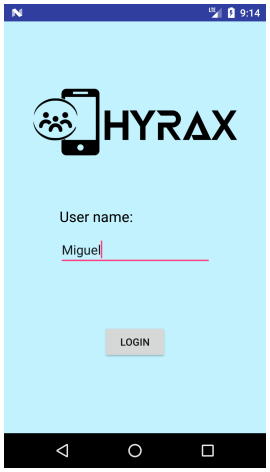
O utilizador começa por se deparar com um ecrã de "login" (Figura 2a). Após introduzir o seu nome, a comunicação em rede é ativada e o utilizador passa a ser anunciado nesta mesma. Não há qualquer autenticação pedida, dado que, tendo em conta o objetivo da aplicação, poderia ser necessário depender de um nó central na rede que tratasse de autenticar permissões e/ou senhas de acesso.

2.2 Lista de contactos

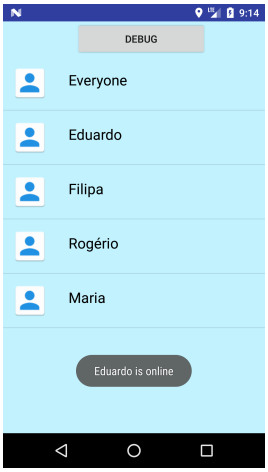
Após o "login", é apresentado ao utilizador uma lista de contactos (Figura 2b) que entretanto vai sendo preenchida dinamicamente em função da descoberta de pessoas na rede. O utilizador pode clicar em cada um dos contactos para iniciar uma sessão de troca de mensagens, e ainda o item "Everyone" para trocar mensagens com todos os contactos disponíveis em simultâneo.

2.3 Chat

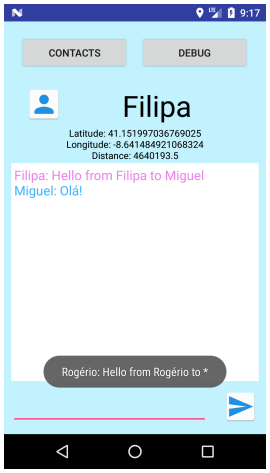
Para realizar as comunicações propriamente ditas entre utilizadores, a aplicação permite abrir uma janela de "chat" privada (Figura 2c) ou também uma janela de



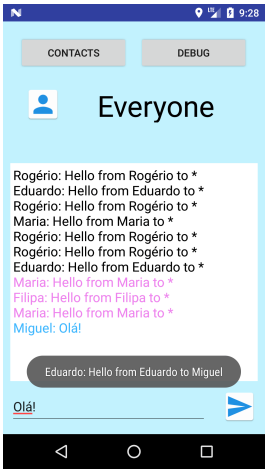
(a) Ecrã de "login".



(b) Lista de contactos.



(c) "Chat" privada



(d) "Chat" pública.

(e) Janela de "debug".

Figura 2: HyraxMsg – interface com o utilizador.

"chat" pública onde é possível comunicar com todos os utilizadores simultaneamente (Figura 2d).

A janela de "chat" privada apresenta o nome do utilizador remoto, assim como as suas coordenadas geográficas e distância a que se encontra, em metros. Um campo de texto e um botão de envio permite ao utilizador enviar uma mensagem ao utilizador remoto. É possível observar a troca de mensagens não só em tempo real, mas também todo o histórico de mensagens desde o início da aplicação. Adicionalmente o botão "Contactos" permite voltar à lista de contactos e o botão "Debug" dá acesso a uma janela de "debugging", discutida abaixo no texto.

A janela de "chat" pública é similar, mas mostra todas as mensagens endereçadas de forma pública por todos os utilizadores.

2.4 Janela de "debug"

A janela de "debug" (Figura 2e) foi implementada apenas para validar a aplicação em desenvolvimento e facilitar a deteção de "bugs". É mostrada uma listagem de mensagens geradas internamente pela aplicação, relacionada com os vários eventos que vão ocorrendo (ex. de rede, acesso a GPS, ou interface c/o utilizador). Além disso, é também possível enviar uma mensagem de texto para a rede na própria janela.

