**Universidade do Minho**
Escola de Engenharia

Ricardo dos Santos Silva

# Mobilidade em Ambientes de Acesso Heterogéneos com Terminais Android

Outubro de 2010

**Universidade do Minho**
Escola de Engenharia

Ricardo dos Santos Silva

# Mobilidade em Ambientes de Acesso Heterogéneos com Terminais Android

Dissertação de Mestrado
Mestrado em Engenharia Informatica.

Trabalho efectuado sob a orientação de:
Orientador:
**Professor Paulo Martins Carvalho,**
Universidade do Minho.
Co-Orientador:
**Professor Pedro Nuno Miranda de Sousa,**
Universidade do Minho**.**
Orientador na Empresa:
**Eng. Telma Susana Pinto Mota,**
PT Inovacão.

Outubro de 2010

# Agradecimentos

Esta dissertação resultou de um longo trabalho e dedicação. Contudo, não seria possível sem a ajuda e conselhos que me foram dados durante todo o trabalho.

Em primeiro lugar, quero agradecer a disponibilidade e orientação académica dos professores Paulo Martins Carvalho e Pedro Nuno Sousa, e também agradeço aos orientadores da empresa, a Eng. Telma Mota e o Eng. Pedro Neves, pelo apoio e disponibilidade.

Agradeço também aos meus colegas Tiago Cardoso e Nelson Capela, pela ajuda e companheirismo durante a realização deste trabalho.

Um agradecimento a todos os colaboradores da PT Inovação e ao Instituto de Telecomunicações de Aveiro, pelo acolhimento.

Por último, quero agradecer à minha família e à minha namorada, Patrícia, por todo o apoio e compreensão durante a elaboração desta dissertação.

# Resumo

O paradigma de acesso à Internet está a mudar e um novo e vasto conjunto de novas tecnologias de acesso *wireless*, como por exemplo WiMAX, Wi-Fi e 3GPP UMTS/LTE, estão agora ao dispor dos utilizadores e dos seus terminais multi-acesso. Deste modo, novos mecanismos e protocolos são necessários para fornecer mobilidade transparente entre todas as tecnologias de acesso mencionadas. Actualmente, o organismo de normalização IEEE, nomeadamente o grupo 802.21, está a definir uma plataforma para optimizar os processos de mobilidade em redes de acesso heterogéneas, designada Media Independent Handover (MIH) *framework*.

Tendo como base o *framework* MIH definido pelo IEEE 802.21 e protocolos de mobilidade IP do IETF, este projecto propõe a definição de um conjunto de mecanismos e procedimentos com o objectivo de fornecer mobilidade transparente aos utilizadores em ambientes de acesso heterogéneos. Neste sentido, este trabalho focar-se-á na investigação e implementação da norma IEEE 802.21 em terminais móveis com o sistema operativo Android, com o intuito de assegurar a mobilidade entre redes de acesso Wi-Fi e 3G (UMTS), com níveis de qualidade e desempenho adequados.

# Abstract

The Internet access is changing and a new and broad set of new wireless access technologies, such as WiMAX, Wi-Fi e 3GPP UMTS/LTE, are now available to users and their multi-access terminals. Thus, new mechanisms and protocols are needed to provide seamless mobility across the access technologies mentioned. Currently, the IEEE standards organization, including the 802.21 group, is setting a platform to optimize the processes of mobility in heterogeneous access networks, called Media Independent Handover (MIH) framework.

Based on the MIH framework defined by the IEEE 802.21 and IP mobility protocols of the IETF, this project proposes the definition of a set of mechanisms and procedures in order to provide seamless mobility for users in heterogeneous access environments. In this context, this work will focus on research and implementation of IEEE 802.21 mobile terminals with the operating system Android, in order to ensure mobility between Wi-Fi and 3G (UMTS) networks, with adequate levels of quality and performance.

# Conteúdo

# Lista de Figuras

# Lista de Tabelas

# Lista de Acrónimos

| | |
|---|---|
| 3GPP | 3rd Generation Partnership Project |
| ABC | Always Best Connected |
| ADB | Android Debug Bridge |
| AIDL | Android Interface Definition Language |
| AMPS | American Mobile Phone System |
| AP | Access Point |
| ARM | Advanced RISC Machine |
| ASN1 | Abstract Syntax Notation One |
| BSSID | Basic Service Set Identifier |
| BWA | Broadband Wireless Access |
| CDMA | Code-Division Multiple Access |
| CoA | Care-of-Address |
| DAD | Duplicate Address Detection |
| DEX | Dalvik Executable |
| DSL | Digital Subscriber Line |
| EDGE | Enhanced Data rates for GSM Evolution |
| EPC | Evolved Packet Core |
| ESS | Extended Service Set |
| E-UTRAN | Evolved Universal Terrestrial Radio Access Network |
| FA | Foreign Agent |
| FDD | Frequency Division Duplex |

| | |
|---|---|
| FMIP | Fast Mobile IP |
| FN | Foreign Network |
| GPRS | General Packet Radio Service |
| GPS | Global Positioning System |
| GSM | Global System for Mobile communications |
| HA | Home Agent |
| HN | Home Network |
| HSPA | High Speed Packet Access |
| HSPA+ | Evolved HSPA |
| HSDPA | High Speed Downlink Packet Access |
| HSUPA | High Speed Uplink Packet Access |
| IEEE | Institute of Electrical and Electronics Engineers |
| IETF | Internet Engineering Task Force |
| IMS | IP Multimedia Subsystem |
| IP | Internet Protocol |
| IS | Information Server |
| ITU | International Telecommunication Union |
| JNI | Java Native Interface |
| L3MP | Layer 3 Management Protocol |
| LTE | Long Term Evolution |
| MAC | Media Access Control |
| MAN | Metropolitan Area Network |
| Mbps | Mega bit per second |
| MIH | Media Independent Handover |
| MICS | Media Independent Command Service |
| MIES | Media Independent Event Service |
| MIHF | Media Independent Handover Function |
| MIHU | Media Independent HandoverUser |
| MIIS | Media Independent Information Service |

| | |
|---|---|
| MIMO | Multiple-input multiple-output |
| MIP | Mobile IP |
| MMS | Multimedia Message Service |
| MN | Mobile Node |
| NDK | Native Development Kit |
| NGN | Next Generation Network |
| NMT | Nordic Mobile Telephone |
| OFDM | Orthogonal frequency-division multiplexing |
| OHA | Open Handset Alliance |
| PDC | Personal Digital Cellular |
| PMIP | Proxy Mobile IP |
| PoS | Point of Service |
| QoE | Quality of Experience |
| QoS | Quality of Service |
| RAT | Radio Access Technology |
| RIL | Radio Layer Interface |
| RILD | Radio Layer Interface Daemon |
| RISC | Reduced Instruction Set Computer |
| RNC | Radio Network Controller |
| RNS | Radio Network Subsystem |
| RSSI | Received Signal Strength Indication |
| SAE | System Architecture Evolution |
| SAP | Service Access Point |
| SDK | Software Development Kit |
| SIP | Session Initiation Protocol |
| SMS | Short Message Service |
| SSID | Service Set Identifier |
| SSL | Secure Socket Layer |
| TDD | Time Division Duplex |

| | |
|---|---|
| TDMA | Time Division Multiple Access |
| UMA | Unlicensed Mobile Access |
| USIM | UMTS Service Identity Module |
| UTRAN | Universal Terrestrial Access Network |
| VHO | Vertical Handovers |
| VoD | Video on Demand |
| VoIP | Voice over IP |
| UMTS | Universal Mobile Telecommunications System |
| WAN | Wide Area Network |
| W-CDMA | Wideband Code Division Multiple Access |
| Wi-Fi | Wireless Fidelity |
| WiMAX | Worldwide interoperability for Microwave Access |
| XML | Extensible Markup Language |

# Capítulo 1

# Introdução

## 1.1 Enquadramento e Motivação

De acordo com o ITU, em 2010 o número de utilizadores das redes móveis irá atingir os 5 biliões, após em 2009 terem atingido os 4.6 biliões [1]. No mesmo sentido, o número de utilizadores da Internet cresceu nos últimos anos, sendo neste momento 2 biliões de utilizadores em todo o mundo [2] e estudos recentes revelam que no ano 2014 o número de utilizadores móveis que acedem à Internet irá ultrapassar o número de utilizadores que acedem a partir de um computador pessoal [3]. Além disso, os utilizadores exigem uma conectividade permanente à Internet, independentemente da tecnologia de acesso e do local de acesso, com débitos elevados e com garantias de mobilidade sem quebra de serviços, usufruindo deste modo do conceito ABC[1] [4].

Tanto as redes *wireless* de banda larga, como por exemplo 3G, WiMAX e Wi-Fi, e terminais com múltiplas interfaces de rede, têm vindo a massificar-se e, como consequência, novos estudos têm sido desenvolvidos para permitir a continuidade das sessões dos utilizadores durante os processos de mobilidade (*handover*) entre estas redes - mobilidade transparente. Neste sentido, têm sido colocados novos desafios aos operadores para que consigam integrar nas suas redes de próxima geração a capacidade de garantirem essa mobilidade, para um vasto conjunto de tecnologias de acesso. Esta mobilidade vertical transparente representa uma vantagem tanto para o operador, como para o utilizador. O operador pode gerir melhor a sua rede de acordo com a carga e número de utilizadores; o utilizador pode usufruir de uma melhor qualidade de serviço/experiência.

Perante este cenário, o grupo de normalização IEEE 802.21 definiu uma nova plataforma, designada *Media Independent Handover framework* [5], cujo objectivo é optimizar e uniformizar os mecanismos de mobilidade entre diferentes tecnologias de acesso sem interrupção

[1] *Always Best Connected*

do serviço. Uma parte significativa das funcionalidades necessárias para fornecer mobilidade transparente está relacionada com as interacções que é necessário estabelecer entre as especificidades das tecnologias de acesso e as camadas protocolares superiores, nomeadamente o "mundo IP". É precisamente neste universo que o grupo IEEE 802.21 pretende contribuir com a definição da *framework* MIH, fornecendo uma camada de abstracção, independente das tecnologias de acesso [6]. Assim, surge finalmente um protocolo que permite a interacção entre o 'mundo IP' com as tecnologias de acesso sem ter que se preocupar com as particularidades de cada uma delas. Através da disponibilização de um conjunto de serviços, na forma de comandos e eventos, o IEEE 802.21 fornece informação estática e dinâmica sobre o estado actual do link *wireless* da tecnologia de acesso aos protocolos de gestão de mobilidade IP, auxiliando estes protocolos na tomada de decisões sobre o processo de *handover*.

Com grande velocidade, o Android têm vindo a ganhar mercado numa variedade de dispositivos móveis, desde *smartphones* até *netbooks* Além disto, permite um elevado grau de flexibilidade e personalização do terminal, incluindo alterações ao próprio sistema operativo (*open-source*) com o acréscimo de novas funcionalidades. A abertura inerente ao tipo de sistema operativo *open-source*, permite que possa ser usado para a criação de novos tipos de serviços, não só ao nível aplicacional, mas internamente ao Android. Por outro lado, a maioria dos terminais Android possui interfaces de rede Wi-Fi e 3G (voz e dados), sendo uma vantagem integrar mecanismos de mobilidade heterogénea.

Devido à tendência dos operadores para oferecer mobilidade entre diferentes redes de acesso e ao grande sucesso da plataforma Android, este trabalho dará um valioso contributo ao oferecer uma plataforma móvel com suporte à mobilidade heterogénea.

## 1.2 Objectivos

Este trabalho tem como principal objectivo a implementação de uma solução/mecanismo de mobilidade heterogénea no Android, ao integrar o protocolo IEEE 802.21 e um mecanismo de mobilidade IP. O trabalho deverá começar com um estudo do protocolo e uma análise dos mecanismos de mobilidade IP disponíveis. Depois da análise dos requisitos, será necessário analisar o sistema Android, a sua arquitectura e características, de forma a implementar uma aplicação que irá integrar as entidades responsáveis pela gestão de mobilidade.

Desta forma, são enumerados os seguintes objectivos para este trabalho:

- Estudo do conceito de mobilidade heterogénea entre diferentes redes de acesso e o estudo da norma IEEE 802.21.
- Analisar várias abordagens/soluções para implementar a mobilidade vertical.

- Explorar a plataforma Android, obter conhecimento sobre a sua arquitectura e funcionamento.

- Estudar várias formas de integrar a norma IEEE 802.21 e mecanismos de mobilidade IP na plataforma Android.

- Especificação, implementação e teste da solução que integra o protocolo IEEE 802.21 e um mecanismo de mobilidade IP num terminal móvel (Android), com suporte para as interfaces Wi-Fi e 3G.

## 1.3 Sumário das principais contribuições

A principal contribuição resultante do trabalho desenvolvido irá ser uma plataforma Android que permitirá a mobilidade entre diferentes redes de acesso, especificamente, entre Wi-Fi e 3G, e desenhada de forma a ser possível expandir o suporte para outras tecnologias de rede. A tendência é para que o acesso à Internet seja feito principalmente através de dispositivos móveis, como os terminais Android, e que devido à grande variedade de tecnologias de acesso *wireless*, a implementação de um sistema de mobilidade heterogénea é uma vantagem. Como fruto deste trabalho, foi publicado o artigo [7], apresentado na conferência MobiMedia 2010 em Lisboa [8]. Uma nova versão do artigo, com novos resultados publicados neste trabalho, está a ser preparado e irá ser submetido na revista *Mobile Networks and Applications* [9].

## 1.4 Organização da dissertação

Este documento está organizado em 5 capítulos. A descrição de cada um dos capítulos é a seguinte:

- Introdução: este capítulo fez o enquadramento e contextualização do trabalho, ao apresentar o tema geral e objectivos do trabalho a desenvolver.

- Estado de Arte: aborda a base teórica que irá permitir o desenvolvimento da solução de mobilidade heterogénea no Android. São apresentadas as diferentes tecnologias de acesso *wireless*, os mecanismos de mobilidade IP, o protocolo IEEE 802.21 e o sistema Android.

- Desenho e Implementação: aborda todo o trabalho desenvolvido, desde a arquitectura de mobilidade onde o Android se insere até o funcionamento interno dos componentes no Android, inclusive modificações feitas ao sistema.

- Testes e Resultados: demonstra o ambiente de testes que foi usado, a metodologia de testes usada e a avaliação dos resultados obtidos. O objectivo é validar a solução e demonstrar o desempenho conseguido.

- Conclusões: neste último capítulo são apresentadas as principais conclusões obtidas de todo o trabalho. É feita uma análise de todos os capítulos e a principal contribuição desta dissertação. Por último, é feita uma análise de futuros desenvolvimentos para melhorar a solução.

# Capítulo 2

# Estado de Arte

Neste capitulo iremos abordar as diferentes tecnologias de acesso *wireless*[1] que irão ser usadas neste trabalho. Posteriormente, os mecanismos de mobilidade ao nível IP e o protocolo IEEE 802.21 que visa optimizar a mobilidade entre diferentes tecnologias de acesso *wireless*. Por último, irá ser abordado o sistema Android, a sua arquitectura, funcionamento e estrutura do código fonte.

## 2.1 Tecnologias de Acesso Wireless

As tecnologias *wireless*[2] têm vindo a massificar-se nos últimos anos devido as exigências dos utilizadores para terem acesso aos vários serviços independentemente da sua localização. Por outro lado, devido às exigências dos serviços multimédia disponibilizados pelos operadores, como Vídeo-chamada, Mobile TV[3] e vídeo-conferência, tem havido uma evolução das redes para suportar maiores larguras de banda e melhor qualidade de serviço.

Esta secção irá focar-se nas tecnologias com capacidade para disponibilizar serviços multimédia, como Wi-Fi, 3G (UMTS e HSPA) e WiMAX, abordando a sua arquitectura, mecanismos de QoS e mobilidade. As tecnologias de acesso têm vindo constantemente a evoluir, sendo esta evolução liderada pelas duas maiores entidades de normalização no sector das telecomunicações: o IEEE (*Institute of Electrical and Electronics Engineers* e o 3GPP (*3rd Generation Partnership Project*). O 3GPP é responsável pela rede celular mais usada ao nível mundial, a rede GSM, e o IEEE têm um grande historial a normalização de tecnologias, sendo a mais conhecida as redes IEEE 802.21 ou redes Wi-Fi.

[1] Em inglês, Broadband Wireless Technologies
[2] Sem fios
[3] Televisão no telemóvel

### 2.1.1 UMTS

#### Historia das Redes Celulares

A primeira geração (1G) de redes celulares (redes móveis) apareceu no meio da década dos 80, com o NMT e o AMPS. Estas redes ofereciam os serviços básicos, principalmente focados para serviços de voz. Devido à falta de normalização, os sistemas eram incompatíveis. Mais tarde e devido ao aumento do uso das redes móveis, foi iniciada a especificação da segunda geração por instituições de normalização internacionais. Um dos principais requisitos para esta nova especificação era a compatibilidade entre os diferentes sistemas dentro de uma região (por exemplo, dentro da União Europeia), permitindo aos utilizadores acesso em qualquer ponto dentro desta.

Como é da natureza humana, nem todas as regiões adoptaram uma norma única. Surgiram várias como a cdmaOne, TDMA (IS-136) e PDC, mas a que teve um maior sucesso foi a Global System for Mobile Communications (GSM), originalmente chamada Groupe Spécial Mobile. Esta foi criada na Europa e implementada a nível mundial, estando presente em quase 200 redes em mais de 100 países ao nível mundial. Esta tecnologia sofreu algumas evoluções como o GPRS e o EDGE que permitiram maiores larguras de banda na comutação de pacotes dentro da rede GSM.

A terceira geração (3G) oferece além dos serviços da primeira e segunda geração, serviços multimédia e separa os serviços da infra-estrutura de rede. Por outro lado, integra as comunicações tradicionais (comutação de circuitos) com o protocolo IP, oferece maiores taxas de transmissão, melhor segurança e utilização eficiente do espectro. Entre as principais tecnologias 3G encontra-se a rede UMTS e cdma2000, onde a primeira baseia-se na rede GSM e segunda na cdmaOne. Devido ao grande número de utilizadores da rede GSM, espera-se que a rede UMTS venha a atingir um número de utilizadores na mesma grandeza, pois a rede UMTS funciona em conjunto com a rede GSM, permitindo garantir a interoperabilidade com esta tecnologia e reduzir os custos de implementação por parte dos operadores. Isto permite aos operadores implementar a rede UMTS sobre a rede GSM e manter o serviço disponível aos terminais que só suportam a rede GSM. Esta estratégia garante ao 3GPP manter o controlo do mercado de redes móveis já alcançado com a rede GSM.

#### Arquitectura

A rede UMTS (3GPP Release 99), apresentada como a evolução da rede GSM e GPRS, herdou vários elementos da arquitectura e princípios de funcionamento da rede GSM, onde o maior desenvolvimento encontra-se na tecnologia de acesso rádio usada, WCDMA com FDD[4]

[4]Divisão dos canais de envio e retorno por diferentes frequências

ou TDD[5]. O WCDMA usa as mesmas técnicas que o CDMA, mas usa diferentes canais de controlo e sinalização, maior largura de banda e novas funcionalidades. A rede UMTS permite transmissão de voz e dados. A largura de banda suportada para dados é de 384 kbps.

A arquitectura da rede UMTS está divida em 3 partes: o EU (User Equipment), o UTRAN (Universal Terrestrial Radio Access Network) e o CN (Core Network). A rede UMTS define também as interfaces usadas para comunicação entre os diferentes componentes. O EU é o dispositivo/terminal móvel usado pelo utilizador, que actualmente para além de suportar UMTS, suportam GSM, Bluetooth e Wi-Fi. O equipamento do utilizador está dividido em duas partes: equipamento móvel e o USIM[6]. O USIM é usado para guardar informação do utilizador, informação de autenticação, algoritmos de cifra usados e espaço extra para outras informações (por ex. número de telefone).

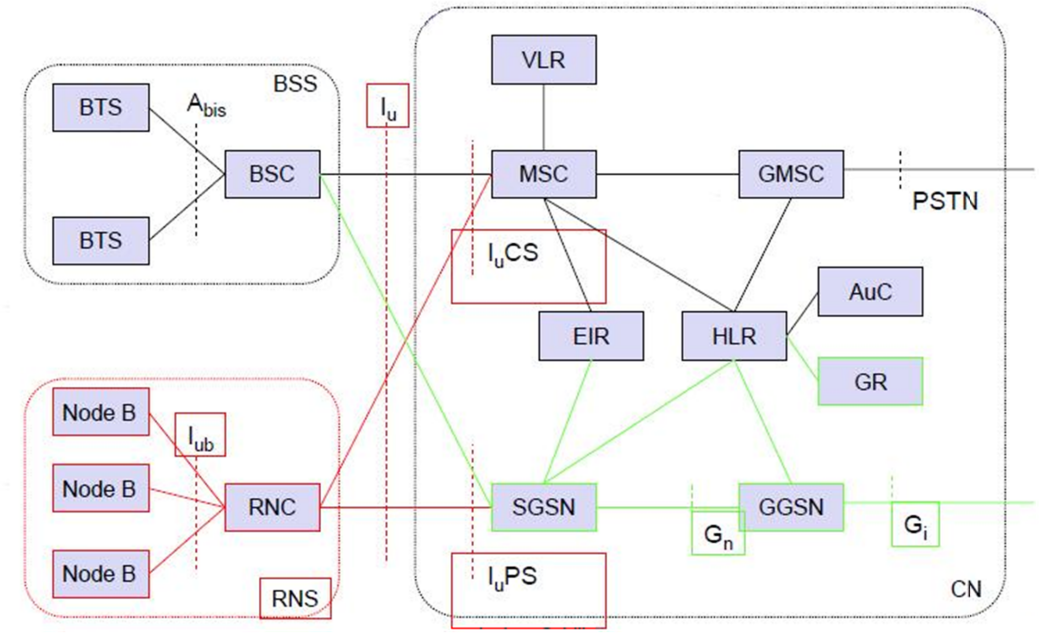


Figura 2.1: Arquitectura da rede UMTS [10]

A Figura 2.1 mostra a arquitectura da rede UMTS, onde os componentes com contornos pretos são componentes já existentes na rede GSM, os componentes com contornos verdes são componentes definidos na rede GPRS e os componentes com contornos vermelhos são componentes novos na rede UMTS.

A arquitectura do CN é baseada na arquitectura da rede GSM e GPRS, fornece os serviços baseados em comutação por circuitos e comutação por pacotes. Tal com na rede GSM, o CN

[5]Divisão dos canais de envio e retorno por intervalos de tempo
[6]Módulo de identificação

gere a localização do EU e fornece mecanismos para suportar processos de *handover*. De forma a suportar UMTS, alguns dos componentes do GSM e GPRS são actualizados. O CN está dividido em duas partes: a rede de comutação de circuitos que dá suporte aos serviços da tecnologia GSM e a rede de comutação de pacotes que dá suporte à tecnologia All-IP. Os componentes que constituem o CN, como explicados na Figura 2.1, são explicados a seguir:

- *Mobile Switching Center* (MSC): efectua a comutação de circuitos e cria conexões entre BSCs e outras MSCs. Efectua o registo, autenticação dos utilizadores, gere o processo de *handover* entre diferentes MSCs, actualiza a localização do terminal e reencaminha conexões para outras redes.
- *Gateway Mobile Switching Center* (GMSC): controla as conexões com outras redes de comutação por circuitos (e.g. rede fixa).
- *Serving GPRS Support Node* (SGSN): gere a comunicação por comutação de pacotes na rede UMTS. Comunica com o GGSN para enviar e receber pacotes de redes externas (e.g. Internet).
- *Gateway GPRS Support Node* (GGSN): efectua a ligação entre a rede de comutação de pacotes interna com as redes externas.
- *Home Location Register* (HLR): base de dados que armazena informações dos utilizadores da rede.
- *Visitor Location Register* (VLR): base de dados que armazena informações dos utilizadores ligados a um determinado MSC, a qual o VLR pertence.
- *Equipment Identity Register* (EIR): suporta serviços de segurança dos terminais. Inclui listas com os equipamentos autorizados, em observação e bloqueados.
- *Authentication Center* (AuC): serviço responsável pela autenticação dos utilizadores. Inclui os algoritmos de cifra e chaves dos utilizadores.

A rede UTRAN é a rede de acesso que permite a comunicação entre os equipamentos e a rede core (ver Figura 2.2). Esta rede permite que um terminal móvel possa ligar-se a múltiplas células (Node B) em simultâneo de modo a suportar um *soft-handover* (*handover* sem quebra de serviço).

O UTRAN está dividido em vários RNSs, que estão formados por dois elementos: o Node B é a antena rádio do UMTS, que pode comparar-se à *Base Station* do GSM; e o RNC é elemento que controla vários Node B, o qual gere os recursos de rádio, controlo de admissão e controlo de *handover*. Os RNS estão conectados com todos os outros RNS da rede UMTS por uma interface interna da rede de acesso, que é usada para gestão de recursos e gestão da mobilidade [11].

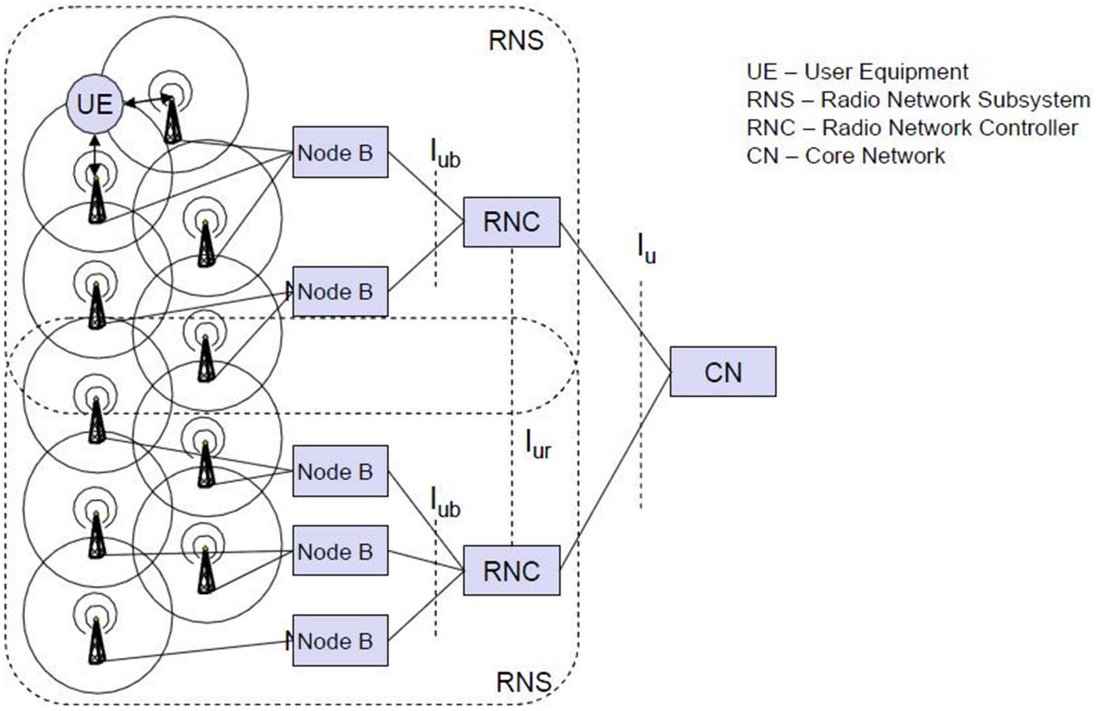


Figura 2.2: Arquitectura do UTRAN [10]

### Handovers

A rede UMTS permite *handovers* entre micro-células, células normais e diferentes RNC, além de permitir também o roaming do utilizador entre diferentes operadores. Permite também o handover entre tecnologias 3GPP diferentes, como o GSM. Os tipos de *handovers* permitidos são *hard-handover*, *soft-handover* e *softer-handover*. No *hard-handover*, o terminal desliga-se de um Node B para poder ligar-se a outro. No *soft-handover*, o terminal liga-se a um novo Node B antes de desligar-se no anterior. O *softer-handover* é derivado do *soft-handover*, na qual o terminal pode ter várias ligações com o mesmo Node B.

Em relação a *handovers* entre redes de outras tecnologias, o consórcio 3GPP definiu duas abordagens para redes Wi-Fi: MIP/GTP (GPRS Tunnelling Protocol) e sinalização inter-SGSN (GTP-C).

### QoS

O QoS na rede UMTS é controlado e gerido por sinalização e mecanismos de escalonamento que efectuam a diferenciação do tráfego por fluxo. Também inclui algoritmos de controlo de acesso e pode efectuar reserva de recursos. Estas funcionalidades permitem garantir o QoS

entre a origem e o destino do serviço. A rede UMTS define quatro classes de tráfego diferentes, sendo a principal diferença entre elas a sensibilidade ao atraso dos dados que transportam [12]. As quatro classes são as seguintes:

- Conversação: usada em tráfego de aplicações de voz e tem como objectivo um baixo atraso e pouca variação de atraso entre pacotes (*jitter*).
- *Streaming*: usada para *streaming* de vídeo e tem como objectivo também um baixo jitter.
- Interactiva: usada em aplicações de acesso à Web (*browser*).
- Background: outro tipo de tráfego que não tenha requisitos ao nível do atraso (e.g. email).

### 2.1.2 HSPA

As larguras de banda oferecidas pelo UMTS são baixas para as necessidades actuais dos utilizadores, por isso foi introduzido um novo mecanismo que permite uma maior largura de banda, chamado HSPA. O HSPA é a terminologia usada quando uma rede implementa o HSDPA (3GPP Release 5) e HSUPA (3GPP Release 6). Esta tecnologia é largamente usada ao nível mundial para acesso móvel à Internet. Estas tecnologias só exigem um simples *upgrade* na rede dos operadores móveis, sendo fácil migrar para esta tecnologia.

#### HSDPA

Este mecanismo implica uma extensão a interface rádio do UMTS, permitindo obter uma maior largura de banda no *down-link*, reduzir a latência e incrementar a capacidade [13]. A largura de banda oferecida pode ir até os 14.4 Mbps. Esta tecnologia introduz novas técnicas como modulação adaptativa, controlo da largura da banda, partilha de canais e recuperação rápida de erros (H-ARQ – Hybrid Automatic Repeat Request). A modulação adaptativa permite ajustar a técnica de modulação de acordo com a distância do equipamento ao Node B.

Por outro lado, existe uma utilização mais eficiente do espectro, o que beneficia assim os operadores móveis, pois podem aumentar o número de utilizadores de uma dada área sem necessitarem de grandes investimentos.

#### HSUPA

O HSUPA, também designado com Enhanced Uplink, baseia-se principalmente nas tecnologias introduzidas pelo HSDPA. Esta nova tecnologia permite melhorar o desempenho no uplink, oferecendo velocidades até 5.76 Mbps. Entre as melhorias feitas encontra-se a recuperação rápida de erros (H-ARQ), prioridade aos terminais com melhores condições de sinal (por exemplo, mais perto do Node B) e canal dedicado para comunicações em uplink [13].

### 2.1.3 Wi-Fi

O grupo IEEE 802.11 produz normas para redes WLAN[7] [14], sendo estas normas também conhecidas como redes Wi-Fi [15]. A primeira norma publicada por este grupo foi a norma IEEE 802.11 no ano 1997. Esta permite 1 ou 2 Mbps e opera na frequência dos 2.4 GHz. No ano 1999 saiu a norma IEEE 802.11a que permite 54 Mbps a operar na frequência 5 GHz. A seguir foi publicada a norma IEEE 802.11b com 11 Mbps e a norma IEEE 802.11g com 54 Mbps, ambas na frequência 2.4 GHz. A última norma desenvolvida por este grupo é a norma IEEE 802.11n, que permite larguras de banda até os 300 Mbps com o uso do mecanismo MIMO, o qual baseia-se no uso de múltiplas antenas no transmissor e no receptor.

#### Arquitectura

A arquitectura da rede IEEE 802.11 suporta dois modos de funcionamento, modo sem infra-estrutura na qual várias estações[8] podem-se ligar directamente[9]. E o modo com infra-estrutura que faz uso de um ponto de acesso[10] que possui as antenas rádio. Neste modo toda a comunicação entre as estações ligadas passa pelo AP.

Os vários componentes da arquitectura são apresentados na Figura 2.3 e explicados a seguir:

- *Basic Service Set* (BSS): é o elemento base de uma rede IEEE 802.11 e é formado pelas várias estações ligadas a um AP (inclusive). Numa rede *ad-hoc* é chamado como *Independent BSS*.

- *Basic Service Area* (BSA): é a área de cobertura de uma BSS, onde as estações podem comunicar com o AP. Esta cobertura vai de 50 metros em ambientes interiores até 450 metros em ambientes externos.

[7] Rede local sem fios
[8] Dispositivo móvel (computador, PDA, etc.)
[9] Normalmente chamado modo *ad-hoc*
[10] Em inglês, *Access Point* (AP)

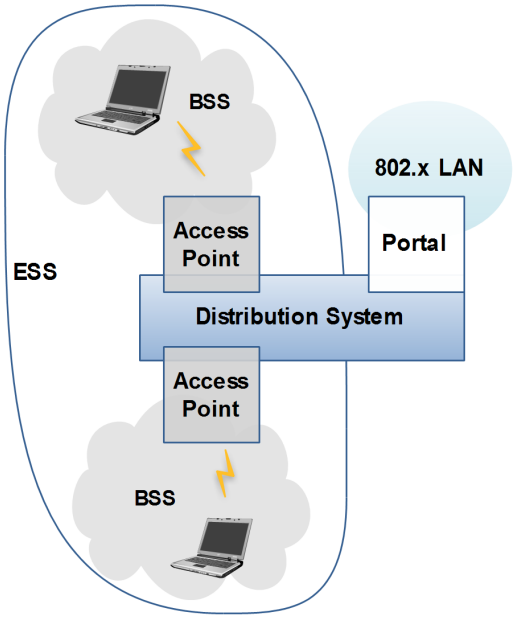


Figura 2.3: Architectura da norma IEEE 802.11

- *Distribution System* (DS): é o componente que permite ligar várias BSSs. Em vez de ser cada BSS independente, várias BSSs podem estar ligadas para formar uma só rede. O DS permite a mobilidade das estações entre os vários BSSs sem quebra de serviço.
- *Extended Service Set* (ESS): é o conjunto de várias BSSs unidas por um DS. O ESS não inclui o DS.

A norma define 3 serviços usados para ligar-se à rede:

- *Association*: estabelecer uma associação inicial entre a estação e o AP.
- *Reassociation*: transferir a associação já existente de um AP para outro.
- *Disassociation*: terminar uma associação existente.

### Handovers

O IEEE 802.11 permite *handovers* entre dois APs diferentes. Este processo está dividido em duas fases: a descoberta e a re-associação/re-autenticação. Na primeira fase, que ocorre ao detectar movimento, o terminal obtém os APs à volta que sejam potenciais APs destino. Ao escolher o novo AP, inicia-se a fase de re-associação e re-autenticação. Esta mobilidade é

só possível dentro de uma mesma DS. Para mobilidade entre diferentes DS, é necessário um gestor de mobilidade o nível IP.

#### QoS

Um dos principais problemas das redes *wireless* é a largura de banda limitada e as altas taxas de erros nos pacotes. Um dos maiores problemas para as aplicações que correm em ambientes *wireless* é que em Wi-Fi não é possível alocar um QoS necessário para uma dada aplicação. Este problema é crítico para aplicações como VoIP. A norma IEEE 802.11e [16] define um conjunto de mecanismos para atribuir prioridades ao tráfego com base nos requisitos QoS. Assim, o acesso ao meio *wireless* é baseado nas prioridades do tráfego.

### 2.1.4 WiMAX

Esta tecnologia foi criada com o objectivo de fornecer uma rede de acesso wireless em locais onde é impossível ou representa um grande custo fornecer acesso à Internet por DSL, fibra ou cabo coaxial. Baseia-se na família de normas IEEE 802.16, chamadas redes Wireless MAN. No entanto, WiMAX é o nome comercial criado pelo consorcio WiMAX Forum [17], o qual promove esta tecnologia e interoperabilidade entre os vários fabricantes. Uma das suas grandes vantagens em relação a outras tecnologias é que a sua implementação é simples e de baixo custo, tal como as redes Wi-Fi, sendo que o WiMAX tem uma área de cobertura superior. Por outro lado, possui mecanismos para garantias QoS e segurança.

Existem várias normas na família IEEE 802.16, no entanto, as duas normas principais são IEEE 802.16d e IEEE 802.16e, também chamadas Fixed WiMAX e Mobile WiMAX, respectivamente. A norma IEEE 802.16d [18], também designada como IEEE 802.16-2004, cumpre o objectivo de fornecer um acesso banda larga, equivalente à tecnologia DSL mas sem fios (*wireless*). Opera nas frequências abaixo dos 11 GHz, mais especificamente 2.5, 3.5 e 5.7 GHz, e permite uma área de cobertura até os 50 km [19].

A norma IEEE 802.16e [20] veio permitir mobilidade e QoS nas redes WiMAX. O QoS é suportado alocando uma conexão para cada terminal para uma determinada classe de QoS. Para esta conexão é possível definir parâmetros específicos, como por exemplo, a largura de banda e a latência. Esta norma surgiu como resposta as tecnologias 3GPP europeias, de modo a criar uma tecnologia concorrente à bem sucedida GSM e os seus sucessores UMTS e LTE.

## 2.2 Mobilidade

A mobilidade é a capacidade de um terminal poder mover-se de um ponto de acesso e é normalmente associada à mobilidade contínua, capacidade do terminal mover-se para outro ponto de acesso sem ocorrer quebra de serviço, quer seja comunicação por voz, dados ou outro tipo qualquer de serviço. Existe outro conceito chamado portabilidade, que é quando o terminal quebra o serviço ao mover-se para outro ponto de acesso. Este conceito é muitas vezes considerado como mobilidade discreta.

Nos últimos anos a computação móvel tem vindo a crescer e os utilizadores exigem uma ligação constante a rede, e por consequência, suporte à mobilidade. Por outro lado, as redes de próxima geração irão oferecer um ambiente heterogéneo e a mobilidade será uma importante vantagem. O protocolo IP requer que um terminal ligado à rede seja identificado por um único endereço IP, logo quando o terminal move-se para uma nova rede, é necessário configurar de novo o endereço. Para resolver estes problemas, foram criados mecanismos de mobilidade ao nível IP que permitem manter o mesmo endereço IP entre diferentes rede e alguns inclusive permitem fazê-lo sem quebra de serviço.

Ao nível lógico várias tecnologias já oferecem mobilidade entre diferentes pontos de acesso, como o IEEE 802.11 (Wi-Fi) e IEEE 802.16 (WiMAX), o qual permite manter o mesmo IP entre os diferentes pontos de acesso. Por outro lado, as redes 3G como a UMTS, são mais avançadas nos mecanismos de mobilidade e permitem mudar entre diferentes células sem quebra de serviço. No entanto, este tipo de mobilidade ao nível lógico não permite efectuar *handovers* entre diferentes redes IP.

### 2.2.1 Conceitos genéricos

#### Localização/Identificação

A localização/identificação tem como objectivo permitir ao terminal manter uma identificação permanente independentemente da sua localização, permitindo ao terminal continuar aceder aos mesmos serviços e ser contactado pela mesma identificação [21]. Para poder suportar este conceito, o sistema de mobilidade deve oferecer mecanismos para descobrir a localização actual do terminal e mecanismos que permitam ao terminal actualizar a sua localização actual. Existem vários tipos de mecanismos para o terminal actualizar a sua localização:

- Baseado no tempo: actualizações periódicas da localização (normalmente usado como backup para outros mecanismos).
- Baseado no movimento: actualização após o terminal ter atravessado um determinado número de áreas (por ex. grupo pontos de acesso ou células).

- Baseado na distância: a localização é actualizada sempre que o terminal tenha percorrido uma determinada distância (por ex. distância física, pontos de acesso, etc.).
- Baseado em parâmetros: a localização é actualizada após um determinado parâmetro ter-se alterado.
- Actualização implícita: o terminal não actualiza a sua localização, mas a rede consegue saber a localização do terminal quando recebe dados ou outra informação de sinalização.
- Actualização probabilística: as operações de actualização são efectuadas de acordo com a distribuição de probabilidade de uma determinada variável.

Por outro lado, existem mecanismos para a rede conseguir obter a localização actual do terminal, chamados mecanismos de *paging*. Neste mecanismo a rede envia uma *paging message*[11] para uma determinada área (podendo ser estática ou dinâmica) para informar ao terminal da existência de dados. A seguir o terminal responde com uma actualização com a informação da sua localização actual. A *paging message* pode ser enviada para todas as áreas em *broadcast*, sequencialmente para cada área ou enviar para um grupo de terminais ou área geográfica.

### Entrega de dados

A entrega de dados à localização actual do terminal pode ser feita através de uma das seguintes alternativas [21]:

- Entrega Directa: o emissor obtém a localização actual do terminal receptor e envia-lhe os dados directamente.
- Entrega retransmitida: o emissor não precisa de saber a localização actual do receptor. Existe uma entidade (*relay*) que intercepta os dados do emissor e os envia para a localização actual do receptor.
- Entrega integrada: esta alternativa junta os dois mecanismos anteriores. No inicio, os dados são interceptados pelo *relay*. Durante o processo, o *relay* ou terminal receptor enviam a localização actual do receptor e o emissor começa a enviar os dados directamente.

### Handover

*Handover* é o transição de um ponto de acesso para outro. Pode ser iniciada pelo terminal móvel ou gestor de mobilidade na rede, por exemplo, devido a uma degradação da qualidade

[11] Mensagem de *paging*

do sinal. O *handover* deve ser executado de forma rápida para haver o mínimo de perdas e ser transparente para o utilizador. Existem dois tipos de *handover*: horizontal e vertical. *Handovers* horizontais é quando o terminal muda de um ponto de acesso para outro da mesma tecnologia, por exemplo, entre pontos de acesso Wi-Fi dentro da mesma DS. No caso de *handovers* verticais é quando o terminal muda de um ponto de acesso para outro de diferente tecnologia (*handover* heterogéneo), por exemplo, entre uma célula UMTS e um ponto de acesso Wi-Fi.

O processo de *handover* pode ser executado de duas formas: *hard-handover* e *soft-handover* (também chamados de *break-before-make* e *make-before-break*, respectivamente). No *hard-handover* a conexão anterior é desligada antes de efectuar uma nova conexão, isto é, nunca estão duas conexões em simultâneo. Esta abordagem pode poupar recursos do sistema mas é a quebra de serviço pode ser crítica para o serviço. Por outro lado, no *soft-handover* é mantida as duas conexões em simultâneo, sendo a antiga conexão desligada quando a nova conexão é estabelecida. Permite que o *handover* ocorra sem quebra de serviço mas em contrapartida são usados mais recursos. Um *handover* onde não há quebra de serviço é um *seamless handover*. Para evitar uma eventual perda de pacotes por degradação de sinal, no *soft-handover* podem ser enviados replicas dos dados pelas duas conexões no processo de *handover*.

O *handover* pode ser iniciado por diferentes razões. *Handovers* imperativos são iniciados devido a uma degradação do serviço, por exemplo, degradação do sinal. Por outro lado, *handovers* alternativos são iniciados com o objectivo de fornecer uma melhor qualidade de serviço ou para satisfazer as preferências do utilizador. No caso de ser a rede, como a decisão é feita pela rede, o processamento é centralizado nela. No caso do terminal ter controlo do *handover*, o processamento é distribuído por todos os terminais mas existe uma maior complexidade. Em qualquer uma das abordagens, uma entidade pode assistir a outra. Por exemplo, quando a rede tem controlo, o terminal pode ajudar a rede na decisão ao fornecer a potência de sinal recebida dos pontos de acesso vizinhos.

### 2.2.2 Mobilidade IP

Nesta secção irão ser abordados diferentes mecanismos de mobilidade ao nível IP, que normalmente são chamados de *Layer 3 Mobility Protocols* (L3MP).

#### Mobile IPv4

O MIPv4 (RFC3344) [22] é um mecanismo ao nível IP da pilha protocolar. Para poder explicar melhor o mecanismo, irá ser usada seguinte terminologia:

- *Mobile Node* (MN): é o terminal móvel que altera o seu ponto de acesso.

- *Home Network* (HN): rede a qual o *mobile node* pertence.
- *Home Address*: endereço do terminal na *home network* e o qual é usado pelo *mobile node* para ser identificado.
- *Home Agent* (HA): é uma entidade (normalmente o router) que encontra-se na *home network* e é encarregue de encaminhar os dados para a localização actual do *mobile node*.
- *Foreign Network* (FN): rede à qual o *mobile node* está a visitar (rede estrangeira).
- *Foreign Agent* (FA): é a entidade que se encontra na rede estrangeira e fornece serviços aos terminais móveis registados (*mobile nodes*). Processa os pacotes que lhe são enviados pelo *home agent*.
- *Care-of-Address* (CoA): endereço atribuído ao *mobile node* pelo *foreign agent* quando se encontra na rede estrangeira.
- *Correspondent Node* (CN): terminal que está a comunicar com o *mobile node*.

Numa rede IP, o terminal precisa de ter um IP estável para poder ser identificado e localizável por outros terminais. Mas ao usar um IP estável, não é possível mover-se para outra rede com esse endereço, dado que esta nova rede (rede estrangeira) encontra-se numa gama de endereços diferentes e os pacotes não irão ser encaminhados. O Mobile IP visa resolver este problema ao fornecer um mecanismo que permita manter o mesmo endereço ao mudar de rede. Este objectivo é conseguido ao usar dois IPs: um para identificação (*home address*), que é permanente ao atravessar diferentes redes e permite manter as conexões estabelecidas com os outros terminais; e outro IP para o encaminhamento de pacotes (*care-of-address* - CoA), que muda quando o terminal move-se para uma nova rede e permite encaminhar os pacotes [23]. Os outros terminais só precisam de saber o *home address* e enviam os pacotes para a rede respectiva (*home network*) onde o *home agent* encarrega-se de reenviar os pacotes para a localização actual do terminal receptor.