3 Implementação

3.1 Arquitectura

A arquitetura da HyraxMsg está ilustrada na Figura 3. Como se pode observar, a aplicação é constituída pelos seguintes componentes:

- as "activities" Android, que definem a interface gráfica com o utilizador;
- um controlador para reger toda a componente lógica da aplicação;
- um módulo de acesso ao dispositivo GPS;
- três módulos de comunicação, que podem ser alternados para efeitos de desenvolvimento: o principal baseado no middleware Hyrax para uso num contexto de "mobile edge clouds", outro baseado em sockets UDP para testes em ambiente de rede tradicional, e finalmente, para efeitos de simulação, um simulador ("Mock") que se limita a injetar mensagens na aplicação sem qualquer suporte de rede.
- Transversalmente a todos os componentes, é usado um protocolo para as mensagens trocadas entre dispositivos pela aplicação.

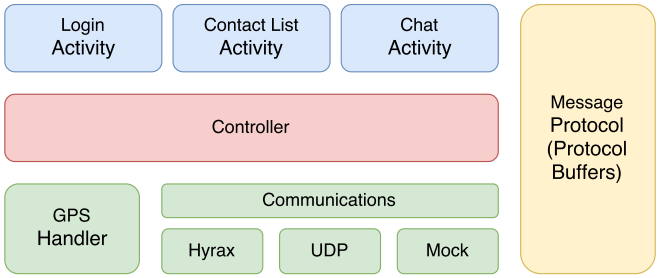


Figura 3: Arquitetura da aplicação.

3.2 Lógica geral da aplicação

Atuando como controlador da lógica geral da aplicação, a classe Controller oferece as funcionalidades listadas na Figura 4. Para interação com o controlador, uma "activity" pode chamar os diversos métodos para envio de mensagens (sendChatMessage()), e para obtenção de vários dados tais como a localização atual (getCurrentLocation(), a lista de pessoas na rede (getPeers()), o histórico de mensagens com determinada pessoa, (getChatHistory()).

Para além disso, cada "activity" implementa a interface Java AppEventObserver, listada na Figura 5, e regista-se como observador de eventos usando o método setObserver. Os eventos concernem aspetos como a atualização de estado (onSelfUpdate, onPeeerUpdate), o envio e a recepção de mensagens de "chat" (onSentChatMessage, onReceivedChatMessage), e atualizações de GPS (onLocationUpdate).

```
public class Controller {
  ...
  // Usado para uma activity se registar e
  // ouvir eventos; cada Activity implementa
  // AppEventObserver
  public
  void setObserver(AppEventObserver observer) {
       ...
  }

  public
  Location getCurrentLocation() { ... }

  public
  void sendChatMessage(String destination,
                       String text) {
    ...
  }

  public
  List<Chat> getChatHistory(String id) {
       ...
  }

   public List<Peer> getPeers() {
     ...
   }
   ...
}
```

Figura 4: A classe Controller.

```
public interface AppEventObserver {
  void
  onInit(Collection<Peer> availablePeers);

  void
  onPeerUpdate(Peer peer, boolean isNew);

  void
  onSelfUpdate(Peer peer);

  void
  onReceivedChatMessage(Chat message);

  void
  onSentChatMessage(Chat message);

  void
  onLocationUpdate(Location location);
}
```

Figura 5: O interface AppEventObserver.

3.3 Uso do Hyrax middleware

A HyraxMsg usa o chamado Network Layer do middleware Hyrax [7], que tem a arquitetura ilustrada na Figura 6. A respetiva API do Hyrax mantém a abstração de uma rede construída simultaneamente sobre várias tecnologias como Wifi, Wifi-Direct e Bluetooth (as que são usadas na HyraxMsg), tratando de forma transparente do processo de formação de rede, de roteamento de mensagens entre dispositivos, e de entrada e saída de dispositivos da rede ("churn"). Na versão atual, a formação de rede tenta construir uma "estrela" de dispositivos por cada tecnologia e a entrega de mensagens opera numa lógica "best effort".

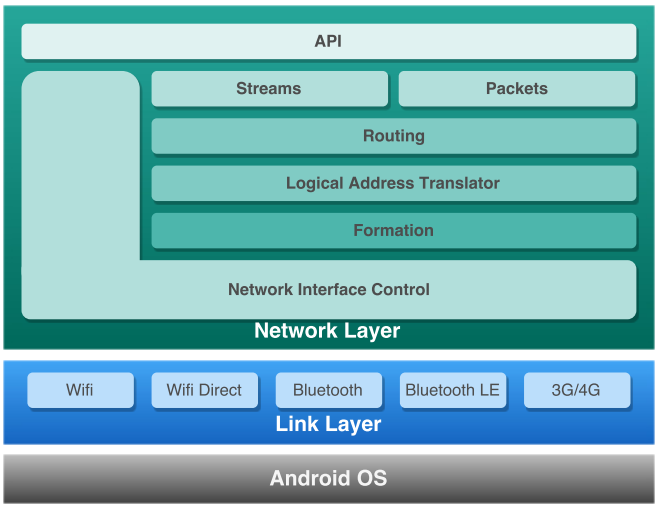


Figura 6: Hyrax Network Layer (figura retirada de [7]).