O protocolo está dividido em 3 processos: *agent discovery*, registo e *tunnelling* [22]. O *agent discovery* é quando o HA e FA anunciam o suporte ao Mobile IP. Um *mobile node* que chegou a uma rede pode enviar uma mensagem (*solicitation message*) para descobrir se está um HA ou FA presente. O processo de registo é quando o *mobile node*, ao encontrar-se fora da *home network*, regista a sua nova localização no HA. Por último, o *tunnelling* é quando o HA encapsula os pacotes para poderem ser entregues na localização actual do *mobile node*, isto é, enviados para o endereço CoA. O *tunneling* pode ser feito entre o HA e FA ou pode ser feito directamente entre o HA e o MN. Um mecanismo importante do MIPv6 é o *Route Optimization*, o qual permite resolver o problema de usar o HA como ponto intermédio para

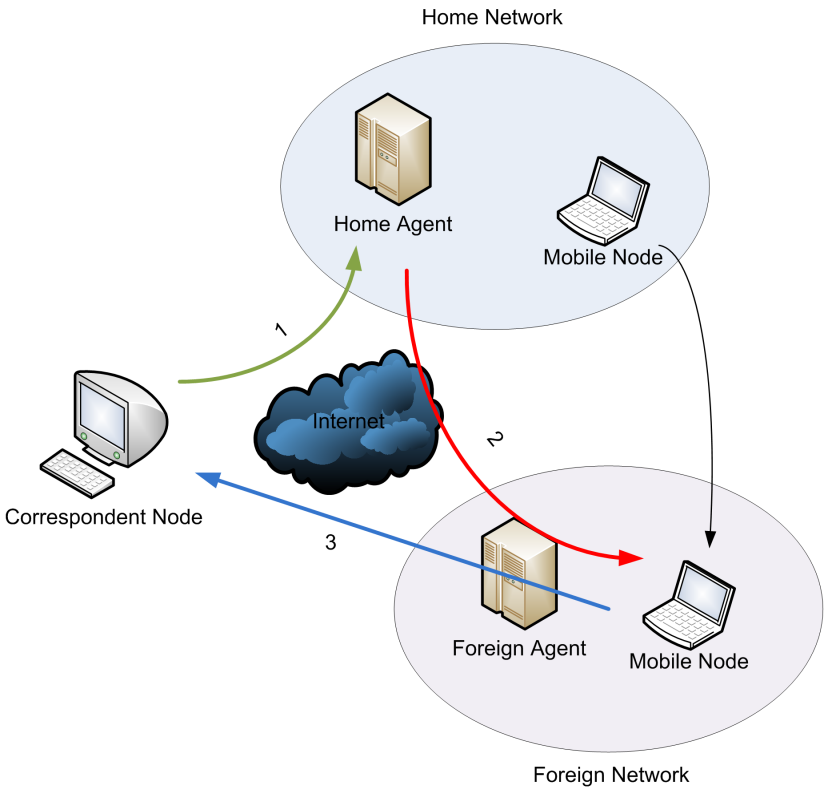


Figura 2.4: Topologia do Mobile IPv4

enviar e receber pacotes do CN, sendo um ponto de falha à medida que o tráfego e número de *mobile nodes* registados aumenta. O objectivo deste mecanismo é criar um túnel directo entre o MN e o CN, sem ser preciso o tráfego passar pelo HA. Este mecanismo permite que os outros terminais saibam o CoA do *mobile node* e assim criar um túnel directamente com o MN, sem ser preciso o tráfego passar pelo HA.

A Figura 2.4 ilustra a topologia do MIPv4. Quando o MN move-se da *home network* para a *foreign network*, é criado um túnel (2) entre o HA e o MN. Os pacotes enviados pelo CN para o MN são interceptados pelo HA (1) e reenviados para o MN pelo túnel (2). Se é utilizado o *Route Optimization*, é criado um túnel directo entre o CN e o MN (3).

### Mobile IPv6

O MIPv6 (RFC3775) [24] baseia-se nos mesmos conceitos do MIPv4, mas tem um melhor desempenho devido às novas funcionalidades do IPv6. Ao contrário do MIPv4, no MIPv6 não existe o conceito de *foreign agent*. No MIPv4, esta entidade é usada para obter o CoA e para o MIPv4 detectar movimento. No MIPv6, para obter CoA e detectar movimento é usado um

mecanismo do próprio IPv6, nomeadamente, o *Neighbor Discovery* (*Router Advertisements* e *Router Solicitation*). Com este mecanismo, o MN pode detectar que encontra-se numa nova rede (*foreign network*) e conFigura automaticamente o seu CoA. A Figura 2.5 apresenta a arquitectura do MIPv6.

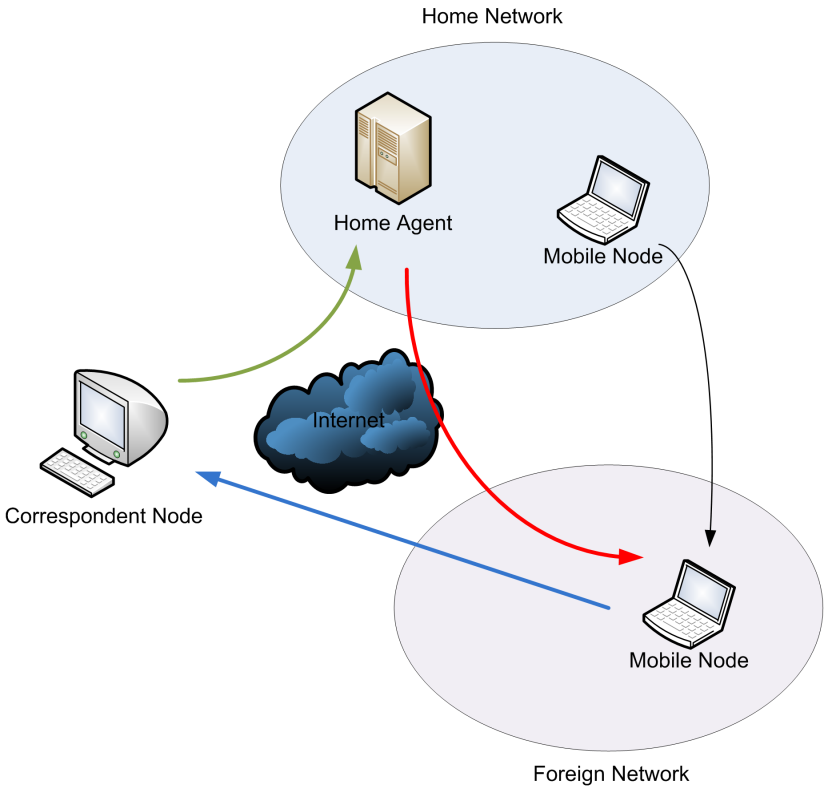


Figura 2.5: Topologia do Mobile IPv6

Permite, tal como no MIPv4, usar o *route optimization* para o MN e o CN criarem um túnel directamente entre os dois. Por outro lado, este túnel pode ser feito de uma maneira segura com o uso de um cabeçalho de autenticação do IPv6 [25].

### Fast Mobile IPv6

O FMIPv6 (RFC5568) [26] é uma extensão do MIPv6 e visa melhorar os procedimentos do MIPv6 de modo a diminuir o atraso no *handover*, ao eliminar as latências na detecção de movimento, conFiguração do CoA e teste do CoA (DAD do IPv6) [26]. O protocolo tem como objectivo descobrir o novo router de acesso e o prefixo de rede (para o novo CoA) antes de perder a ligação com o router anterior. A Figura 2.6 apresenta a arquitectura do FMIPv6.

Os routers de acesso têm uma tabela com o prefixo de rede e endereço MAC dos vizinhos,

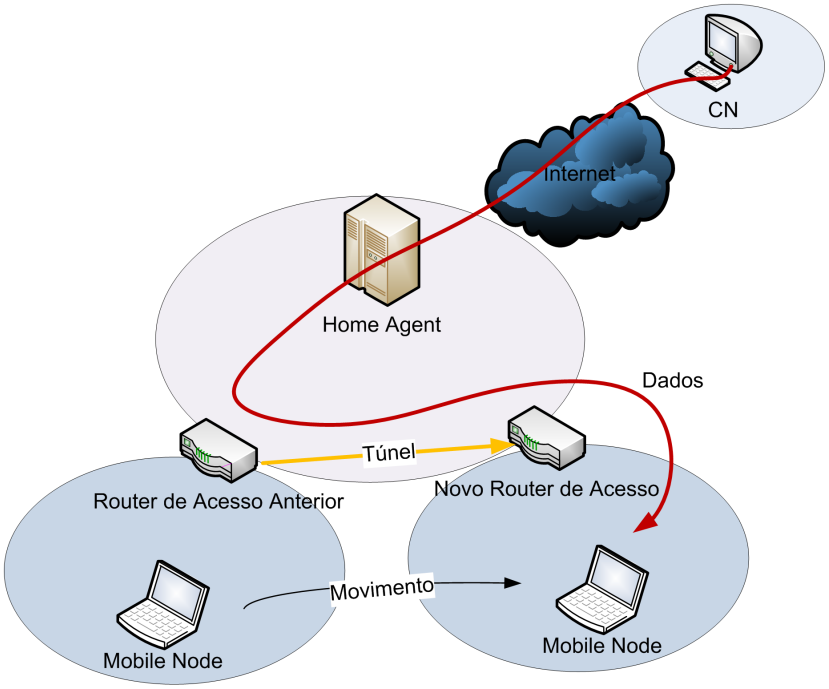


Figura 2.6: Topologia do Fast MIPv6

de modo a que o terminal possa solicitar o prefixo de rede ao router a qual está conectado e formar o novo CoA antes de ligar-se ao novo router de acesso. Com o objectivo de diminuir a perda de pacotes, o router actual estabelece um túnel com o novo router e reencaminha os datagramas para o novo router, que faz *buffering* dos pacotes até o terminal tiver terminado o *handover*.

### Proxy Mobile IPv6

No MIPv6 é necessário que o *mobile node* tenha funcionalidades de mobilidade IP. Existe troca de mensagens de sinalização entre o MN e o HA para poder manter a conectividade IP. No entanto, o PMIv6 (RFC5213) [27] baseia-se na abordagem da rede ter controlo da mobilidade IP do MN sem este intervir no processo. A rede é responsável de gerir todos os mecanismos de mobilidade necessários em vez de serem feitos no *mobile node*. O conjunto de redes onde existe suporte de PMIPv6 é definido como domínio PMIPv6. As novas entidades neste protocolo são as seguintes:

- *Local Mobility Anchor* (LMA): é o *home agent* para os *mobiles node* que estão dentro do domínio PMIPv6. Faz a gestão dos prefixos de rede do MN (*home address*) e possui a informação do localização dos MN.

- *Mobile Access Gateway* (MAG): é o router de acesso de uma rede que gere toda a sinalização relativa aos terminais ligados à sua rede. Detecta os movimentos dos terminais e actualiza a informação no LMA.

## 2.3 IEEE 802.21

O protocolo IEEE 802.21 - *Media Independent Handover Services* [5] especifica um conjunto de mecanismos que optimiza e facilita o *handover* entre diferentes tecnologias de acesso, sejam redes da família 802 ou outras, como o 3GPP GPRS/UMTS/LTE. Este protocolo tem com objectivo facilitar a descoberta de redes e processo de selecção através da recolha de informação local do terminal e remota de uma outra entidade 802.21 na rede. Através de uma interface abstracta, as camadas superiores podem controlar as interfaces de rede e o próprio *handover*, normalmente iniciado devido à degradação do sinal recebido ou à mudança para uma rede com melhor qualidade de serviço.

### 2.3.1 Arquitectura

O protocolo define 3 componentes (ver Figura 2.7): o *Media Independent Handover Function* (MIHF), *Service Access Points* (SAP) e *MIH User* (MIHU). O MIHF é componente principal da protocolo, o qual expõe serviços abstractos aos MIHU através de uma interface SAP. SAP é uma interface de comunicação entre o MIHF e as outras camadas (e.g. interfaces de rede, *MIH Users*, entidades remotas, etc.). E finalmente, os MIHU são as entidades que tiram partido dos serviços do MIHF e gerem o processo de mobilidade (*handover*). Estes podem estar localizados no próprio terminal ou na rede.

O MIHF tem 3 SAPs que fornecem serviços síncronos e assíncronos:

- MIH-SAP: fornece uma interface unificada/abstracta para comunicação entre o MIHF e as camadas superiores (MIHU).
- MIH-LINK-SAP: fornece uma interface entre o MIHF e as interfaces de rede.
- MIH-NET-SAP: fornece uma interface para trocar mensagens entre o MIHF local e um MIHF remoto.

A vantagem de usar uma interface abstracta é que o MIHU pode gerir o processo de *handover* sem conhecer as especificidades de cada tecnologia de rede. Logo, esta *framework* é uma solução escalável e eficiente para suportar *handovers* entre tecnologias de acesso diferentes. Além disso, para adicionar suporte a uma nova tecnologia, só é necessário fazer uma

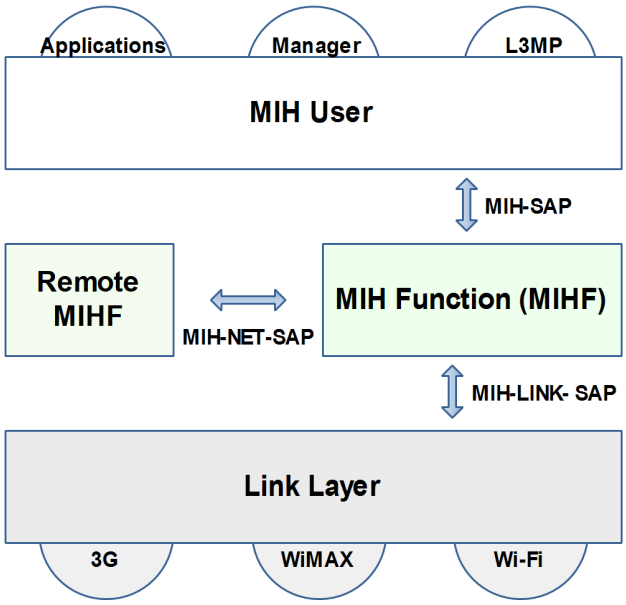


Figura 2.7: Arquitectura do IEEE 802.21

extensão para assegurar interoperabilidade com as outras tecnologias de acesso suportadas na *framework*.

### 2.3.2 MIHF Services

O MIHF está composto por 3 principais serviços que interagem com as camadas superiores (MIHU) e camadas inferiores (interfaces de rede). Os serviços são conFigurados e geridos por um quarto componente, o *Management Service* [28]. Este serviço consiste no *MIH capability discovery* (descobrir outras entidades MIHF), registo dos MIHF e subscrição de eventos. A Figura 2.8 mostra a arquitectura dos serviços na *framework* MIH.

#### Media Independent Event Service

O *handover* é iniciado normalmente com base na qualidade do serviço que o terminal está a receber, seja pelo terminal ou pela rede. Logo, o MIES é um serviço que permite reportar estes eventos às camadas superiores que gerem a mobilidade. Para vários MIHU poderem receber eventos de um MIHF, é usado um mecanismo de subscrição onde os MIHU subscrevem-se aos eventos que pretendem receber num determinado MIHF. Estes eventos podem indicar só uma mudança de estado, como *Link Going Down*, ou podem incluir outros parâmetros, como no caso do *Link Up*, o qual pode incluir o IP atribuído na nova rede. Os MIHU podem

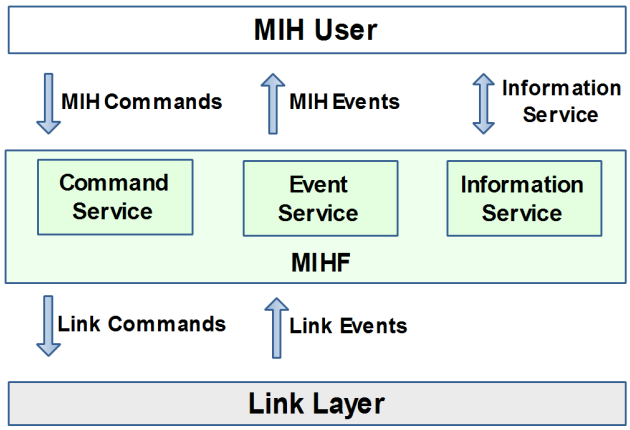


Figura 2.8: Arquitectura dos serviços MIH

subscrever-se tanto a eventos de MIHFs locais ou a eventos de MIHFs remotos, os quais são transportados pelo *MIH Protocol*.

Os eventos podem ser designados de eventos lógicos ou eventos MIH [5]. Os eventos lógicos são originados nas camadas inferiores e propagados para o MIHF local. Os eventos MIH podem ser originados do próprio MIHF ou a partir de eventos lógicos e são propagados para os MIHU (locais ou remotos). Quando um MIHU subscreve-se a um evento local, a indicação é entregue directamente. No caso de subscrever-se a um evento remoto, a indicação é enviada do MIHF local para o MIHF remoto pelo protocolo MIH e depois entregue ao MIHU.

### Media Independent Command Service

Os MIHU usam o MICS para controlar e monitorizar o estado das interfaces de rede. Este serviço fornece comandos que permitem obter o estado de várias interfaces de rede no terminal e executar comandos para mudar o estado (e.g. desligar interface de rede e ligar outra interface). Alguns comandos servem para configurar o MIHF e este gerar automaticamente eventos em determinadas condições/situações. As informações que podem ser obtidas dos comandos do MICS são tipicamente dinâmicas, como potência de sinal, obter os pontos de acesso à volta do terminal, etc.

Tal como acontece no MIES, existem dois tipos de comandos: comandos lógicos e comandos MIH. Os comandos MIH são usados pelo MIHU para executar acções no MIHF. Os comandos lógicos são usados MIHF para executar acções nas interfaces de rede. Os comandos MIH podem ser locais ou remotos, enquanto comandos lógicos são só locais. Tal como nos eventos, o envio de comandos remotos entre dois MIHFs é feito pelo *MIH protocol*.

#### Media Independent Information Service

O MIIS é um serviço que permite ao MIHU, seja local ou remoto, aceder a informações sobre as redes vizinhas, as quais são usadas para assistir o *handover*. Toda a informação obtida por este serviço é guardada numa entidade chamada *Information Server* (IS)[12]. A informação pode ser representada em XML ou ASN1, e pode conter informação de diferentes tecnologias de acesso (802 e não 802). Por exemplo, esta informação pode ser usada para obter as redes à volta do terminal, sem este ter que realizar um *scan* activo para descobrir os pontos de acesso, logo poupa energia (crítica em dispositivos de pequena dimensão).

A informação pode ser estática ou dinâmica. Por exemplo, informação estática pode ser o nome das redes vizinhas, enquanto informação dinâmica pode ser os parâmetros de rede como canal de radio, recursos disponíveis, etc. [29].

#### Management Service

Este serviço está composto por 3 funcionalidades que permitem gerir e configurar os outros serviços MIHF. As funcionalidades são as seguintes:

- *MIH capability discovery*: permite aos MIHUs descobrir as capacidades dos serviços disponibilizados por MIHFs locais ou remotos, isto é, interfaces de rede suportadas pelo terminal, eventos e comandos.
- *MIH registration*: usada pelo MIHU para registar-se num MIHF e poder ter acesso aos serviços disponibilizados por este.
- *MIH event subscription*: permite ao MIHU subscrever-se a eventos num MIHF local ou remoto.

### 2.3.3 Handover control model

O *handover* pode ser controlado por vários tipos de entidades, onde o IEEE 802.21 tem um papel importante em facilitar este controlo. Os tipos de controlo do *handover* são os seguintes:

- Controlado pelo terminal: o terminal recebe os eventos e informações do estado das interface, e toma a decisão de executar o *handover*.
- Controlado pela rede: o rede controla o *handover* e baseia a decisão na qualidade do sinal recebida do terminal ou em políticas mais complexas (e.g. contexto).

[12]Servidor de Informação

- Controlado pelo terminal, assistido pela rede: o terminal controla o *handover* e obtém informações da rede para executar o *handover* (e.g. células vizinhas ou recursos disponíveis).
- Controlado pela rede, assistido pelo terminal: a rede controla o *handover* e recebe eventos do terminal (e.g. alteração da força do sinal).

## 2.4 Android

O Android é um sistema operativo *open-source* para terminais móveis baseado no Linux (*Android Open Source Project*), desenvolvido pela Google e pelo consórcio *Open Handset Alliance*, que inclui empresas como a HTC, Samsung, Nvidia, Intel, etc. Este consórcio foi criado com o objectivo de inovar o mercado móvel ao criar software aberto e assim permitir a cooperação entre as várias empresas dos diferentes sectores para poder responder rapidamente as necessidades emergentes dos utilizadores [30]. O Android é disponibilizado com uma licença *open-source* que permite a qualquer pessoa usar e modificar o código, podendo disponibilizar o código para a comunidade *open-source* ou manter o código proprietário [31]. Ao longo dos últimos anos, o Android tem ganho mercado dos terminais móveis significativamente. Devido a ser uma plataforma *open-source*, o Android está a ser utilizado não só em *smartphones*, como em electrodomésticos, televisores, *tablets* e *netbooks*.

O Android é focado nas aplicações e no desenvolvimento de aplicações por entidades e programadores externos, por isso disponibiliza uma *framework* de desenvolvimento de aplicações em Java, *Software Development Kit* (SDK). Disponibiliza ainda uma extensão a esta *framework*, *Native Development Kit* (NDK), que permite usar código C/C++ nas aplicações Java. Isto é possível com o uso do *Java Native Interface* (JNI), que permite invocar funções C/C++ a partir do Java e vice-versa. O SDK inclui um conjunto de APIs que permite aceder aos recursos e serviços do sistema.

Devido ao Android baseia-se no Linux, é possível usar as ferramentas disponibilizadas no código fonte para compilar aplicações nativas para o sistema (e.g compilador *cross-platform* para processadores ARM). No entanto, estas interfaces limitam-se à linha de comandos ou um processo em *background*, devido a que o Android não usa um gestor de janelas nativo.

### 2.4.1 Arquitectura

Como mostra a Figura 2.9, a arquitectura do Android está composta por 5 camadas: aplicações, *application framework*, bibliotecas nativas, *Android runtime* e o *kernel* Linux [32]. O *kernel* Linux é a camada de abstracção entre o *hardware* do dispositivo e o sistema Android. Fornece os serviços do Linux como segurança, gestão de memória, gestão de processos, etc.

Por cima desta camada, existe uma camada de abstracção do *hardware*. Esta camada poderia estar ao nível do *kernel*, mas devido ao Linux ser GPL e muitos fabricantes não quererem disponibilizar o código dos seus *drivers*, a Google optou por criar uma camada especifica que não obriga a ser código aberto e que não fica completamente integrado com o kernel, logo não se aplica a licença GPL.

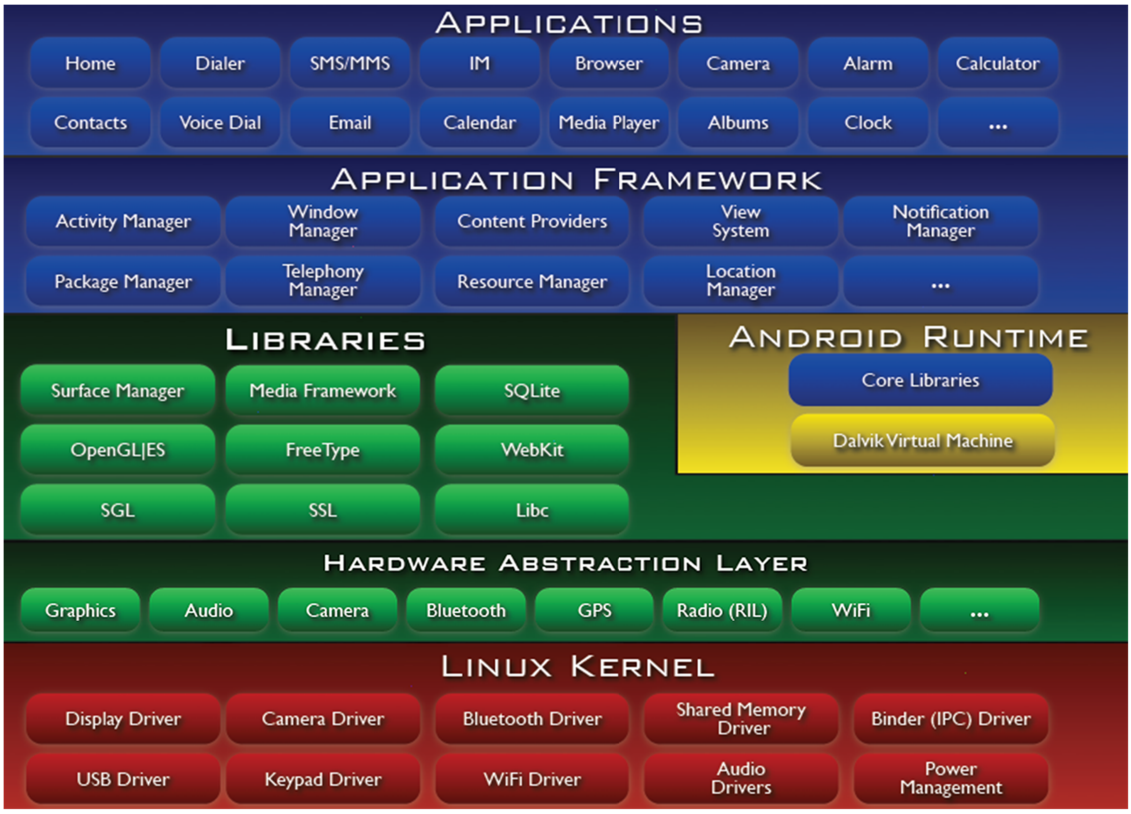


Figura 2.9: Arquitectura do Android [32]

As bibliotecas[13] fornecem um conjunto de funcionalidades aos componentes do sistema Android. O *Android Runtime* é o *core* do sistema que gere todas as aplicações e serviços. Faz a gestão do ciclo de vida dos processos e gestão do processamento e memória usada pelos mesmos. Está composto por dois componentes: as bibliotecas *core* do Java; a *Dalvik Virtual Machine* é a máquina virtual Java que corre as aplicações. A *Application Framework* fornece as APIs para as aplicações poderem interagir com o sistema, aceder seus recursos e ao *hardware*. Por último, a camada das aplicações inclui um conjunto base de aplicações que já vêm incluídas no sistema, como conFigurações, calendário, contactos, etc. Nesta camada também encontram-se as aplicações instaladas posteriormente.

A estrutura de dados do Android está divida nas seguintes partições:

[13]Em inglês, *libraries*.

- *boot*: partição onde encontra-se o *kernel* Linux e os ficheiros de iniciação do sistema (e.g. *init.rc*).
- *system*: partição onde está o sistema Android.
- *recovery*: usada para recuperação do sistema (e.g. recuperação ou *upgrade* do sistema).
- *user data*: aplicações instaladas e dados do utilizador.
- *cache*: usada para fazer *cache* dos dados das aplicações.
- *sdcard*: memória interna do telemóvel (e.g. fotos, vídeos e músicas):

A utilização de uma partição para o *kernel* Linux e outra para o sistema Android, permite mudar um dos componentes sem ter que mudar o outro. Por exemplo, é possível mudar o sistema Android para uma versão mais recente sem mudar o *kernel* usado. Por outro lado, pode-se instalar um *kernel* Linux com conFigurações diferentes (e.g. IPv6, mobilidade, etc.), sem mudar o sistema. No entanto, uma versão do sistema Android pode exigir uma determinada versão do *kernel* Linux.

### 2.4.2 Aplicações no Android

O desenvolvimento de aplicações para Android tem sido bem aceite pela comunidade de programadores, dado que usa uma linguagem orientada aos objectos mais utilizada, o Java. Por outro lado, além de ser um projecto *open-source*, as ferramentas necessárias para desenvolver as aplicações são disponibilizadas gratuitamente. As aplicações são depois disponibilizadas através da loja virtual Android Market, onde um utilizador pode vender ou disponibilizar gratuitamente a sua aplicação. Esta loja virtual permite a partir do telemóvel instalar facilmente as aplicações, sendo necessária ligação à Internet. No entanto, podem ser directamente instaladas no Android, sem precisar de ser através da loja virtual. Para aplicações que são vendidas a um preço, a Google fica com 30% do valor de venda, sendo o resto para o criador. Outra forma dos criadores obter receitas é incluir publicidades na aplicação.

As aplicações são compiladas para *byte-code* (DEX) e correm sobre uma máquina virtual optimizada para o Android, a *Dalvik Virtual Machine*. Este tipo de *byte-code* está optimizado para que as aplicações usem a mínima quantidade de memória possível. Com o objectivo de tornar segura a execução de aplicações, o Android implementa uma técnica designada *sandboxing*. Baseia-se em executar cada aplicação isoladamente, isto é, numa instância da máquina virtual diferente e associada a uma conta do Linux diferente com permissões específicas, e evita assim que ataques feitos a uma aplicação sejam propagados ao resto das aplicações ou ao próprio sistema Android. O Android usa um sistema de permissões baseado em etiquetas,

as quais representam os recursos e interfaces as quais a aplicação tem acesso. As aplicações do próprio sistema têm permissões não disponíveis a aplicações externas.

Existem 4 tipos de componentes para construir uma aplicação, com objectivos diferentes [33]:

- *Activity* (Actividade): permite a interacção com o utilizador por meio do ecrã táctil e o teclado. Tipicamente, cada interface gráfica mostrada numa aplicação é uma *Activity* diferente. Enquanto uma actividade está a ser mostrada, as outras permanecem suspendidas, independentemente da aplicação a qual pertencem.

- *Service* (Serviço): os serviços permitem realizar processamento em *background* e não têm interface gráfica. Os serviços podem fornecer uma API remota RPC (*Remote Procedure Call*), a qual permite as outras aplicações invocar remotamente métodos num determinado serviço.

- *Broadcast Receivers*: fornece um mecanismo assíncrono de notificações de eventos. Permite a uma aplicação executar um determinado código ao receber um determinado evento, por exemplo, quando o sistema termina a fase de inicialização.

- *Content Providers*: permite partilhar dados entre as aplicações. Uma aplicação implementa uma API de forma a outras aplicações lerem ou escreverem dados.

A comunicação entre aplicações é suportada pelo mecanismo *Binder*, que implementa um mecanismo RPC e permite a troca de mensagens entre componentes de diferentes aplicações. As mensagens são transportadas pelo objecto *Intent* e podem conter uma *action string* (acção). Podem ser definidos filtros de *Intents* para receber determinadas acções e invocar um *Broadcast Receiver*. A definição da API remota é feita com a linguagem AIDL, que permite declarar os métodos que podem ser invocados por outras aplicações.

### 2.4.3 Conectividade

#### Wi-Fi

A conectividade Wi-Fi é suportada por 3 componentes: o driver, *wpa-supplicant* e o *WifiManager*. O driver é implementado por um módulo do *kernel*. O *wpa-supplicant* é um serviço nativo[14] que abstrai o driver Wi-Fi e disponibiliza mecanismos de controlo da interface e autenticação na rede [34]. O *WifiManager* disponibiliza uma API na *application framework* para controlar a interface Wi-Fi através do *wpa-supplicant* (*java.android.net.wifi*) [30]. O próprio

[14]Normalmente designado como *daemon*

sistema Android usa esta API para controlar a interface de acordo com as preferências do utilizador.

### 2G & 3G

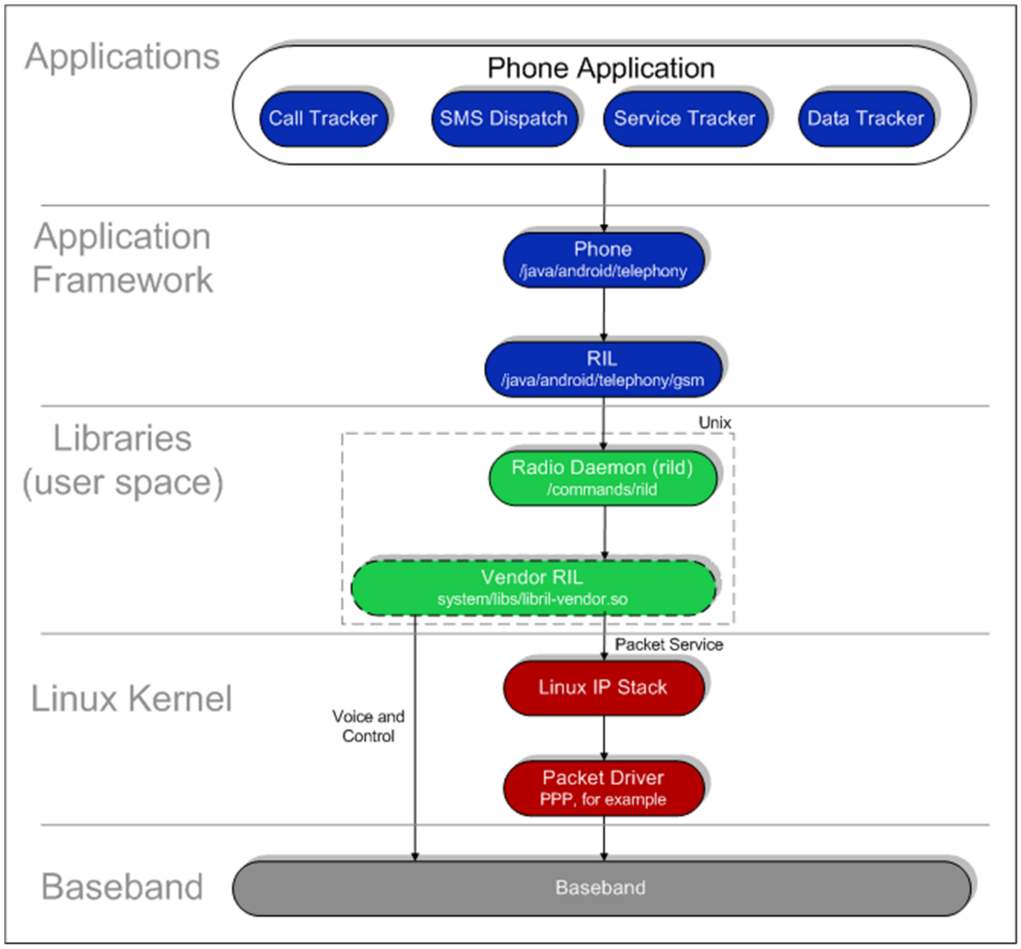


Figura 2.10: Arquitectura 2G & 3G do Android [35]

A Figura 2.10 mostra a arquitectura que suporta a conectividade 2G e 3G do Android. Esta arquitectura está composta por 5 níveis: o processador de comunicação (*baseband*), *kernel* Linux, o processo RILD, a API na *application framework* e as aplicações que usam esta API. O componente principal é o processo RILD pois fornece uma API entre os serviços do Android e o *hardware* (driver). O componente *Vendor* RIL é o driver especifico para o *baseband*, normalmente código fechado e proprietário. A comunicação entre o RILD e os serviços do Android é feita por sockets Unix.

O serviço *TelephonyManager* na *application framework* fornece uma API muito incompleta

para as aplicações. No caso da *WifiManager*, a API disponibilizada permite realizar um conjunto alargado de operações. No entanto, na API do *TelephonyManager*, operações como desligar a conexão de dados ou a interface 3G não é possível [30]. As únicas operações possíveis permitem obter o estado da ligação de rede ou das chamadas. Operações que alterem o estado da ligação só estão disponíveis para aplicações do próprio sistema (e.g. aplicação que realiza as chamadas) através de uma API interna, a qual o *TelephonyManager* usa mas não disponibiliza todas as suas operações nem permite o seu acesso.

### 2.4.4 Código Fonte

Dado que o Android é um projecto *open-source*, o código fonte pode ser facilmente obtido através do seu próprio *website* [36]. O código inclui um sistema de compilação[15] optimizado, onde se pode seleccionar o *target*[16], adicionar um novo tipo de *target* (inclui drivers, *kernel*, bibliotecas proprietárias) ou adicionar novas aplicações ao sistema. A estrutura das *makefiles* é diferente das *makefiles* do GNU. Possui uma estrutura específica que abstrai o *linking* de bibliotecas ou *includes* de cabeçalhos C/C++ e usa um compilador *cross-platform* para processadores ARM, incluído no sistema de compilação.

Ao compilar o código fonte são geradas várias 'imagens' para as diferentes partições do Android (e.g. *boot*, *system*, etc.). Depois usa-se uma ferramenta disponibilizada pelo sistema de compilação (*fastboot*) que permite instalar as imagens no terminal. Por outro lado, pode-se usar o emulador (também incluído no código fonte) e carregar as imagens geradas. A instalação de imagens do Android não é possível em todos os terminais com Android, pois existem terminais específicos para desenvolvimento onde é possível instalar facilmente as imagens, e terminais de produção (comerciais) onde é necessário desbloquear o terminal para poder instalar as imagens.

Devido à licença do Android, que permite usar e modificar o código, o sistema tem sido adaptado para vários tipos de dispositivos. O Android não só é usado em *smartphones*, como também é usado em *tablets*, *netbooks* e até tem sido adaptado para electrodomésticos.

## 2.5 Sumário

Neste capítulo foram abordadas as tecnologias de rede 3G e Wi-Fi, o funcionamento dos mecanismos de mobilidade IP candidatos a serem usados neste trabalho e a norma IEEE 802.21 que permitirá optimizar o *handover* heterogéneo. Por último e mais importante, abordou-se a plataforma Android onde irá ser implementada a mobilidade heterogénea entre as tecnologias

[15] Em inglês, *toolchain*.
[16] Tipo do *hardware* destino. Pode ser um simulador ou um dispositivo real.

de acesso Wi-Fi e 3G. Estes conceitos abordados neste capítulo irão servir de base ao trabalho realizado. A plataforma Android é muito complexa e não suporta mobilidade entre diferentes redes de acesso, logo conhecimentos do seu funcionamento são necessários de modo a poder alterar o seu funcionamento com o objectivo de permitir o suporte à mobilidade heterogénea e *handovers make-before-break*.

# Capítulo 3

# Desenho e Implementação

Neste capítulo serão descritos todos os passos seguidos para a solução de mobilidade heterogénea no Android. Será feito um enquadramento da solução num cenário real e o seu funcionamento, de modo a especificar os requisitos a implementar no Android. Seguidamente, irá ser abordada a arquitectura da solução implementada e o seu funcionamento interno. Finalmente, serão explicadas as modificações feitas a alguns mecanismos do Android. Sendo assim, este capítulo abordará o trabalho que foi desenvolvido nesta dissertação de mestrado, desde a descrição dos vários componentes até alguns diagramas ilustrativos do modo de operação.

## 3.1 Arquitectura de Mobilidade Heterogénea

Nesta secção pretende-se enquadrar o trabalho desenvolvido no âmbito do tema desta dissertação, com especial ênfase nos aspectos que abordam o terminal Android. Neste contexto, o terminal Android é um elemento central numa arquitectura de mobilidade em ambientes heterogéneos. O protocolo IEEE 802.21 aparece aqui com o objectivo de optimizar e facilitar toda a gestão da mobilidade do terminal Android. No trabalho desenvolvido, o controlo da mobilidade do terminal está do lado da rede, sendo este protocolo que permitirá a uma entidade da rede gerir a mobilidade do terminal de uma forma optimizada, de modo a poder executar um *handover* correctamente e com um impacto mínimo na qualidade do serviço, isto é, um *handover make-before-break*.

A Figura 3.1 mostra um cenário geral onde o terminal Android está conectado a uma rede multi-acesso e move-se entre as diferentes tecnologias de acesso, nomeadamente, Wi-Fi, 3G e WiMAX. No entanto, os terminais que irão ser usados neste trabalho só irão ter as interfaces 3G e Wi-Fi, logo o WiMAX não irá ser suportado nesta solução. O gestor de mobilidade IEEE 802.21, chamado *Mobility Manager* (MM)[1], implementa um MIHU e tem como objectivo gerir

[1]Em português, gestor de mobilidade.

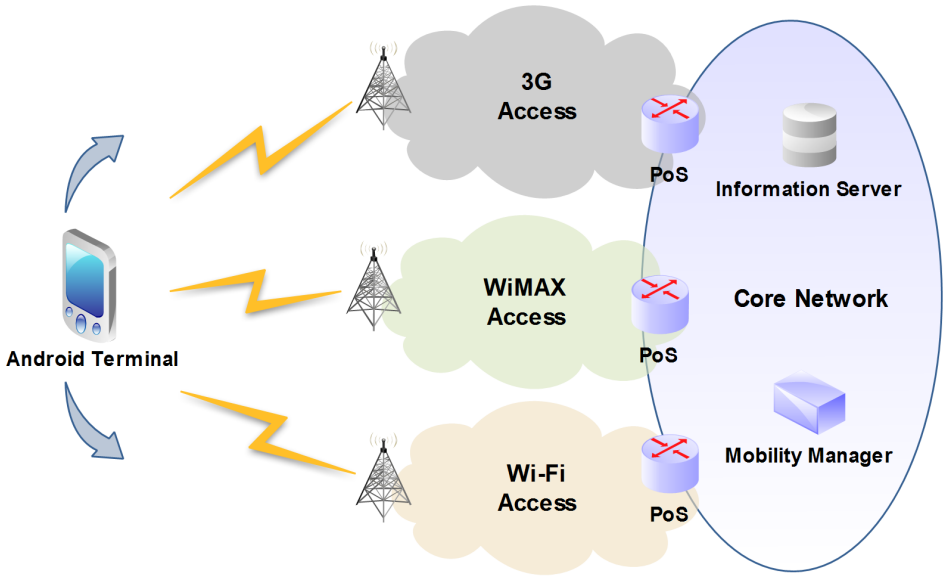


Figura 3.1: Cenário geral

a mobilidade dos terminais, numa abordagem em que o *handover* é controlado pela rede e assistido pelo terminal. As funcionalidades desta entidade são as seguintes:

- Configurar terminais de modo a receber eventos gerados nas interfaces de rede.
- Iniciar o *handover* tendo em base alterações das condições do terminal, preferências do utilizador ou politicas da rede, que podem basear-se em parâmetros QoS.
- Reservar e libertar recursos das redes (por exemplo, largura de banda).

Adicionalmente, este gestor de mobilidade interage com outras entidades IEEE 802.21 da rede para ajudar na decisão de *handover*, as quais são o IS e o PoS. O IS é uma base de dados com informações das redes, nomeadamente custo de utilização, autenticação da rede, nome da rede, etc. A partir destas informações e ao saber quais as redes à volta do terminal, o gestor de mobilidade pode decidir qual é a melhor rede para a qual efectuar o *handover*. O PoS é a entidade que se encontra na rede de acesso e permite saber os recursos disponíveis da rede e efectuar a reserva dos mesmos para um dado terminal.

O gestor de mobilidade da rede é que toma as decisões de mobilidade, mas cabe ao terminal executar o *handover* e controlar o mecanismo de mobilidade IP. Ao receber o pedido para iniciar o *handover*, o terminal deve comunicar com o mecanismo de mobilidade IP para iniciar o *handover* ao nível de rede, de modo a trocar de interface em uso e manter a conectividade IP. O mecanismo de mobilidade IP usado é o MIPv6 com um funcionamento optimizado para

*handovers make-before-break*, incluindo melhorias em relação ao funcionamento normal, nomeadamente, controlo externo sobre o *handover*. As principais funcionalidades deste mecanismo são as seguintes:

- Permite a interacção com entidades externas, nomeadamente, para o controlo do *handover*.
- Permite a escolha da interface de rede destino do *handover*, além de permitir executar o *handover* sem necessidade de perder conectividade na interface de rede, contrariamente ao que acontece com o MIPv6 normal.
- Suporte para túneis 6in4 (IPv6 sobre IPv4).

É sobre o terminal Android e a consequente interacção com entidades externas, nomeadamente o gestor de mobilidade e o MIPv6, que este trabalho vai incidir. Pretende-se investigar o sistema Android e implementar uma camada de software que permita a interacção do terminal, nomeadamente, interfaces de rede ou preferências do utilizador, com as diferentes entidades externas, de acordo com a norma IEEE 802.21 [7]. Por outro lado, pretende-se estudar e modificar o *kernel* Linux do Android, por forma a suportar o mecanismo de mobilidade MIPv6.

Através do terminal será possível obter informações sobre o estado das interfaces de rede e controlar o estado das mesmas, mediante o uso do protocolo IEEE 802.21. Igualmente, o terminal deve ter capacidade de notificar o gestor de mobilidade da rede quando a potência de sinal ultrapassar um determinado nível, de forma a mover o terminal para uma nova rede de acesso com melhores condições e assim evitar que haja uma perda da qualidade de serviço. Por outro lado, o terminal Android deve permitir ter as interfaces conectadas em simultâneo para poder executar um *handover make-before-break* e sem quebra de serviço. Finalmente, com o objectivo de avaliar o impacto da solução em ambientes heterogéneos, serão obtidas métricas de desempenho e QoS. Estas métricas servirão para analisar se a solução satisfaz os objectivos e permite também analisar quais são os pontos que devem ser melhorados, de forma a optimizar a solução.

Finalizado o enquadramento do terminal na arquitectura de mobilidade heterogénea e os diferentes componentes que irão interagir e a especificação dos requisitos, procede-se a análise do modo de operação da arquitectura de mobilidade, com ênfase para as mensagens que o terminal Android irá trocar com o gestor de mobilidade.

## 3.2 Modo de operação

Após apresentar os requisitos, irá ser abordado o processo de mobilidade e o fluxo de mensagens entre o terminal Android e o gestor de mobilidade. Esta interacção segue o que está definido na norma IEEE 802.21, mas com ligeiras alterações de modo a adaptar-se à arquitectura anteriormente apresentada. O processo de mobilidade é dividido em dois sub-processos distintos: configuração do terminal e o processo de *handover*. Na configuração é realizada um reconhecimento das capacidades do terminal e a configuração dos eventos a receber. O processo de *handover* inclui 4 fases distintas: inicio do *handover*, preparação para o *handover*, execução do *handover* e conclusão do *handover*.

### 3.2.1 Configuração do Terminal

Este processo é realizado quando o terminal é ligado à rede e engloba vários passos, de acordo com a norma IEEE 802.21, necessários para a configuração do terminal Android. A Figura 3.2 apresenta as mensagens IEEE 802.21 usadas entre o terminal e o gestor de mobilidade. Num primeiro passo o terminal regista-se no gestor de mobilidade para que este tenha conhecimento da sua existência e da necessidade de gestão da mobilidade do terminal. De seguida é feita a descoberta das interfaces de rede disponíveis no terminal e os eventos e comandos [2] suportados. Com base nesta informação, o gestor de mobilidade subscreve-se a determinados eventos de modo a recebe-los quando estes ocorram. Por último, o gestor de mobilidade define um *threshold*, isto é, a potência de sinal limite que ao ser ultrapassada o terminal notifica o gestor de mobilidade. Este *threshold* permite que quando a qualidade de uma ligação estiver a piorar, o terminal Android notifica o gestor de mobilidade e este inicia o *handover* para a outra interface de rede, para evitar a perda de conectividade do terminal Android.

### 3.2.2 Processo de *Handover*

Este processo é mais complexo e ocorre depois de ter o terminal estar devidamente configurado, sendo iniciado a partir do momento em que o terminal Android notifica que o *threshold* foi ultrapassado. Está dividido em 4 fases: iniciação, preparação, execução e conclusão.

#### Iniciação e preparação do *handover*

Após o *threshold* ser ultrapassado, o terminal Android envia uma mensagem *MIH-Link-Parameters-Report* para o gestor de mobilidade, como mostra a Figura 3.3. A partir deste

[2] Na norma IEEE 802.21 são também designados primitivas.