Na Figura 7 é ilustrado de forma abreviada o uso da API no contexto da classe HyraxNLCommunications. Durante a inicialização (início da listagem), é registado um "listener" para lidar com mensagens recebidas (mostrado mais abaixo na listagem), e são ativadas comunicações Wifi, Wifi-Direct e Bluetooth. É possível enviar uma mensagem para um dispositivos específico usando a operação sendTo, usada no contexto da funcionalidade de "chat" privada, ou para todos os dispositivos que estiverem alcançáveis usando a operação sendToAll, usada para disseminação da presença de um dispositivo e no contexto da funcionalidade de "chat" pública.

```
public final class HyraxNLCommunications {
  // Network instance.
  private final Network net;

  // Constructor.
  HyraxNLCommunications(Context context) {
    NetworkService.boot(context);
    net = NetworkService.getNetwork();
    net.listenMessages(nlListener);
    net.enableNetworkInterface(BLUETOOTH, ...);
    net.enableNetworkInterface(WIFI, ...);
    net.enableNetworkInterface(WIFI_P2P, ...);
  }
  // Incoming message listener.
  MessageListener nlListener
    = new MessageListener() {
      @Override
      public void onMessage(int id,
                            byte tag,
                            byte[] data,
                            Address source) {
        onIncomingData(type, data);
      }
  };
  // Send a message to a specific peer.
  void sendToPeer(Address peer, int tag, byte[] data) {
    net.send(peer, data, (byte) tag, ...);
  }
  // Send a message to all.
  void sendToAll(int tag, byte[] data) {
    net.sendToAll(data, (byte) tag, ...);
  }
}
```

Figura 7: Uso da API do Hyrax Network Layer.

3.4 Protocolo de mensagens

A aplicação usa um protocolo simples de mensagens, recorrendo à biblioteca Protocol Buffers [4]. O protocolo, ilustrado na Figura 8, é expresso na linguagem de domínio específico da Protocol Buffers, engloba duas mensagens: Chat corresponde ao envio de uma mensagem de texto, e Peer a uma mensagem de presença enviada por um dispositivo periodicamente. Ambas as mensagens têm um "timestamp" (time) associado e identificam a posição GPS do dispositivo de origem (latitude , longitude e height). O campo id em Peer identifica o nome do utilizador, e a mensagem Chat contém campos para a origem, destino e texto da mensagem (source, destination e text).

A partir da especificação do protocolo, a Protocol Buffers permite a geração de código para a manipulação das mensagens em várias linguagens, entre as quais Java. São geradas classes em correspondência às mensagens do

```
syntax = "proto3";
package org.hyrax.rescue;
...

message Chat {
  int64 time = 1;
  string source = 2;
  string destination = 3;
  string text = 4;
  double latitude = 5;
  double longitude = 6;
  double height = 7;
}

message Peer {
  int64 time = 1;
  string id = 2;
  string status = 3;
  double latitude = 4;
  double longitude = 5;
  double height = 6;
}
```

Figura 8: Protocolo de mensagens.

protocolo, incluindo a funcionalidade de conversão das mensagens para um formato binário de serialização (e vice-versa), que é usado depois na comunicação em rede. Na Figura 9 é ilustrada a construção de uma mensagem Chat e posterior conversão da mensagem num "array" de bytes.

```
  // Build message
  Chat message =
    Chat.newBuilder()
        .setTime(System.currentTimeMillis())
        .setSource(userId)
        .setDestination(destination)
        .setText(text)
        .setLatitude(currentLocation.getLatitude())
        .setLongitude(currentLocation.getLongitude())
        .setHeight(currentLocation.getAltitude())
        .build();
  ...
  // Convert to byte array and transmit it
  byte[] data = message.toByteArray();
  comm.send(data);
  ...
}
```

Figura 9: Uso de uma mensagem Chat.