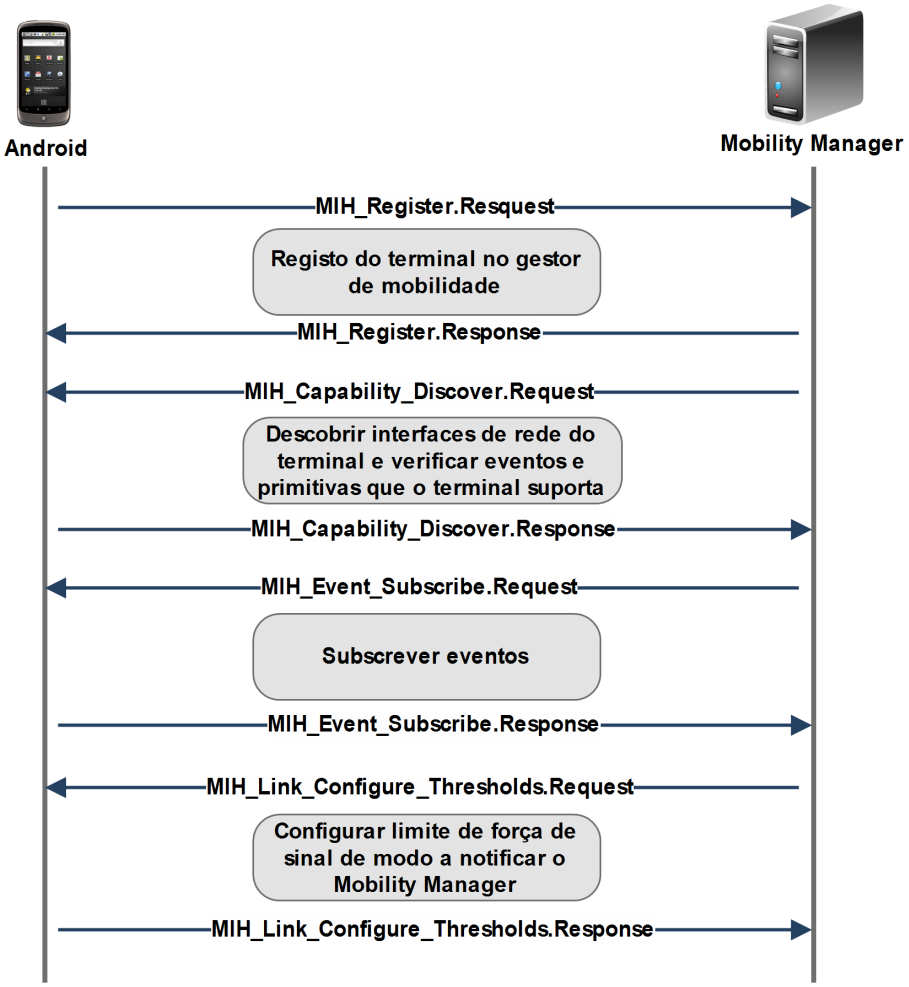


Figura 3.2: Processo de configuração do terminal Android

instante é iniciado o processo de *handover* para a outra interface de rede e inicia-se a fase de preparação para o *handover*.

De modo a descobrir as redes candidatas para as quais efectuar o *handover*, o gestor de mobilidade efectua um pedido ao terminal para ligar a interface de rede destino e obter as redes a volta deste. No caso da interface de rede 3G, gestor de mobilidade não recebe as redes a volta porque não é possível ter controlo sobre a escolha da célula UMTS a partir da API disponibilizada pelo Android, sendo possível só na interface de rede Wi-Fi.

Através da mensagem *MIH-Net-HO-CandidateQuery*, o gestor de mobilidade envia as redes candidatas ao terminal Android, o qual efectua uma intersecção das redes recebidas com as preferências do utilizador. A lista obtida a partir desta intersecção é enviada ao gestor de mobilidade e este verifica os recursos disponíveis nestas redes. Após o gestor de mobilidade

escolher a rede destino e ter efectuado a reserva dos recursos, o terminal Android recebe mensagem *Link-Action(PowerUp)* e estabelece a ligação com a rede escolhida. Finalmente, conecta-se à rede e envia o evento *Link-Up* ao gestor de mobilidade.

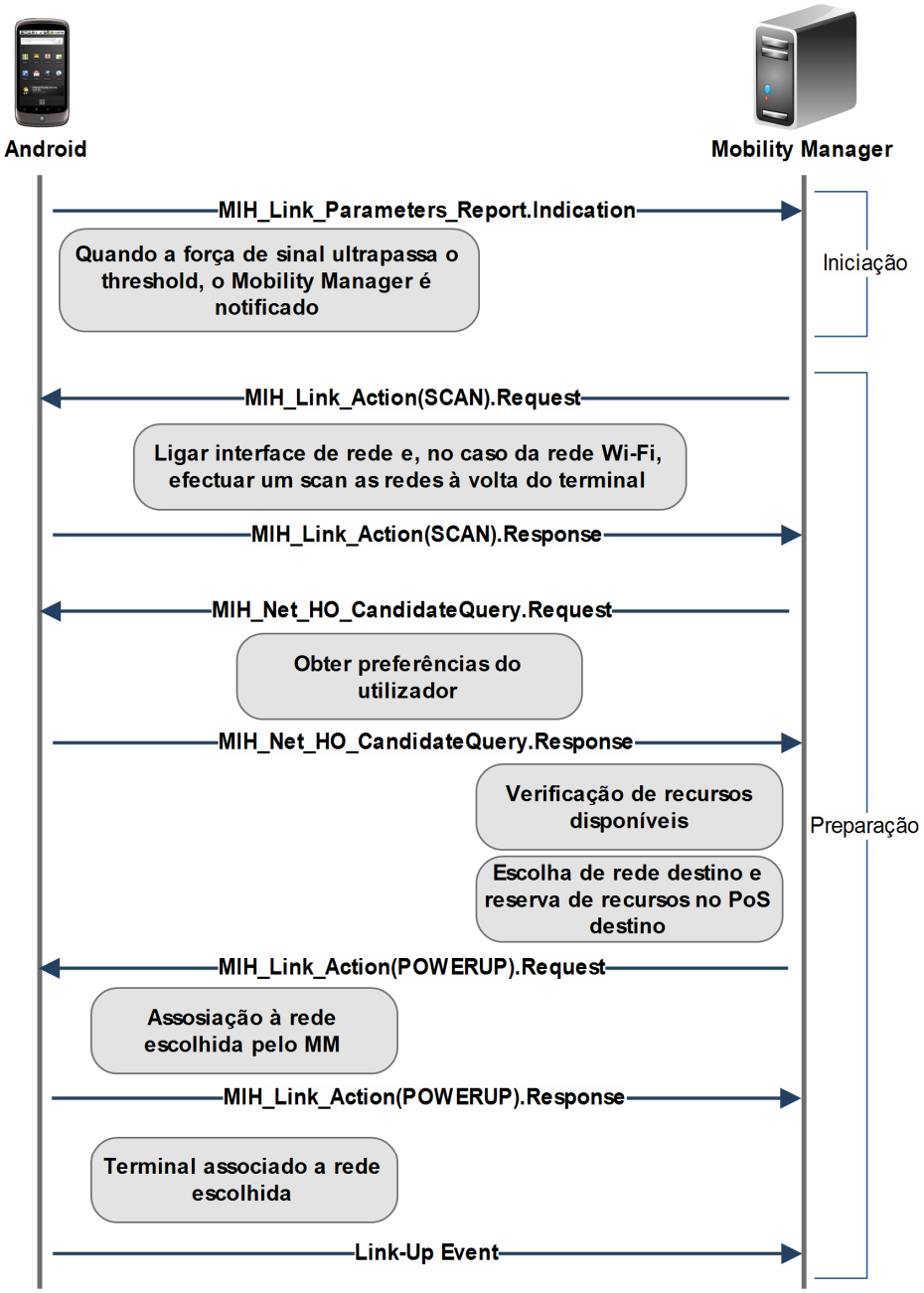


Figura 3.3: Fases de iniciação e preparação do *handover*

### Execução do *handover*

Na execução é desencadeado o mecanismo de mobilidade IP, de modo a manter a conectividade IP do terminal Android. Como apresentado na Figura 3.4, o terminal ao receber a mensagem *MIH-Net-HO-Commit*, efectua o *trigger* do *handover* para a interface de rede destino no mecanismo de mobilidade IP (MIPv6). O terminal notifica o gestor de mobilidade se o *handover* foi efectuado com sucesso.

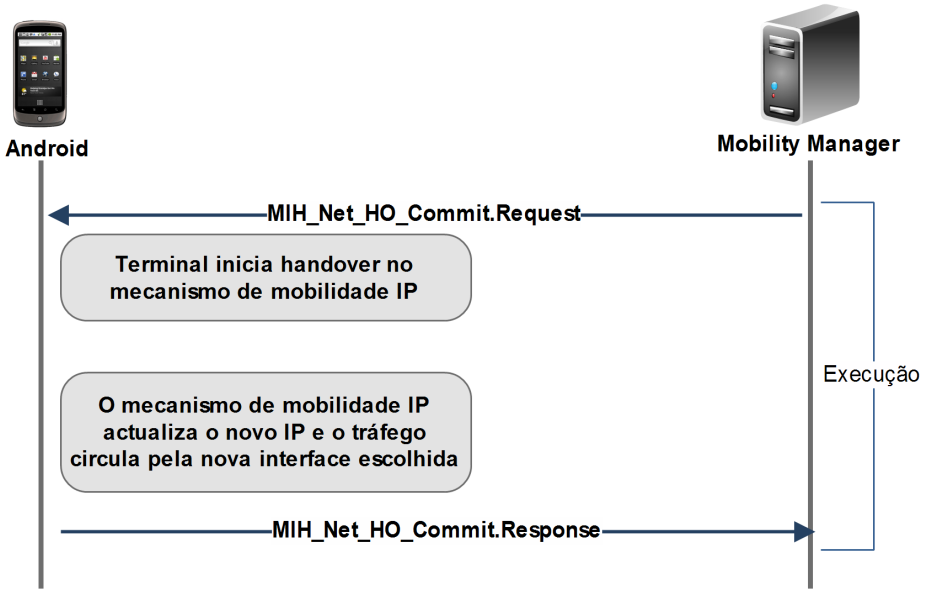


Figura 3.4: Fase de execução do *handover*

### Conclusão do *handover*

Nesta fase, apresentada na Figura 3.5, o terminal desconecta-se da rede anterior e o gestor de mobilidade liberta os recursos reservados na rede anterior.

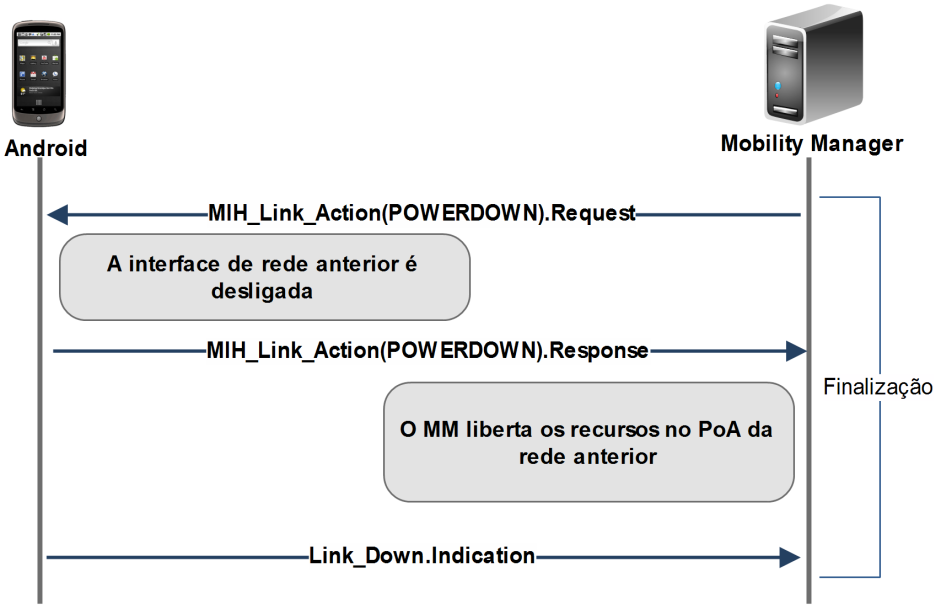


Figura 3.5: Fase de conclusão do *handover*

## 3.3 Arquitectura do software no Android

Nesta secção irá ser abordada camada de software que implementará a norma IEEE 802.21 e permitirá interagir entre o sistema Android, o mecanismo de mobilidade IP e o gestor de mobilidade. É um software que abstrai as especificidades do Android para com o gestor de mobilidade, nomeadamente com o uso do protocolo IEEE 802.21. A implementação desta camada de software foi realizada tendo por base a linguagem orientada aos objectos Java. A Figura 3.6, para além de demonstrar como se encontra estruturada a aplicação, reflecte também as interacções que existem com entidades externas. A seguir é apresentado um breve resumo sobre cada componente.

- Android Mobility Manager (AMM): implementa um MIHU e tem como objectivo gerir o mecanismo de mobilidade IP, gerir as preferências do utilizador e fornecer uma interface gráfica para mostrar informações ao utilizador sobre o estado do serviço.

- Android Interface Manager (AIM): tem como objectivo gerir a interface de rede, implementa os eventos e as primitivas definidas na norma IEEE 802.21 (*Link-SAP*).

- MIHF: implementa o MIHF da norma IEEE 802.21, o qual permite interagir com outros componentes, como o gestor de mobilidade.

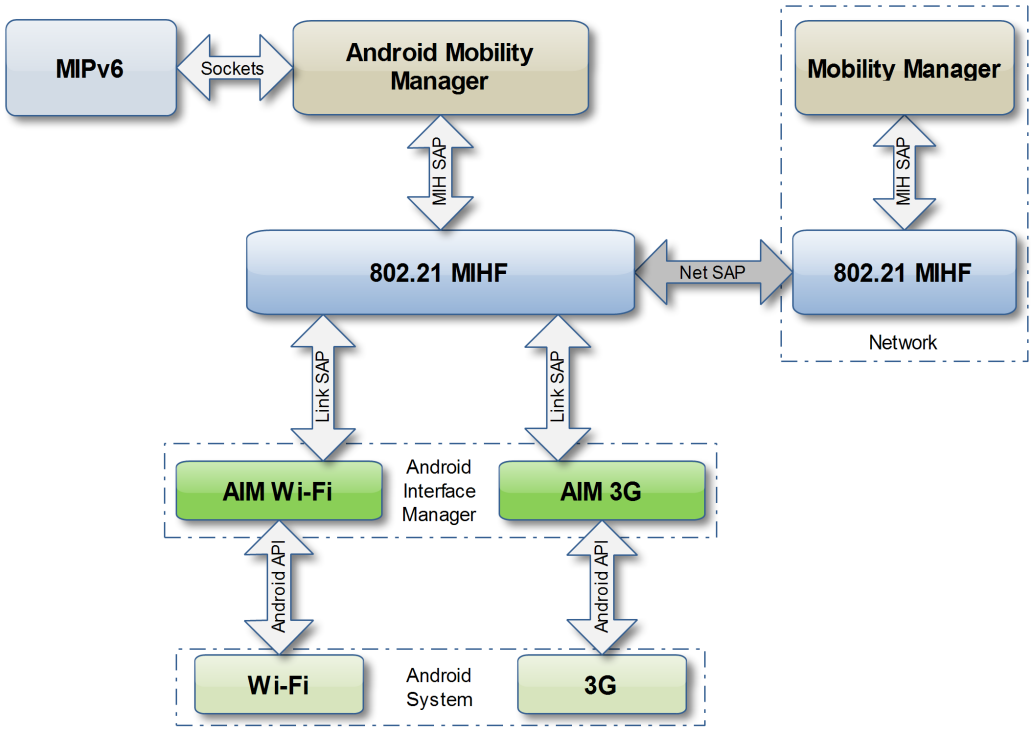


Figura 3.6: Arquitectura dos componentes a correr no Android

O MIHF é o ponto central que integra os diferentes componentes. Através deste são trocadas as mensagens entre o AIM, AMM e as entidades externas. A comunicação entre o AMM e o MIPv6 é feita directamente com o uso de *sockets* TCP. Na Figura 3.6 também se pode visualizar os componentes do lado da rede com os quais o Android interage, nomeadamente, o gestor de mobilidade (*Mobility Manager*) e o MIHF remoto.

Os 4 componentes que compõem a aplicação estão a correr como serviços (*services*), os quais são executados em *background* e foram implementados na linguagem Java. Ao ligar à aplicação, são iniciados os 4 componentes e o MIPv6, e são estabelecidas as conexões entre os componentes. Cada componente fica a correr independentemente num processo à parte, o que permite que alguma falha num dos componentes, não afecte os restantes. Por outro lado, o componente que falhou é automaticamente reiniciado pelo Android. Para poder mostrar informações ao utilizador, todos os serviços possuem mecanismos que permitem notificar o utilizador com mensagens que aparecem no ecrã, além de ficarem também armazenadas na barra de notificações do Android, de modo a poder serem visíveis posteriormente.

A implementação do IEEE 802.21 usada está escrita na linguagem C++ e pertence ao projecto HURRICANE, no qual este trabalho está inserido. Utilizou-se o mecanismo JNI para poder interagir entre o código Java e a implementação do MIH-SAP e Link-SAP. A seguir

irão ser explicados os componentes implementados no Android, nomeadamente, o Android Mobility Manager e o Android Interface Manager.

### 3.3.1 Android Mobility Manager

Como abordado anteriormente, este serviço tem o objectivo de gerir os pedidos referentes ao MIPv6, preferências do utilizador e interface gráfica. A Figura 3.7 mostra a arquitectura do AMM.

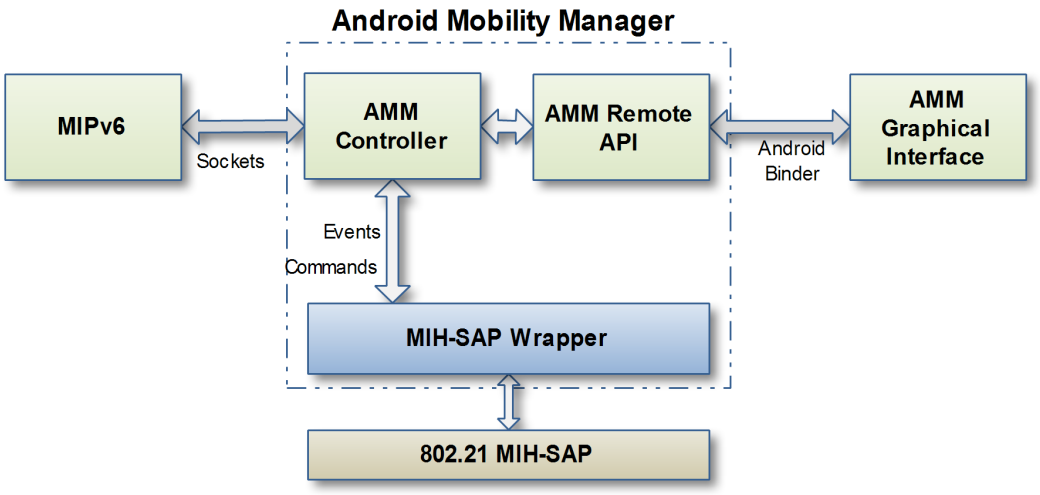


Figura 3.7: Arquitectura do Android Mobility Manager

As mensagens IEEE 802.21 suportadas pelo AMM são as seguintes:

- *MIH-Net-HO-CandidateQuery*: o AMM obtêm as preferências do utilizador armazenadas, efectua a intersecção com a lista de redes candidatas recebidas e devolve a lista resultante ao gestor de mobilidade.
- *MIH-Net-HO-Commit*: esta mensagem é usada para iniciar o *handover* no MIPv6 para uma determinada interface de rede, mas também ao usar parâmetros diferentes, pode ser usada para obter a interface de rede em uso actualmente pelo MIPv6.
- *MIH-MN-HO-Commit*: esta mensagem é enviada pelo AMM para o gestor de mobilidade quando o AMM quer iniciar explicitamente o *handover* para uma determinada interface.

A mensagem *MIH-Net-HO-CandidateQuery* é usada para obter as redes Wi-Fi preferidas do utilizador. Por outro lado, o próprio gestor de mobilidade tem conhecimento e não envia

esta mensagem quando a rede destino é 3G. O MIH-SAP Wrapper tem como objectivo integrar a implementação do MIH-SAP, em C++, com o código Java da aplicação, com o uso do JNI. Por outro lado, permite abstrair a implementação do IEEE 802.21 do AMM, de modo a poder facilmente mudar para outra implementação e alterar só o MIH-SAP Wrapper.

O AMM Controller processa as mensagens IEEE 802.21 recebidas do gestor de mobilidade e comunica com o MIPv6. Por outro lado, também fornece informação de estado do serviço a outros processos/aplicações a correr no Android, através da interface remota (AMM Remote API), definida em AIDL. O AMM Graphical Interface consiste na interface gráfica para o utilizador e é executada numa actividade (*Activity*) externa ao AMM. A ligação ao AMM é feita através mecanismo Android Binder, que permite criar uma ligação RPC entre a actividade e a interface remota do AMM. Esta interface gráfica fornece informação de estado sobre os diferentes serviços (AMM, MIHF, AIM e MIPv6), a interface de rede em uso pelo MIPv6, além de permitir iniciar ou terminar os serviços.

O AMM Controller também disponibiliza, através da interface remota, mecanismos para poder iniciar o *handover* explicitamente, sem ser iniciado pelo gestor de mobilidade devido a uma perda de sinal. Estes mecanismos foram implementados com o objectivo de serem usados só para testes, devido a que o cenário implementado define que o processo de *handover* é controlado pelo gestor de mobilidade na rede. Os mecanismos disponibilizados são dois: iniciar *handover* directamente no MIPv6, sem haver nenhum conhecimento do gestor de mobilidade; iniciar *handover* explicitamente no gestor de mobilidade, onde irá ocorrer o processo de *handover* normal. Estes mecanismos podem ser iniciados directamente através da interface gráfica.

A utilização do AMM Controller e a implementação da mensagem *MIH-MN-HO-Commit*, deixa em aberto a possibilidade de implementar uma arquitectura de mobilidade onde o terminal controla o *handover* e é assistido pelo gestor de mobilidade.

Na Figura 3.8 está representada a interactividade entre os diferentes objectos do AMM e componentes externos. Esta situação ocorre quando o gestor de mobilidade envia um pedido, *MIH-Net-HO-Commit*, para executar o *handover*. O MIHF no Android recebe a mensagem do MIHF remoto, e envia esta mensagem para o AMM através do MIH-SAP. A mensagem é recebida pelo MIH-SAP Wrapper, este verifica os parâmetros e invoca um método com os parâmetros necessários no AMM Controller. Este filtro é feito porque alguns dos parâmetros que vêm na mensagem não são necessários ou estão vazios, além de serem adaptados de C++ para Java. O AMM Controller efectua o pedido ao MIPv6 para efectuar o *handover* para a rede indicada, através da ligação TCP, onde é enviado o endereço MAC da interface de rede destino. Após o MIPv6 ter iniciado o *handover*, este envia a resposta para o AMM Controller, que também envia até o gestor de mobilidade. Esta resposta indica se o *handover* foi efectuado com sucesso.

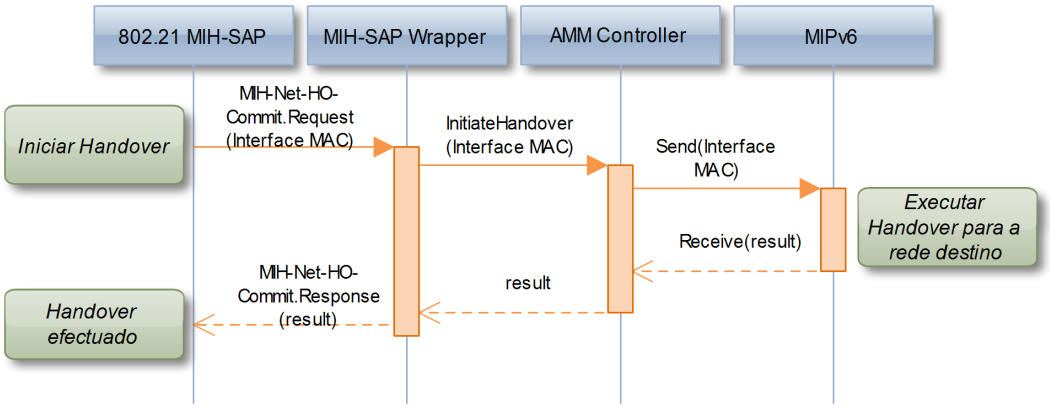


Figura 3.8: Sequencia de mensagens quando é recebido um *MIH-Net-HO-Commit*

A Figura 3.9 apresenta a troca de mensagens quando a interface gráfica invoca métodos na API remota do AMM, nomeadamente obter a interface de rede em uso. Dado que o Android Binder é um mecanismo RPC, a interface gráfica contém uma referência para a API remota do AMM, sendo no exemplo a *RemoteService*, pela qual pode invocar os métodos no AMM. O AMM Remote API recebe o pedido e envia para o AMM Controller, que faz um pedido ao MIPv6. Após receber a resposta, envia de novo para a interface gráfica. É de salientar que quando o AMM Controller recebe a resposta, isto é, o MAC da interface de rede em uso, ele devolve um parâmetro diferente para a interface gráfica. Este parâmetro é o tipo de interface, seja Wi-Fi ou 3G, dado que o utilizador não quer saber o endereço MAC mas sim o tipo de interface de rede em uso. Esta opção de usar o endereço MAC deve-se ao facto do MIPv6 reconhecer as interfaces de rede a partir do MAC e não do tipo de tecnologia.

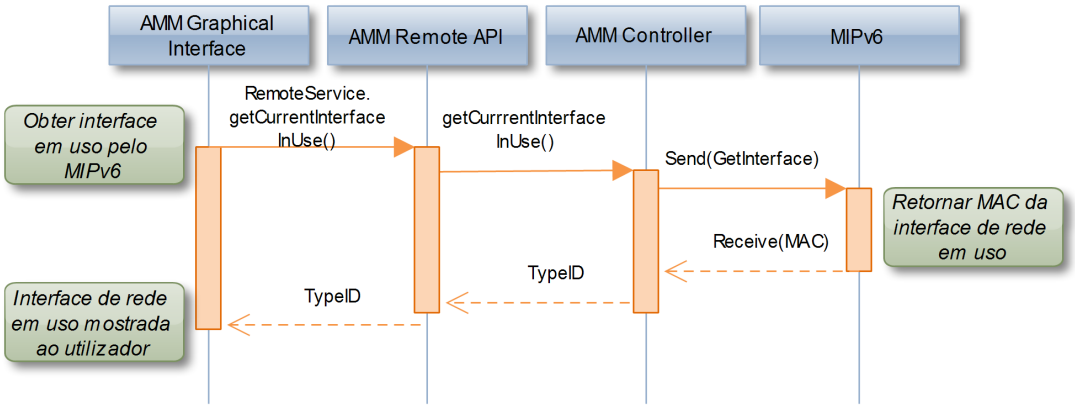


Figura 3.9: Sequência de mensagens quando é actualizada na interface gráfica o tipo de rede em uso

O parâmetro que identifica o tipo de interface de rede, o *TypeID*, é o mesmo utilizado no IEEE 802.21 para identificar o tipo de tecnologia de acesso e está de acordo com o RFC 2865 [37]. Para a tecnologia Wi-Fi (IEEE 802.11) o identificador é o 19 e para UMTS é 23.

### 3.3.2 Android Interface Manager

Este componente interage directamente com as APIs para aceder as interfaces de rede do Android. A Figura 3.10 apresenta a arquitectura deste componente, a qual é a mesma tanto para a interface de rede Wi-Fi e 3G (UMTS), respectivamente, os serviços AIM Wi-Fi e AIM 3G. No entanto, estes serviços correm independentemente em processos separados e são reconhecidos pelo MIHF e pelo gestor de mobilidade como duas entidades diferentes.

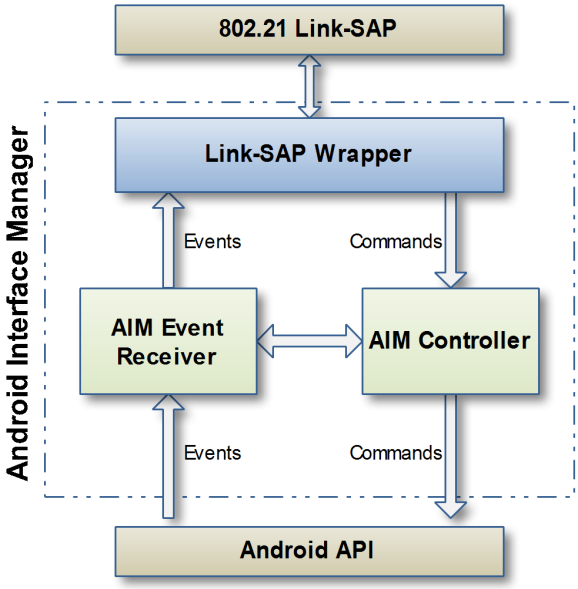


Figura 3.10: Arquitectura do Android Interface Manager

O AIM Controller é o que trata dos comandos recebidos pelo Link-SAP Wrapper e interage com as interfaces de rede através da API do Android. Os eventos são tratados pelo AIM Event Receiver e depois enviados através do Link-SAP Wrapper para o MIHF, como por exemplo, *Link Up* ou *Link Down*. No entanto, as implementações dos objectos diferem entre as duas interfaces de rede, dado que os mecanismos que tratam as mensagens (comandos e eventos) e a API do Android são diferentes. Tal como o MIH-SAP Wrapper no AMM, o Link-SAP Wrapper tem o objectivo de integrar e abstrair o Link-SAP.

### AIM Wi-Fi

Este serviço usa a API *WifiManager* disponibilizada pelo Android. A tabela 3.1 mostra os comandos e eventos suportados pelo AIM Wi-Fi.

Tabela 3.1: Comandos e eventos suportados pelo AIM Wi-Fi

| **Fases do *Handover*** | **Função** | **Primitiva** |
|---|---|---|
| **Iniciação** | Configurar o *threshold* da força do sinal | Link-Configure-Thresholds |
| | Enviar informação sobre o estado da ligação quando o *threshold* é ultrapassado | Link-Parameters-Report.indication |
| **Preparação** | Ligar interface Wi-Fi e realizar um *scan* as redes à volta do terminal | Link-Action(SCAN) |
| | Conectar-se a uma rede Wi-Fi enviada como parâmetro | Link-Action(PowerUp) |
| | Ligação estabelecida | Link-Up.indication |
| **Conclusão** | Desligar interface Wi-Fi | Link-Action(PowerDown) |
| | Ligação perdida com a rede Wi-Fi | Link-Down.indication |

A configuração do *threshold* é feita pela implementação IEEE 802.21 usada. O 802.21 Link-SAP ao receber a mensagem *Link-Configure-Thesholds*, configura-se e depois efectua um pedido ao AIM Controller para verificar se a potência de sinal ultrapassou o *threshold* configurado. Se a condição se verificar, envia um evento *Link-Parameters-Report* para o MIHF, que irá ser enviado para o gestor de mobilidade.

### AIM 3G

Este serviço é encarregue da interface 3G e usa a API *TelephonyManager* do Android. No entanto, para a implementação deste serviço foi necessário investigar o funcionamento interno do sistema, de forma a conseguir ter maior controlo sobre a interface de rede, dado que a API é muito limitada. Como abordado anteriormente, esta só fornece métodos para aceder a informações do estado das ligações e não permite alterações desses estados, ou seja, não é possível ligar ou desligar a conexão de dados UMTS/HSPA.

Após ser feita uma análise do funcionamento desta API a partir do código fonte, foi constatado que esta usa um serviço interno do Android para obter as informações. Este serviço é o *PhoneInterfaceManager* e fornece uma API muito mais completa onde se pode controlar o estado da interface de rede 3G, como ligar ou desligar a conexão de dados. O *TelephonyManager* liga-se a este serviço através de uma ligação RPC (Android Binder), onde a interface remota (API remota) disponibilizada pelo serviço está definida em AIDL e chama-

se *ITelephony*. Esta interface define os métodos Java que o serviço *PhoneInterfaceManager* disponibiliza e permite assim invocá-los remotamente.

O Android não fornece nenhuma API para poder obter uma referência para o serviço interno. No entanto, o *TelephonyManager* possui um método para obter interface remota para o serviço *PhoneInterfaceManager*, chamado *getITelephony*, mas este método é privado e não permite ser invocado por outros objectos. De modo a contornar este problema, foi usada a biblioteca Java *Reflect*, que permite modificar as permissões de acesso dos métodos e variáveis de um objecto, e assim altera-se, em *run-time*, a permissão de acesso do método *getITelephony* para público. Esta alteração é feita pelo AMM Controller e permite assim ligar ou desligar a conexão de dados através dos métodos *enableDataConnectivity* e *disableDataConnectivity*, respectivamente. A tabela 3.2 mostra os comandos e eventos suportados pelo AIM 3G.

Tabela 3.2: Comandos e eventos suportados pelo AIM 3G

| **Fases do *Handover*** | **Função** | **Primitiva** |
|---|---|---|
| **Iniciação** | Configurar o *threshold* da força do sinal | Link-Configure-Thresholds |
| | Enviar informação sobre o estado da ligação quando o *threshold* é ultrapassado | Link-Parameters-Report.indication |
| **Preparação** | Ligar interface 3G, se não estiver ligada | Link-Action(SCAN) |
| | Estabelecer conexão de dados 3G | Link-Action(PowerUp) |
| | Ligação estabelecida | Link-Up.indication |
| **Conclusão** | Desligar conexão de dados | Link-Action(PowerDown) |
| | Conexão de dados desligada | Link-Down.indication |

### Sequência de mensagens do AIM

Nesta secção em particular e a título meramente ilustrativo, pretende-se apresentar a interacção que existe entre os componentes internos do AIM quando é efectuado o processo de *handover*. Pretende-se mostrar alguma da filosofia seguida na implementação das mensagens suportadas pelo AIM no processo de *handover* de 3G para Wi-Fi. No *handover* inverso, de Wi-Fi para 3G, a interacção de mensagens é a mesma, onde só difere a API do Android usada e o uso do *scan* que não é suportado no AIM 3G. É importante salientar que o componente MIHF/Gestor de Mobilidade representa o MIHF local a correr no Android e o qual comunica com a rede, nomeadamente, o MIHF remoto e o gestor de mobilidade.

A Figura 3.11 ilustra a interacção na fase de iniciação a qual ocorre no AIM 3G, onde o 802.21 Link-SAP verifica se o potência de sinal[3] ultrapassou o valor definido pelo gestor de mobilidade. Esta verificação é feita periodicamente num intervalo de tempo definido pelo gestor de mobilidade na fase de configuração do terminal, com a mensagem *MIH-Link-Configure-Thresholds*. Se o *threshold* for ultrapassado, o gestor de mobilidade é informado com a mensagem *Link-Parameters-Report*. O valor da potência de sinal é armazenado pelo AIM Controller e é actualizado sempre que o valor se altera.

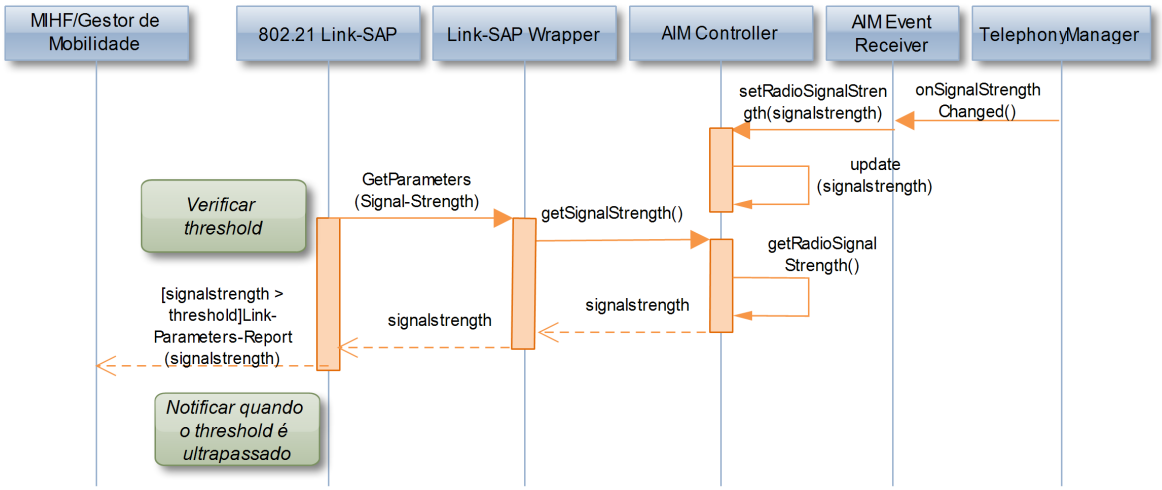


Figura 3.11: Diagrama de sequência de mensagens na fase de iniciação do *handover* no AIM 3G

Tal como abordado na secção 3.2, após ter recebido o evento a informar que o *threshold* foi ultrapassado, o gestor de mobilidade efectua um *scan* as redes Wi-Fi à volta do terminal. A Figura 3.12 mostra a comunicação quando é recebido o pedido para ligar a interface de rede Wi-Fi e efectuar o *scan*. O Link-SAP Wrapper liga a interface Wi-Fi e depois efectua a

[3]Em inglês, *signal strength*.

actualização da lista de redes à volta do terminal. Após a lista ter sido actualizada, obtém a lista e envia para o gestor de mobilidade.

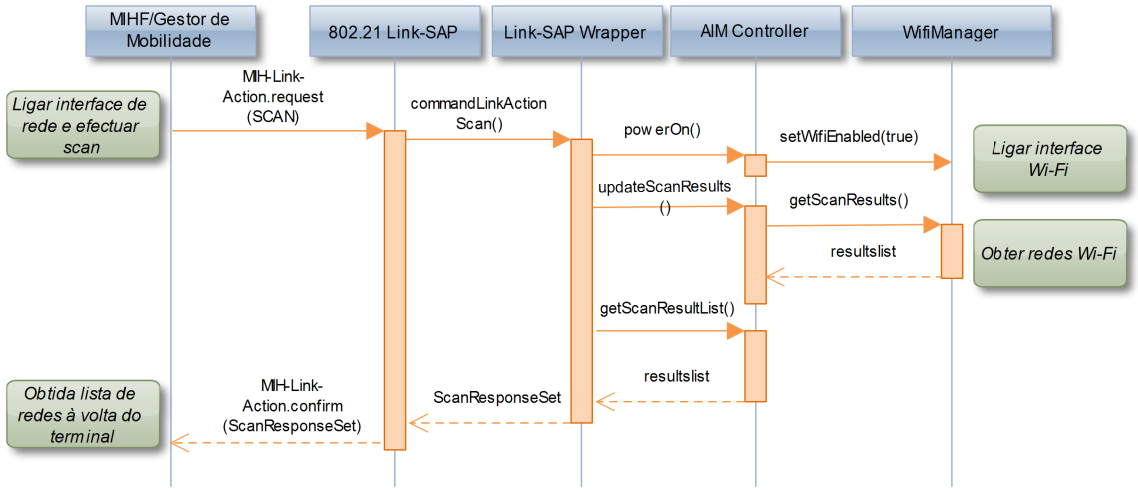


Figura 3.12: Diagrama de sequência de mensagens na fase de preparação do *handover* no AIM Wi-Fi, quando é feito o *scan* as redes à volta do terminal

Após o gestor de mobilidade ter recebido a lista de redes e ter decidido a rede destino, envia um pedido através da mensagem *MIH-Link-Action* para ligar-se à rede destino(ver Figura 3.13). O BSSID[4] da rede destino é enviado no parâmetro *PoALinkAddress* da mensagem. O AIM Controller ao receber o pedido, configura e conecta-se à rede destino. As outras redes configuradas no Android são desactivadas de modo ao terminal não mudar para outra rede. Após o terminal se conectar e associar à rede Wi-Fi, a API *WifiManager* lança um evento que é recebido pelo AIM Event Receiver e é enviado para o gestor de mobilidade com a mensagem *Link-Up.Indication*, incluindo o BSSID no novo ponto de acesso, o qual deve ser o mesmo que o anteriormente enviado escolhido pelo gestor de mobilidade. No entanto, pode haver situações em que este endereço mude, dado que ao ligar-se a uma rede (SSID[5]), esta pode ter vários pontos de acesso (vários BSSID).

A Figura 3.14 mostra a conclusão do *handover*. Nesta fase o gestor de mobilidade envia um pedido ao AIM 3G para desligar a interface 3G. Como se pode visualizar no diagrama da Figura, o AIM Controller usa o serviço interno *PhoneInterfaceManager*, através da interface remota *ITelephony*, e não a API *TelephonyManager*. Como foi abordado anteriormente, o método para ligar ou desligar a conexão de dados 3G só é possível ao usar o serviço interno *PhoneInterfaceManager*. Após ter sido desligada a conexão de dados, o Android envia um evento a notificar a alteração do estado da conexão, o qual é recebido pelo AIM Event Receiver

[4]Endereço MAC do ponto de acesso da rede Wi-Fi.
[5]Identificador da rede.

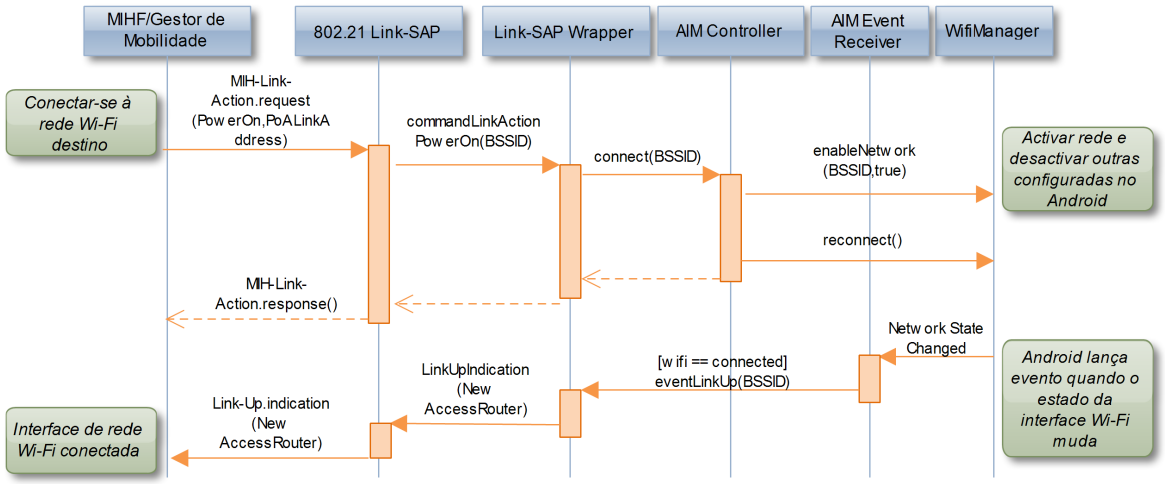


Figura 3.13: Diagrama de sequência de mensagens na fase de preparação do *handover* no AIM Wi-Fi, quando o terminal conecta-se à rede Wi-Fi destino

e enviado para o gestor de mobilidade através da mensagem *Link-Down.Indication*.

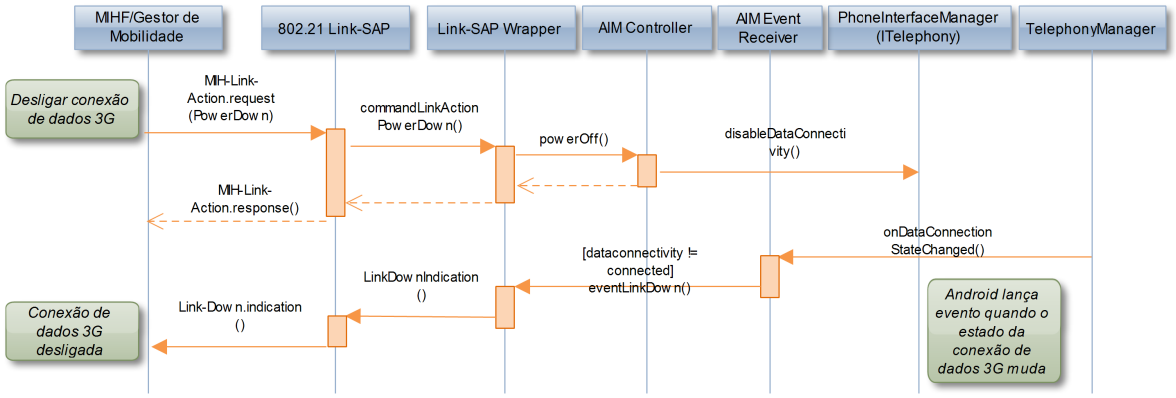


Figura 3.14: Diagrama de sequência de mensagens na fase de conclusão do *handover* no AIM 3G

### 3.3.3 Interface gráfica do AMM

Como foi abordado na secção 3.3.1, foi implementada uma interface gráfica por forma a fornecer informação ao utilizador, visualizar os serviços em execução e algumas funcionalidades implementadas para testes. A Figura 3.15(a) mostra a interface gráfica do AMM. Pode-se visualizar a interface de rede em uso actualmente e os serviços que estão a correr, nomeadamente, o MIHF, AMM, MIPv6 e AIM 3G/Wi-Fi. Por outro lado, fornece outras funcionalidade que

são usadas só para testes, tal como forçar o *handover* no gestor de mobilidade ou registar-se manualmente no gestor de mobilidade.


(a) Interface gráfica do AMM (b) Notificações do AMM

Figura 3.15: Interacção com o utilizador

A Figura 3.15(b) mostra uma notificação enviada pelo AMM. Tem como objectivo notificar o utilizador quando ocorre um *handover*, além de especificar qual foi a rede destino do *handover*.

## 3.4 Modificações ao sistema Android

Nesta secção irão ser abordadas as modificações que foram feitas ao Andriod com o objectivo de suportar os requisitos da arquitectura pretendida. Estas modificações foram feitas ao sistema Android versão 2.1. Pretende-se demonstrar o funcionamento original do mecanismo do sistema, o problema que apresenta para a solução e as alterações que foram feitas ao mecanismo.

### 3.4.1 Controlo de interfaces de rede

Dado que o controlo está nas entidades IEEE 802.21, nomeadamente, o AIM 3G e AIM Wi-Fi, através destes componentes é que o gestor de mobilidade irá ligar ou desligar as respectivas interfaces de rede. No entanto, o Android implementa um mecanismo que dá prioridade à ligação Wi-Fi. Quando a ligação Wi-Fi é estabelecida, o próprio sistema desliga automaticamente a conexão de dados 3G. O objectivo deste mecanismo é minimizar o consumo de energia e os custos para o utilizador, devido ao uso da conexão de dados 3G. Por outro lado, permite que exista sempre uma conexão activa, assim ao desligar-se da rede Wi-Fi, volta a estabelecer a conexão de dados 3G, de modo a manter a conectividade à Internet. Este mecanismo só permite uma conexão activa, onde não é possível voltar a ligar a conexão 3G enquanto estiver conectado a uma rede Wi-Fi. É possível mudar a preferência para a rede 3G, no entanto, não permite as duas ligações em simultâneo.

Este mecanismo apresenta um problema para esta solução de mobilidade heterogénea, dado que não permite o controlo por parte do gestor de mobilidade de quando ligar ou desligar uma determinada interface de rede. Por outro lado, ao desligar-se da rede Wi-Fi, só após alguns milissegundos é estabelecida a ligação de dados 3G. Isto origina perda de conectividade do terminal e não permite ao gestor de mobilidade ligar a interface 3G antes de perder a conectividade Wi-Fi, de modo a executar um *handover make-before-break*. De forma a mitigar este problema, o serviço interno do Android que controla este mecanismo, o *ConnectivityService*, foi modificado por forma a permitir ter as duas interfaces conectadas simultâneamente, ficando assim o controlo fica no gestor de mobilidade, que controla o estado das interfaces de rede através do AIM 3G e Wi-Fi.