4 Trabalho relacionado

Há já várias aplicações comerciais para trocas de mensagens sem ligação, tais como o FireChat [2] ou o ZombieChat [12], cujo uso se popularizou em festivais de música. Em comparação com a HyraxMsg, estas aplicações são naturalmente mais ricas em termos de funcionalidades e interação com utilizador, dado a sua natureza comercial e tempo de maturação. A conceção da HyraxMsg não teve como objetivo explorar extensivamente as várias funcionalidades oferecidas por estas aplicações, mas apenas algumas funcionalidades centrais que constituíssem uma aplicação válida como caso-de-uso para o middleware Hyrax. Vários aspetos podem no entanto ser considerados para que a aplicação se torne mais apelativa para a sua utilização no mundo real, como discutimos na Secção 5.

Subjacente a aplicações de "messaging" e outras no contexto de "mobile edge clouds", está o uso de middlewares de comunicação, tais como o Apple Multipeer-Connectity para iOS [1], o Google Nearby [3], o MeshKit pela OpenGarden [6] (que é usado pelo FireChat), ou o P2Pkit from Ueppa [11]. De forma similar ao middleware Hyrax, são empregues as várias tecnologias de comunicação em dispositivos móveis para formar "mobile edge clouds".

5 Conclusões e trabalho futuro

Apresentamos a aplicação HyraxMsg, que permite a troca de mensagens e localização entre utilizadores de dispositivos Android, recorrendo ao middleware Hyrax para comunicação numa "mobile edge cloud". A aplicação demonstra o potencial do middleware Hyrax em si, providenciando um caso de estudo relevante para futuros desenvolvimentos e experiências no âmbito do projecto Hyrax.

Como trabalho futuro, há vários aspetos que podem ser considerados para enriquecer a funcionalidade da aplicação:

- O uso de um servidor na "cloud" que permita comunicação na presença de Internet. Esta funcionalidade poderia permitir a comunicação entre utilizadores distantes no espaço.

- O servidor acima mencionado poderia adicionalmente providenciar suporte para uma plataforma de observação dos utilizadores na rede. Esta funcionalidade poderia ser especialmente relevante para aplicações civis de resposta a emergência.

- O uso mais rico da informação de localização, nomeadamente a representação da localização de utilizadores num mapa, para além da melhoria das indicações visuais atualmente implementadas sobre a proximidade de utilizadores.

- A troca de conteúdos entre utilizadores para além de mensagens de texto, por exemplo na forma de áudio, vídeo, fotografias, ou "geo-tags".

- Avaliação em cenário real, por exemplo num evento desportivo ou cultural, acompanhando futuros desenvolvimentos e experiências do middleware Hyrax.

Referências

[1] Apple MultipeerConnectivity. https://developer.apple.com/reference/multipeerconnectivity

[2] FireChat. http://opengarden.com/firechat/

[3] Google Nearby. https://developers.google.com/nearby/messages/overview

[4] Google Protocol Bufers. https://developers.google.com/protocol-buffers/

[5] Martins, U.D.R., Tan, J., Chheda, A., Sanghavi, M., Gandhi, R., Narasimhan, P.: The case for mobile edge-clouds. In: Proc. UIC/ATC'13. IEEE (2013)

[6] MeshKit SDK. https://www.opengarden.com/meshkit.html

[7] Rodrigues, J., Marques, E.R.B., Martins, R., Lopes, L., Silva, F.: Towards a middleware for mobile edge-cloud applications. In: Proc. MECC'17. ACM (2017)

[8] Rodrigues, J., Silva, J., Martins, R., Lopes, L., Drolia, U., Narasimhan, P., Silva, F.: Benchmarking Wireless Protocols for Feasibility in Supporting Crowdsourced Mobile Computing. In: Proc. DAIS'16. pp. 96–108. Springer (2016)

[9] Silva, J., Silva, D., Marques, E.R.B., Lopes, L., Silva, F.: P3-Mobile: Parallel Computing for Mobile Edge-Clouds. In: Proc. CrossCloud'17. pp. 5:1–5:7. ACM (2017)

[10] Silva, P.M.P., Rodrigues, J., Silva, J., Martins, R., Lopes, L., Silva, F.: Using Edge-Clouds to Reduce Load on Traditional WiFi Infrastructure and Improve Quality of Experience. In: Proc. ICFEC'17. IEEE Computer Society (2017), (to appear)

[11] Uepaa AG p2pkit. `http://p2pkit.io/`

[12] ZombieChat. http://getzombiechat.com