### 3.4.2 Suporte para MIPv6

O Android não implementa nenhum mecanismo de mobilidade nem o *kernel* Linux tem as opções activadas para suportar mobilidade. A implementação do MIPv6 usada é a UMIP versão 0.4 [38], mas modificada com as funcionalidades abordadas na secção 3.1. Está implementada na linguagem C++ e suporta Linux. O suporte para MIPv6 no Android foi conseguido com duas alterações ao sistema: adicionar suporte a mobilidade no *kernel* e alterar *load/unload* do *driver* Wi-Fi.

Para adicionar o suporte à mobilidade, foram activadas opções no *kernel* necessárias para correr o MIPv6. Entre estas opções encontra-se o suporte para túneis IP, *Mobile* IP e IPv6. Depois de realizadas as alterações às opções do *kernel*, este foi recompilado e adicionado ao sistema Android com as modificações abordadas na secção anterior. O MIPv6 foi compilado com o compilador *cross-platform* para processadores ARM, os usados nos dispositivos Android.

A alteração do funcionamento do *load/unload* do *driver* Wi-Fi deve-se ao facto que no Android, quando a interface Wi-Fi é desligada, o *driver* é retirado do *kernel*, de modo a poupar bateria, dado que desliga por completo o *hardware*. No entanto, para a implementação MIPv6 usada, isto representa um problema. Ao efectuar o *unload* do *driver*, a interface de rede é desconfigurada e ao voltar a ser ligada, ao fazer o *load* do *driver*, o MIPv6 não a detecta de novo. Isto deve-se ao facto que o MIPv6 só analisa as interfaces disponíveis no arranque e não volta a verificar. De modo a modificar este funcionamento, foi analisado o serviço interno *WifiService*, o qual efectua a *load* e *unload* do *driver* quando a interface de rede é ligada e desligada, respectivamente. A alteração feita obriga então a fazer o *load* do *driver* no arranque do sistema e depois quando é ligada ou desligada a interface de rede, só é activada ou desactivada, respectivamente. No entanto, isto representa um maior consumo de energia, dado que a interface rádio nunca é realmente desligada, mantendo-se em *stand-by*.

## 3.5 Sumário

Este capítulo centrou-se na arquitectura e no modo de operação da solução de mobilidade no Android. Inicialmente foi feita uma contextualização do Android no cenário de mobilidade e o seu modo de operação, focando-se nas mensagens trocadas entre o terminal e o gestor de mobilidade. De seguida, foi feita a especificação da arquitectura da aplicação a correr no Android, mensagens suportadas e diagramas onde mostram a interacção entre os diferentes componentes. Por último, foram explicadas as modificações feitas ao sistema Android, de forma a suportar mobilidade heterogénea.

No próximo capítulo irá ser abordado o cenário de testes, as metodologias usadas e a avaliação dos resultados obtidos.

# Capítulo 4

# Testes e Resultados

De modo a poder validar a solução de mobilidade proposta, a solução para o Android foi submetida a vários testes, para obter resultados e poder avaliar a solução. O ambiente de testes usado é uma rede multi-acesso que permite ao Android executar *handovers* entre 3G e Wi-Fi. São apresentados e comentados os resultados obtidos, e são efectuadas comparações com cenários sem o suporte IEEE 801.21.

## 4.1 Ambiente de testes

Esta secção tem como objectivo abordar o ambiente de testes dos *handovers* que foi usado para obter os resultados. A topologia do ambiente de testes (*testbed*) está representada na Figura 4.1 e integra as tecnologias Wi-Fi, WiMAX e 3G (UMTS com suporte HSPA). No ponto de vista do terminal Android existem duas redes de acesso, nomeadamente, Wi-Fi e 3G. O objectivo da rede WiMAX é permitir conectividade entre a rede de acesso Wi-Fi e a rede *core*, numa ligação ponto a ponto, e permitir a reserva de recursos para os terminais ligados à rede Wi-Fi. A rede de acesso 3G é a rede comercial TMN e devido a isto existem algumas limitações que influenciam no desempenho do processo de *handover*, que irão ser explicadas posteriormente.

Os diferentes componentes da *testbed* são os seguintes:

- MM/MIPv6 HA: este computador está a correr o gestor de mobilidade (MM) e a entidade *home agent* do MIPv6, que dá suporte ao MIPv6 *mobile node* no Android. O gestor de mobilidade comunica com o WiMAX PoS e o 3G PoS para efectuar a reserva de recursos para o terminal.

- Streaming Server: neste computador correm dois serviços usados pelo terminal:

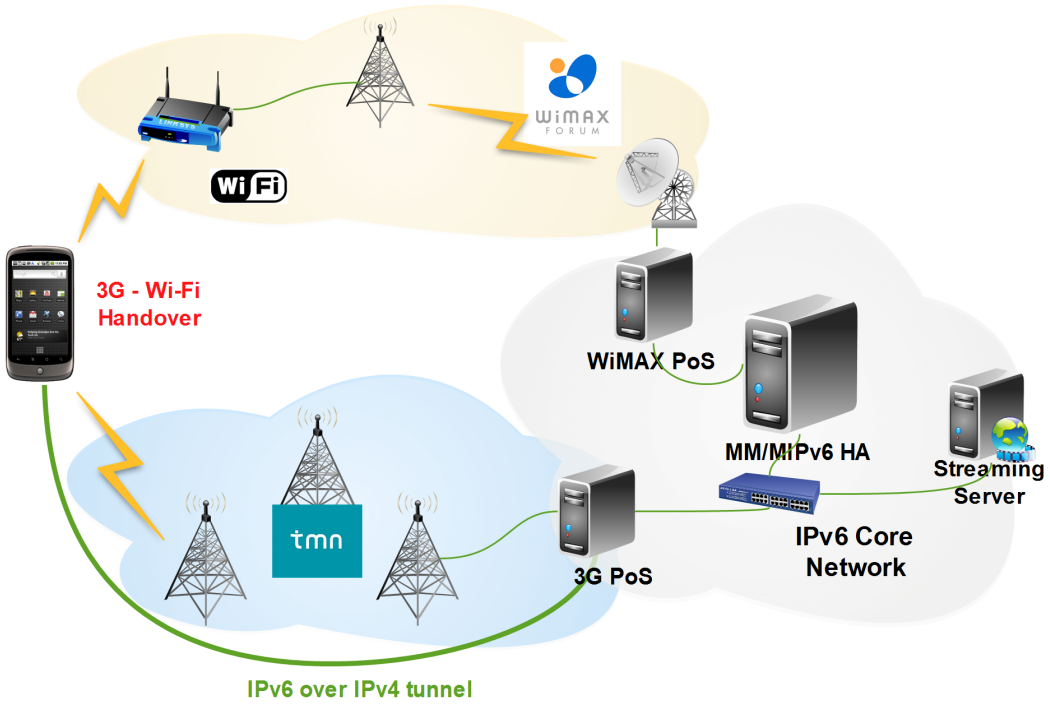


Figura 4.1: Ambiente de testes usado

  - RTSP Stream Server: servidor para *streaming* de vídeo com o protocolo RTSP sobre RTP/UDP.
  - D-ITG Traffic Generator: aplicação que permite gerar tráfego de prova e assim obter métricas QoS, nomeadamente, perda de pacotes, largura de banda, atraso dos pacotes e *jitter*.

- WiMAX PoS: permite gerir a reserva de recursos na rede WiMAX, de forma a garantir QoS ao utilizador ligado à rede Wi-Fi.
- 3G PoS: permite gerir a reserva de recursos na rede 3G, no entanto, a reserva irá ser simulada devido às limitações da rede comercial 3G.

As entidades WiMAX PoS e 3G PoS também comunicam com o gestor de mobilidade por meio do protocolo IEEE 802.21. Este protocolo define também mensagens para o gestor de mobilidade comunicar com estas entidades e poder efectuar a reserva ou liberação de recursos.

Existem limitações devido ao uso de uma rede comercial 3G (TMN). A primeira é a impossibilidade de efectuar a reserva de recursos. Dado que é uma rede comercial, esta não permite que entidades externas controlem a reserva de recursos. A segunda limitação deve-se ao facto da rede TMN só suportar IPv4 e a nossa arquitectura usar IPv6. Para ultrapassar

este problema, foi criado um túnel IPv6 sobre IPv4 entre o terminal Android e o 3G PoS, o qual permite encapsular os pacotes IPv6 em pacotes IPv4 e fornecer conectividade IPv6 ao terminal. Este túnel pode afectar muito os tempos de *handover* e as métricas QoS quando os pacotes são transportados na rede 3G, dado que as entidades envolvidas têm um maior *overhead* ao encapsular/desencapsular os pacotes IPv6 sobre IPv4. Este túnel não deve ser confundido com o túnel criado automaticamente pelo MIPv6 para transportar os pacotes entre o terminal e o *home agent* do MIPv6.

## 4.2 Metodologia

De forma a poder comparar e avaliar a solução, foram feitos testes de *handovers* com vários tipos de tráfego e obtidas várias métricas para poder avaliar diferentes aspectos. Com o objectivo de minimizar o tempo dos testes e efectuar os testes de uma forma optimizada, foi criada uma aplicação que corre no Android e permite controlar os testes e obter os valores das métricas. Esta aplicação, designada *TestApplication*, permite efectuar um teste com vários *handovers* contínuos, definidos pelo cliente, e guardar os valores obtidos em ficheiros de texto que são de fácil importação para folhas de cálculo. De modo a poder efectuar vários *handovers* seguidos para obter os valores, a aplicação comunica com o gestor de mobilidade e força os *handovers* definidos, que são efectuados intercaladamente, de 3G para Wi-Fi e Wi-Fi para 3G. Foram efectuados para cada tipo de tráfego 10 testes de *handover* em cada sentido, com duração de 6 segundos, onde o *handover* é sempre iniciado aos 2.5 segundos de cada teste.

Esta aplicação também controla o gerador de tráfego usado, que é o D-ITG [39]. Foi usado este gerador de tráfego dado que nos permite gerar vários tipos de tráfego e obter facilmente vários tipos de métricas QoS. Este gerador possui um cliente que corre no terminal, recebe o tráfego de prova e guarda os valores obtidos em ficheiros de texto. A partir do D-ITG é possível obter as seguintes métricas QoS:

- Largura de banda (*bandwitdth*).
- Pacotes perdidos (*packet loss*).
- Atraso dos pacotes (*one-way-delay*).
- Variação do atraso dos pacotes (*jitter*).

A *TestApplication* também obtém outros tipos de métricas para avaliar melhor a solução, as quais são:

- *Handover Execution Delay*: é o intervalo de tempo entre o envio da mensagem *Binding*

*Update*[1] do MIPv6 até que o primeiro pacote de dados é recebido na nova interface de rede. Este valor representa o tempo de configuração da nova ligação, isto é, estabelecimento do túnel e recepção de novos pacotes na nova interface de rede.

- *Handover Delay*: é o intervalo de tempo entre o último pacote de dados recebido na interface de rede anterior até que o primeiro pacote de dados é recebido na nova interface de rede. Este valor representa o tempo em que o terminal está sem conectividade.
- Pacotes fora de ordem (*Packets out-of-order*): número de pacotes que ainda continuam a chegar na antiga interface depois de o *handover* ter sido efectuado e o túnel ter sido estabelecido na nova interface de rede.

Estes valores são obtidos mediante a captura de pacotes nas interfaces de rede do Android. Esta aplicação é executada ao nível do *kernel* e foi desenvolvida em C/C++. Para que os valores obtidos representem o funcionamento de um cenário real, foram simulados 4 tipos de tráfego:

- Voz – VoIP a simular o *codec* G.711.2
- *Streaming* de vídeo – RTP 128kbps e 1024kbps
- Transferência de ficheiros – FTP/TCP

As características dos tipos de tráfego são apresentadas na Tabela 4.1.

Tabela 4.1: Caracteristicas dos tipos de tráfego de prova

| | ***Bitrate* (kbps)** | **Taxa de pacotes (ptks/seg)** |
|---|---|---|
| **VoIP** | 70 | 50 |
| **Vídeo 128kbps** | 128 | 16 |
| **Vídeo 1024kbps** | 1024 | 128 |
| **FTP** | 128 | 13 |

Também serão avaliados os tempos de duração de cada fase do *handover*. Estes valores são obtidos pelo gestor de mobilidade na rede e mostram o tempo de duração de cada fase do *handover*, nomeadamente, a configuração do terminal, iniciação e preparação do *handover*. A fase de execução não será avaliada porque é obtida a partir do *handover execution delay* e *handover delay* com mais detalhe. A fase de conclusão também não irá ser avaliada porque não tem relevância para o desempenho da solução, dado que o tempo de execução é desprezável em relação tempo geral do processo de *handover*.

[1]Mensagem usada para actualizar o túnel e assim redireccionar o tráfego para a nova interface de rede.

Outra funcionalidade implementada na aplicação *TestApplication* é a possibilidade de obter métricas a partir de tráfego real de *streaming* de vídeo entre o Android e o servidor de *streaming* RTSP, onde foi usado o Darwin Streaming Server [40]. A métrica QoS obtida a partir deste tráfego é a perda de pacotes, conseguida a partir da captura dos pacotes RCTP *Receiver Report* [41] (RR), enviado pelo Android, em períodos de 1 segundo, ao servidor de *streaming* de forma a informar a qualidade de recepção do vídeo. Dado que o vídeo também contem áudio, são enviados RR por cada *stream* de dados, isto é, áudio e vídeo. O campo que contém a perda de pacotes no RR é o *cumulative number of packets lost* e tem um tamanho de 24 bits. A *TestApplication* captura estes pacotes e armazena os valores num ficheiro de texto formatado. A Tabela 4.2 mostra as características do tráfego RTSP,

Tabela 4.2: Caracteristicas do tráfego RTSP (valores médios)

| | ***Bitrate* (kbps)** | **Tamanho médio dos pacotes (bytes)** | **Taxa de pacotes (ptks/seg)** |
|---|---|---|---|
| **Vídeo 128kbps*** | 163 | 1044 | 20 |
| **Vídeo 256kbps*** | 297 | 1185 | 32 |
| * Acrescentar áudio 32kbps | | | |

## 4.3 Avaliação de Resultados

Nesta secção irão ser abordados os vários testes feitos e avaliados os resultados obtidos. Com o uso da aplicação *TestApplication*, foram feitos testes com vários tipos de tráfego e obtidos vários tipos de métricas, onde em cada teste são feitos 20 *handovers*, (10 de Wi-Fi para 3G e 10 de 3G para Wi-Fi). Com o objectivo de comparar a solução de mobilidade implementada com uma solução de mobilidade onde não é usado o protocolo IEEE 802.21 como suporte, foram obtidos resultados de *handovers* feitos sem IEEE 802.21, só com MIPv6 e onde o *handover* é iniciado quando a interface em uso perde conectividade (funcionamento normal do MIPv6 sem modificações).

### 4.3.1 Mobilidade sem IEEE 802.21

Estes testes servem para perceber melhor a optimização conseguida ao usar o protocolo IEEE 802.21 e *handovers make-before-break*. Nestes testes foi usada a versão do MIPv6 sem alterações, onde o *handover* é *break-before-make*[2]. Foram obtidos os valores da perda de pacotes, *handover execution delay* e *handover delay*, com diferentes tipos de tráfego. Estes valores foram obtidos com a *TestApplication* e o gerador de tráfego (D-ITG).

[2]O *handover* só é iniciado após ter perdido conectividade na interface de rede actual.

Estes testes consistem em geração de um fluxo de tráfego durante 30 segundos, onde ocorre uma quebra de conectividade na ligação Wi-Fi e o MIPv6 efectua o *handover* para a rede 3G. Devido ao uso do túnel IPv6-sobre-IPv4 na rede 3G, só foram efectuados testes de *handovers* de Wi-Fi para 3G, dado que ao quebrar a ligação 3G, o túnel IPv6 sobre IPv4 é eliminado. Logo, para não estar a reconfigurar sistematicamente o túnel, não foram feitos testes de 3G para Wi-Fi neste cenário. A Tabela 4.3 mostra os resultados obtidos neste cenário, para cada tipo de tráfego de prova.

Tabela 4.3: Resultados obtidos nos testes sem o protocolo IEEE 802.21

| | ***Handover Execution Delay* (s)** | ***Handover Delay* (s)** | **Pacotes perdidos** |
|---|---|---|---|
| **VoIP** | 0.25 | 3.04 | 155 |
| **Video 128kbps** | 0.36 | 2.99 | 41 |
| **Video 1024kbps** | 0.30 | 3.01 | 460 |
| **FTP** | 0.34 | 3.37 | 0 |

A Tabela 4.3 mostra que o funcionamento normal do MIPv6, sem uso do IEEE 802.21 e *handovers make-before-break*, afecta muito a qualidade do serviço. Neste tipo de mobilidade, quando ocorre um *handover*, o terminal está muito tempo sem conectividade e a perda de pacotes é grande, pois o MIPv6 só reconhece que perdeu a conectividade numa interface após 2-3 segundos. Este atraso depende também da configuração da rede, dado que o MIPv6 reconhece que perdeu conectividade quando deixa de receber os *Router Advertisements* do IPv6. Dado que o tempo sem conectividade é grande, a perda de pacotes afecta aplicações VoIP ou de *streaming* de vídeo, logo não é um *handover seamless*[3]. Em relação à perda de pacotes, o tráfego FTP não é afectado, dado que este usa o protocolo TCP e este implementa mecanismos de retransmissão de pacotes, logo não ocorre perda de pacotes do ponto de vista da aplicação porque estes são retransmitidos após o terminal efectuar o *handover*.

Com estes resultados obtidos, pode-se concluir que mobilidade só com o MIPv6 não satisfaz completamente os requisitos para tráfego de aplicações em tempo real, nomeadamente, VoIP ou vídeo-conferência. No entanto, a integração com outros mecanismos, como o IEEE 802.21, permitem minimizar as perdas e optimizar o *handover*.

### 4.3.2 Mobilidade com IEEE 802.21

Neste cenário foi utilizada a solução para o Android abordada no capítulo 3, que integra o gestor de mobilidade e o MIPv6 modificado. Irão ser apresentados testes feitos com o gerador de tráfego e com a aplicação real de *streaming* de vídeo. Esta secção irá ser divida em 4 partes: resultados obtidos nas fases iniciais antes do *handover*, execução do *handover* com

[3]Em português, *handover* transparente para o utilizador.

tráfego de prova, execução do *handover* com tráfego real e resultados gerais. Assim irá ser avaliado todo o processo de *handover*, desde a configuração do terminal até a conclusão do *handover*, sem deixar de analisar bem a execução do *handover*, pois é o ponto crítico e mais importante de todo o processo de mobilidade. Os resultados foram obtidos de um conjunto de testes, cada um 20 *handovers* para cada tipo de tráfego (10 *handovers* de 3G para Wi-Fi e 10 de Wi-Fi para 3G intercalados). A partir dos valores obtidos, foi calculada a média e o desvio padrão, com um intervalo de confiança de 95%.

### Fase inicial

A Tabela 4.4 apresenta os tempos médios de cada uma das 3 fases iniciais, nomeadamente, configuração, iniciação e preparação. Na configuração são apresentados os tempos médios quando a configuração é feita pela interface 3G ou Wi-Fi, isto é, a interface em uso inicialmente pelo terminal Android quando fez o registo no gestor de mobilidade e toda a troca de mensagens de configuração foi feita por essa interface de rede. Para as fases de iniciação e preparação é apresentado o tempo médio quando para cada um dos cenários de *handover*.

Tabela 4.4: Tempo médio das fases do *handover* (ms)

| | **3G** | **Wi-Fi** |
|---|---|---|
| **Configuração** | 789 | 488 |
| | **3G - Wi-Fi** | **Wi-Fi – 3G** |
| **Iniciação** | 142 | 68 |
| **Preparação** | 279 | 95 |

Como é possível ver na Tabela 4.4, os valores da coluna esquerda são maiores que os da coluna direita. Isto deve-se ao facto de quando a configuração é feita pelo 3G ou o *handover* é de 3G para Wi-Fi, toda a troca de mensagens inicial ocorre na rede 3G. Este atraso maior ocorre porque os pacotes que transitam pela rede 3G passam por um maior número de routers e além disso são transportados dentro de um túnel IPv6 sobre IPv4, o que acrescenta maior processamento devido ao encapsulamento dos pacotes. Além disso, os pacotes que transitam na rede 3G estão sujeitos ao congestionamento da rede, o que não acontece na rede Wi-Fi, dado que é uma rede só usada no ambiente de testes. Por outro lado, na fase de preparação, quando o *handover* é de 3G para Wi-Fi, o atraso também irá ser maior dado que o terminal efectua um *scan* para obter as redes Wi-Fi à volta dele.

### Execução do *handover* com tráfego de prova

A fase de execução é a mais crítica de todo o processo de *handover*, pois têm de ser executada com impacto mínimo para as aplicações e assim não afectar a qualidade o serviço. Para cada tipo de tráfego, foram feitos testes para obter as 3 métricas mais importantes na execução

do *handover*: *handover execution delay*, *handover helay* e perda de pacotes. Nas figuras apresentadas nesta secção, as linhas que se encontram na parte de cima das barras representam o intervalo de confiança de 95%. A Figura 4.2 mostra os valores obtidos da métrica *handover execution delay*, que representa o tempo de configuração da nova ligação quando é executado o *handover*. Os valores obtidos no *handover* de Wi-Fi para 3G são maiores dado que toda a troca de mensagens de sinalização para o estabelecimento do túnel (entre o terminal e o *home agent* do MIPv6) e a recepção do primeiro pacote ocorre na interface 3G. As razões destes valores serem maiores é a mesma que foi explicada na secção anterior 4.3.2.

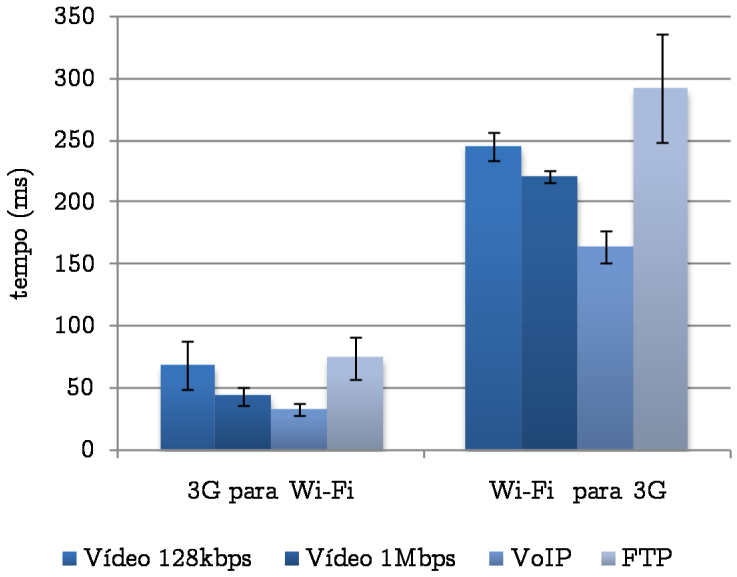


Figura 4.2: Resultados - *Handover Execution Delay*

Para o *handover delay* é esperado que seja menor que o *handover execution delay*, como é possível ver na Figura 4.3. Isto deve-se ao facto do *handover delay* depender do último pacote recebido na interface antiga e não da mensagem *binding udpate* do MIPv6, pois podem ser recebidos pacotes enquanto o túnel não é estabelecido (depois de ter enviado o *Binding Update*). A grande discrepância entre os tempos do *handover* nos diferentes sentidos é devido a que após a execução do *handover*, a interface antiga continua a receber pacotes, o que resulta em tempos menores no caso de 3G para Wi-Fi dado que os pacotes chegam mais rápido pela nova interface Wi-Fi. Por outro lado, no *handover* de Wi-Fi para 3G, a chegada do primeiro pacote a nova interface é mais demorada, pois chega pela interface 3G.

Como pode ser visualizado nas ilustrações, existem diferenças na perda de pacotes nos diferentes tráfegos. Isto deve-se a que cada tráfego tem uma taxa de pacotes por segundo diferente e logo a medida que a taxa for maior, o número de pacotes perdidos no *handover* irá

ser maior. Da mesma forma, os tempos de *handover execution delay* e *handover delay* serão menores a medida que a taxa de pacotes por segundo é maior, dado que o primeiro pacote de dados irá chegar mais rápido se a taxa de envio for grande.

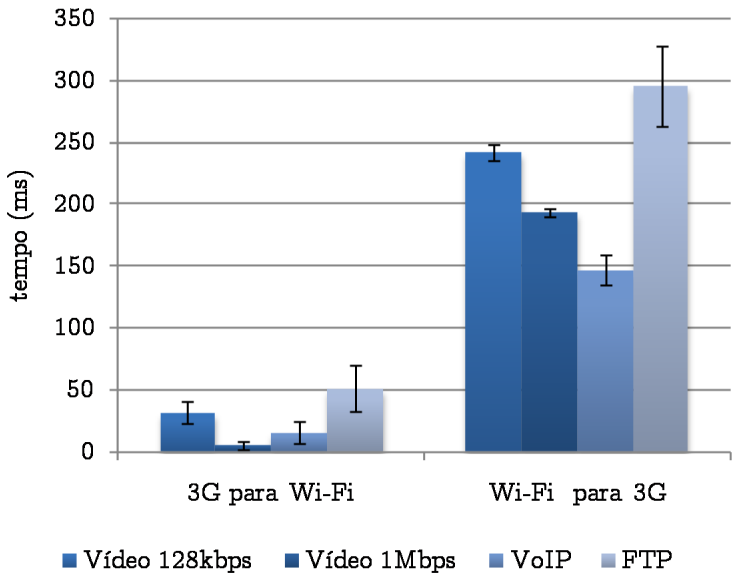


Figura 4.3: Resultados - *Handover Delay*

Na Figura 4.4 pode-se observar os pacotes perdidos para cada tipo de tráfego. Este gráfico não inclui o tráfego FTP dado que este não teve perdas. Contudo, isto não quer dizer que não houve perdas, pois devido a ser o protocolo TCP, existe retransmissão de pacotes e o cliente do gerador de tráfego não consegue detectar se houve perdas. Como pode-se visualizar na Figura 4.4, mesmo com uma abordagem *make-before-break*, existe perda de pacotes dado que é feita a reconfiguração do túnel para a nova ligação. A perda de pacotes é maior de 3G para Wi-Fi dado que alguns pacotes chegam depois da nova ligação ter sido estabelecida com sucesso (após o *handover*), sendo ignorados e contados como pacotes perdidos.

A Figura 4.4 mostra uma maior perda de pacotes de 3G para Wi-Fi. No entanto, a Figura 4.3 mostra que o *handover delay* é menor nessa situação, o que parece ser contraditório, dado que se o *handover* demorou mais tempo, a perda de pacotes deveria ser maior. Isto deve-se ao facto que, como explicado no parágrafo anterior, alguns pacotes que chegam após ter estabelecido o novo túnel são ignorados e contados como pacotes perdidos. No entanto, são contabilizados para calcular o *handover delay*, pois este valor é obtido a partir dos pacotes que chegam nas interfaces de rede Wi-Fi e 3G, e não a partir dos que chegam ao túnel do MIPv6.

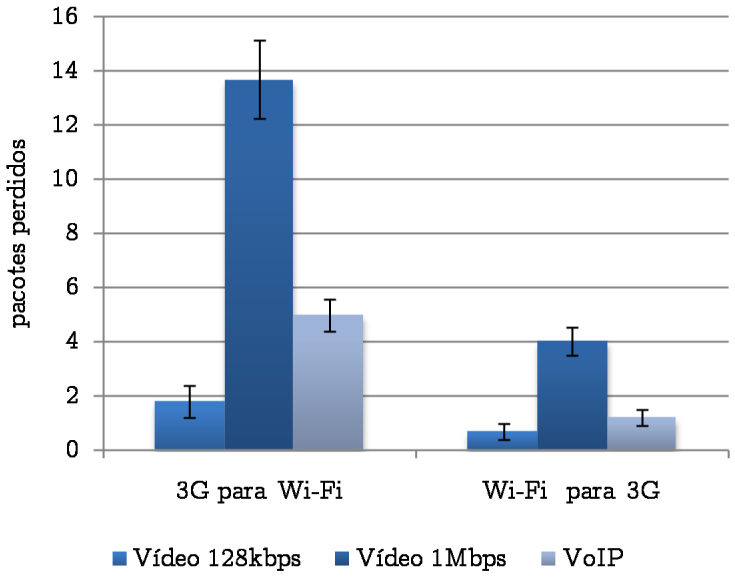


Figura 4.4: Resultados – Pacotes Perdidos

Com o objectivo de melhorar a execução do *handover* e diminuir a perda de pacotes, foram feitos testes com a optimização de rota activa no MIPv6. Em termos gerais, não houve grandes diferenças nos resultados, excepto no caso dos pacotes perdidos, pois a perda de pacotes é nula no *handover* de Wi-Fi para 3G e no caso de 3G para Wi-Fi houve uma diminuição do número de pacotes perdidos, como se pode observar na Figura 4.5. A diminuição da perda de pacotes observada deve-se ao facto de, com o processo de optimização de rota activo, os pacotes entre o terminal e o gerador de tráfego serem enviados directamente sem necessitarem de ser reencaminhados pelo o *home agent*[4], nem são afectados pela reconfiguração do túnel entre o terminal e o *home agent*. No caso do tempo de *handover*, este não varia dado que gerador de tráfego está na mesma rede que o *home agent* e não existem grandes mudanças nas rotas dos pacotes em relação ao cenário sem optimização de rota.

[4]Com optimização de rota, a triangulação não é efectuada.

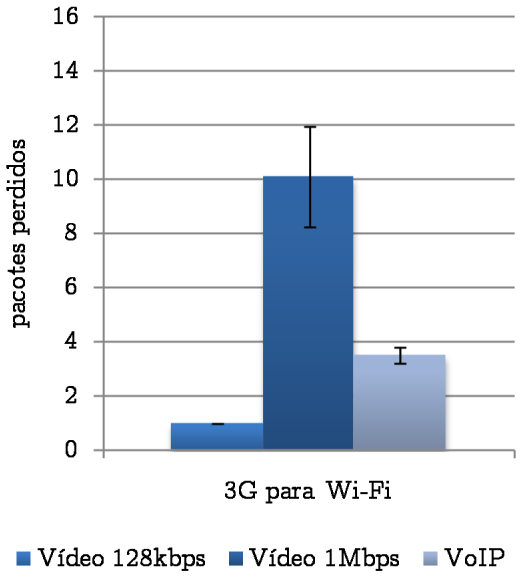


Figura 4.5: Resultados com optimização de rota no MIPv6 – Pacotes Perdidos

### Execução do *handover* com tráfego real

Nesta secção serão apresentados resultados obtidos com o tráfego real de vídeo RTSP. A métrica obtida foi a perda de pacotes a partir dos pacotes RTCP RR, como abordado na secção 4.2. Os tempos de *handover* não foram aqui avaliados porque os resultados eram basicamente iguais aos anteriores, dado que o cenário é o mesmo, onde só muda quem está a gerar o tráfego, que neste caso é uma aplicação real de *streaming* de vídeo. Não foi testado vídeos com mais do que 256kbps pois a utilização do túnel IPv6 sobre IPv4 na rede 3G afectava gravemente o vídeo, sem sequer ter feito um *handover*, e não era possível obter valores correctos sobre a perda de pacotes no *handover*. A Figura 4.6 mostra os pacotes perdidos nos testes feitos. Pode-se observar que a perda de pacotes está dentro do mesmo conjunto de valores obtidos com o tráfego de prova. Por outro lado, a perda de pacotes é superior de 3G para Wi-Fi, como já foi demonstrado e justificado com o tráfego de prova.

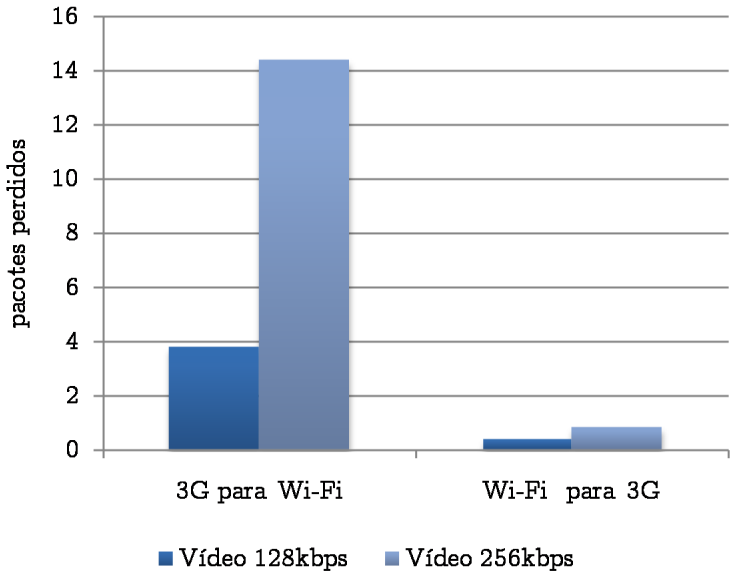


Figura 4.6: Resultados com tráfego real – Pacotes Perdidos

### Aplicação de vídeo RTSP

As figuras 4.7 e 4.8 mostram a aplicação desenvolvida que permite visualizar a *streaming* de vídeo no Android. Pode-se observar uma notificação no centro do ecrã que informa ao utilizador que ocorreu um *handover* e qual é a rede destino. Esta notificação é feita pelo Android Mobility Manager após o *handover* ter sido executado com sucesso.


Figura 4.7: *Streaming* de vídeo no Android


Figura 4.8: *Streaming* de vídeo no Android

**Resultados gerais**

Finalmente, depois de ter abordado cada uma das fases do processo de *handover*, a Tabela 4.5 mostra o tempo total médio que dura todo o processo de *handover*, incluindo todas as fases. Este tempo é calculado pelo gestor de mobilidade na rede. Como era de esperar, o tempo total é menor de Wi-Fi para 3G dado que neste sentido não é feito o *scan* das redes nem obtida as preferências do utilizador, o que leva a um tempo total menor, como já foi demonstrado na secção 4.3.2.

Tabela 4.5: Tempo médio do processo de *handover*, incluindo todas as fases (ms)

| **3G - Wi-Fi** | **Wi-Fi – 3G** |
|---|---|
| 1051 ms | 801 ms |

### 4.3.3 Comparação de Resultados

As secções 4.3.1 e 4.3.2 abordaram os resultados obtidos nos testes de *handovers* sem e com suporte IEEE 802.21, respectivamente. É de salientar que na solução sem suporte IEEE 802.21, só foram efectuados *handovers* de Wi-Fi para 3G. Pode-se concluir que existe um melhor desempenho ao usar a solução IEEE 802.21. Os valores do *handover execution delay* não foram muito diferentes entre os dois cenários, pois este valor reflecte só o tempo de estabelecimento do túnel após o *handover*, o que é esperado que seja parecido. No entanto, no *handover delay* e na perda de pacotes nota-se uma grande diferença entre os resultados obtidos. O *handover delay* é em média 10 vezes menor na solução com suporte IEEE 802.21. Da mesma forma, a perda de pacotes é cerca de 100 vezes menor. Isto representa um ganho significativo na execução do *handover* quando é usado o IEEE 802.21.

## 4.4 Sumário

Neste capitulo foram abordadas as metodologias usadas nos testes e avaliados os resultados obtidos em todo o processo de *handover*, com principal foco na execução. O desempenho dos resultados é boa, mas no entanto, precisam ser feitas melhorias para conseguir efectuar um *seamless handover* sem perda de pacotes. Como foi abordado, existiam alguns problemas no ambiente de testes que limitaram os testes feitos e afectaram os resultados obtidos. O maior problema foi o uso do túnel IPv6 sobre IPv4 na rede 3G dado que agrava o desempenho ao exigir um maior processamento das entidades por forma a encapsular/desencapsular os pacotes.

# Capítulo 5

# Conclusões

Ao longo da dissertação foi apresentado e explicado todo o trabalho feito, e é neste capítulo que irão ser abordadas as conclusões do trabalho desenvolvido. Será feita uma síntese de cada uma das etapas do trabalho, seguindo-se de uma conclusão do resultado obtido.

## 5.1 Resumo do trabalho desenvolvido

Por forma a ter uma base de conhecimento para poder implementar a solução de mobilidade no Android, foram estudadas as tecnologias de acesso *wireless*, mecanismos de mobilidade IP, o protocolo IEEE 802.21 e o sistema Android. Nas tecnologias foram abordadas principalmente as duas tecnologias que o Android irá usar, redes 3G (UMTS/HSPA) e Wi-Fi. Foram abordados os principais mecanismos de mobilidade IP e as suas características. Posteriormente, foi explicado o protocolo IEEE 802.21 e quais são os mecanismos que implementa de forma a optimizar e facilitar os *handovers* verticais. Por último, foi analisado ao pormenor o sistema Android, dado que era necessário, não só implementar uma aplicação de mobilidade heterogénea e aceder as interfaces de rede, mas também modificar o sistema de forma a suportar o MIPv6 e *handovers make-before-break*.

O desenho e implementação da solução envolveram as fases desde a definição da arquitectura de mobilidade e o seu funcionamento até a arquitectura dos componentes desenvolvidos no Android e as modificações feitas ao sistema. Na arquitectura foram definidos os componentes com os quais o Android interage, principalmente o gestor de mobilidade. Na secção de modo de operação é definida a troca de mensagens com o gestor de mobilidade em cada uma das fases do processo de *handover*, sendo a partir desta definição extraídas as mensagens IEEE 802.21 que foram implementadas. Foi especificado o software implementado no Android e como este estava interligado com o gestor de mobilidade e o MIPv6. Foram explicadas as entidades AMM e AIM, e as suas arquitecturas internas e funcionamento interno. Por úl-

timo, foram identificados algumas limitações do Android e abordadas alterações efectuadas de forma a ultrapassar as limitações.

Com o objectivo de avaliar o desempenho da solução de mobilidade no Android, foram efectuados vários testes para obter resultados e validar a solução. Foi possível concluir que a solução responde às expectativas iniciais do trabalho, ao fornecer uma mobilidade heterogénea transparente ao utilizador e optimizada com o uso do protocolo IEEE 802.21. No entanto, a qualidade do serviço é ligeiramente afectada na execução do *handover*, devido ao mecanismo de mobilidade IP usado. Este não implementa todos os mecanismos necessários para fornecer um *handover* transparente ao utilizador e aplicações. Por outro lado, algumas limitações ao ambiente de testes, já abordadas anteriormente, condicionaram os resultados obtidos.

## 5.2 Principais Contribuições

A contribuição desta dissertação é a implementação e demonstração de um cenário de mobilidade heterogénea com terminais Android. Este trabalho diferencia-se de outros trabalhos na área de mobilidade heterogénea com IEEE 802.21, pois são obtidos resultados práticos a partir de uma implementação num terminal móvel, nomeadamente, Android. A tendência actual é para o acesso à Internet seja feito principalmente através de dispositivos móveis, como os terminais Android, e que devido à grande variedade de tecnologias de acesso *wireless*, a implementação de um sistema de mobilidade heterogénea é desta forma muito vantajosa. O protocolo IEEE 802.21 é uma boa escolha para optimizar e facilitar a mobilidade heterogénea. Por outro lado, sendo o Android uma plataforma *open-source*, permite modificar e inovar o sistema com novas funcionalidades. Como fruto deste trabalho, foi publicado o artigo [7], apresentado na conferência MobiMedia 2010 em Lisboa [8]. Uma nova versão do artigo, com novos resultados publicados neste trabalho, está a ser preparado e irá ser submetido na revista *Mobile Networks and Applications* [9].

## 5.3 Trabalho Futuro

Em relação aos componentes desenvolvidos para o Android, nomeadamente, AMM e AIM, são necessários novos mecanismos para obter informação de contexto que pode ser usada na decisão do *handover*. Por exemplo, saber qual a aplicação em uso pelo utilizador permite ao gestor de mobilidade saber se é preciso mover o terminal para poder oferecer uma melhor qualidade de serviço. No entanto, também é preciso uma evolução do protocolo IEEE 802.21, dado que não foram definidas mensagens para poder transportar este tipo de dados. Por outro lado, o mecanismo de mobilidade IP utilizado não é o mecanismo ideal, pois existe uma perda de pacotes no *handover* e assim ocorrem falhas na qualidade do serviço. De forma a

melhorar a solução, é necessário usar um mecanismo de mobilidade IP que permita melhorar a qualidade do serviço, como o FMIPv6. Este mecanismo permite efectuar o *buffering* de pacotes e o envio simultâneo de pacotes para as duas interfaces de rede do terminal. Assim, diminui-se a perda de pacotes que ocorre devido ao *handover*.

Finalmente, a integração de outras tecnologias de acesso, como o WiMAX, permitirá avaliar melhor o desempenho de toda a solução. Alguns terminais Android já incorporam a interface WiMAX no lugar da interface 3G (UMTS). No entanto, espera-se que devido à rápida evolução do Android, possam vir a aparecer terminais com as três interfaces de rede, nomeadamente, UMTS, WiMAX e Wi-Fi. A integração de um terminal com estas capacidades nesta solução permitirá avaliar melhor o desempenho real do protocolo IEEE 802.21.

# Bibliografia


[1] Itu press release. http://www.itu.int/net/pressoffice/press_releases/2010/06.aspx.

[2] Internet world stats. http://www.internetworldstats.com/stats.htm.

[3] Mary Meeker, Scott Devitt, and Liang Wu. Internet trends. Technical report, Morgan Stanley Research, April 2010.

[4] Pedro Neves, Márcio Melo, Susana Sargento, Kostas Pentikousis, and Francisco Fontes. Enhanced media independent handover framework. In *Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th*, pages 1–5, Barcelona, Espanha, Abril 2009. IEEE.

[5] IEEE Computer Society. *IEEE Standard for Local and metropolitan area networks - Part 21: Media Independent Handover Services*, January 2009.

[6] Ashutosh Dutta, Subir Das, David Famolari, Yoshihiro Ohba, Kenichi Taniuchi, Toshikazu Kodama, and Henning Schulzrinne. Seamless handover across heterogeneous networks - an ieee 802.21 centric approach. In *International wireless submit-wireless personal multimedia communications*.

[7] Ricardo Silva, Paulo Carvalho, Pedro Sousa, and Pedro Neves. Heterogeneous mobility in next generation devices: An android-based case study. In Springer, editor, *Lecture Notes of ICST (LNICST)*, Setembro 2010.

[8] *MobiMedia*, September 2010. http://www.mobimedia.org.

[9] *Mobile Networks and Applications (MONET)*. Springer and ACM, 2011.

[10] Jochen Schiller. *Mobile Communications*. Pearson Education Limited, 2003.

[11] Mischa Shwartz. *Mobile Wireless Communications*. Cambridge University Press, 2005.

[12] Sotirios Maniatis, Constantin Grecas, and Iakovos Venieris. End-to-end quality of service issues over next generation mobile internet. In SCVT-2000, editor, *Symposium on Communications and Vehicular Technology*, pages 150–154, Oct 2000.

[13] Erik Dahlman, Stefan Parkvall, Johan Skold, and Per Beming. *3G Evolution, Second Edition: HSPA and LTE for Mobile Broadband*. Academic Press, 2008.

[14] IEEE Computer Society. *IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, June 2007.

[15] Wi-fi alliance. http://www.wi-fi.org/.

[16] IEEE Computer Society. *IEEE Std 802.11e - Wireless LAN medium access control (MAC) and physical layer (PHY) specifications: Medium access control (MAC) quality of service (QoS) enhancements*, November 2005.

[17] Wimax forum. http://www.wimaxforum.org/.

[18] IEEE Computer Society. *IEEE Standard for Local and metropolitan area networks - Part 16: Air Interface for Fixed Broadband Wireless Access Systems*, October 2004.

[19] Loutfi Nuaymi. *WiMAX: technology for broadband wireless access*. John Wiley Sons, 2007.

[20] IEEE Computer Society. *IEEE Standard for Local and metropolitan area networks - Part 16: Air Interface for Broadband Wireless Access Systems*, May 2009.

[21] Pedro Nuno Sousa. *Redes IP Avançadas*. Departamento de Informática - Universidade do Minho, 2009.

[22] C. Perkins. IP Mobility Support for IPv4. RFC 3344 (Proposed Standard), August 2002. Updated by RFC 4721.

[23] Charles E. Perkins. Mobile ip. *IEEE Communications Magazine*, 35(5), May 1997.

[24] D. Johnson, C. Perkins, and J. Arkko. Mobility support in ipv6. RFC 3775 (Proposed Standard), June 2004.

[25] Alvaro Gomes, Nuno Carapeto, Pedro Neves, Dimitris Komnakos, and Lambros Sarakis. *D2.1: Handover reference scenarios, requirements specification and performance metrics*. HURRICANE - Handovers for Ubiquitous and optimal bRoadband connectIvity among CooperAtive Networking Environments, 2008.

[26] R. Koodli. Mobile IPv6 Fast Handovers. RFC 5568 (Proposed Standard), July 2009.

[27] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, and B. Patil. Proxy Mobile IPv6. RFC 5213 (Proposed Standard), August 2008.

[28] Kenichi Taniuchi, Yoshihiro Ohba, Victor Fajardo, Subir Das, Miriam Tauil, Yuu-Heng Cheng, Ashutosh Dutta, Donald Baker, Maya Yajnik, and David Famolari. Ieee 802.21: Media independent handover: Features, applicability, and realization. *IEEE Communications Magazine*, pages 112–120, January 2009.

[29] Adriano Tavares da Silva Pinho. Implementation of the ieee 802.21 in the network simulator 3. Master's thesis, Faculdade de Engenharia da Universidade do Porto, June 2008.

[30] Rick Rogers, John Lombardo, Zigurd Mednieks, and Blake Meike. *Android Application Development*. O'Reilly, 1 edition, May 2009.

[31] Apache license, version 2.0, January 2004. http://www.apache.org/licenses/LICENSE-2.0.

[32] Google i/o 2008 - anatomy & physiology of an android. http://sites.google.com/site/io/anatomy–physiology-of-an-android.

[33] Android developer guide. http://developer.android.com/guide.

[34] Jouni Malinen. Wpa supplicant, 2009. http://hostap.epitest.fi/wpa_supplicant.

[35] Android open source project - porting guide. http://source.android.com/porting/telephony.html.

[36] Android open source project - source code. http://android.git.kernel.org.

[37] C. Rigney, S. Willens, A. Rubens, and W. Simpson. Remote Authentication Dial In User Service (RADIUS). RFC 2865 (Draft Standard), June 2000. Updated by RFCs 2868, 3575, 5080.

[38] Umip: Usagi-patched mobile ipv6 for linux. http://umip.linux-ipv6.org/.

[39] Distributed internet traffic generator (d-itg). http://www.grid.unina.it/software/ITG/.

[40] Darwin streaming server. http://dss.macosforge.org/.

[41] Audio-Video Transport Working Group, H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. RFC 1889 (Proposed Standard), January 1996. Obsoleted by RFC 3550.