Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática

2010

Flávio da Silva Fonseca

Cliente IPTV Multi-Plataforma com Personalização Automática de Canais

Multiplatform IPTV Client with Automatic Channels Personalization

Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática

2010

Flávio da Silva Fonseca

Cliente IPTV Multi-Plataforma com Personalização Automática de Canais

Multiplatform IPTV Client with Automatic Channels Personalization

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor Paulo Salvador e do Doutor António Nogueira, Professores Auxiliares do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro.

o júri

presidente

Prof. António Luís Jesus Teixeira
Professor Associado da Universidade de Aveiro

Prof. António Manuel Duarte Nogueira
Professor Auxiliar da Universidade de Aveiro

Prof. Paulo Jorge Salvador Serra Ferreira
Professor Auxiliar da Universidade de Aveiro

Prof. Joel José Puga Coelho Rodrigues
Professor Auxiliar da Universidade da Beira Interior

agradecimentos

Dedico esta dissertação à minha família pelo apoio durante todos estes seis anos de estudo na Universidade de Aveiro e aos meus amigos que me ajudaram quando eu precisei.

palavras-chave

IPTV, Televisão Digital, Personalização, Personalização Automática, Aprendizagem, Serviços, Universalidade

resumo

Os serviços de distribuição de conteúdos multimédia via Internet estão a crescer e a evoluir exponencialmente. Os serviços que se propõem entregar os conteúdos multimédia mais adequados às preferências do cliente necessitam de possuir a capacidade de aprender os perfis dos utilizadores em múltiplas vertentes. Os perfis dos utilizadores necessitam de ser classificados a diversos níveis: pessoais, contextuais e tecnológicos. Perante esta classificação multi-dimensional dos clientes, o serviço irá compor de forma automática canais de televisão personalizados ao cliente e ao contexto em que este está inserido nos diversos momentos. Assim, os terminais do cliente deverão, para além das capacidades de visualização dos conteúdos multimédia, permitir a interacção explícita do utilizador com o serviço mas também toda a interacção implícita que fornecerá informação contextual ao servidor.

Esta dissertação insere-se num trabalho mais amplo de criação de um serviço de IPTV com personalização automática de canais e classificação de conteúdos. Em paralelo com o desenvolvimento de um servidor de IPTV, existiu necessidade de criar um cliente móvel multi-plataforma. Este cliente irá permitir ao utilizador a reprodução dos conteúdos multimédia e a iteração (explícita e implícita) com o servidor. Esta dissertação apresenta a metodologia e processo de criação de um cliente IPTV que possa ser executado em múltiplas plataformas e em diversos tipos de dispositivos.

keywords

IPTV, Digital Television, Customization, Automatic Customization, Learning, Universality

abstract

Internet services that provide the distribution of multimedia contents are growing exponentially and evolving in a constant way. Services that intend to deliver the multimedia contents that are more appropriate to the client preferences need to have the ability to learn the user profiles on multiple contexts. User profiles need to be learned an classified at different levels: personal, contextual and technological. Given this multi-dimensional classification of customers, the service will automatically compose television channels that are customized to the client and to the context where it is inserted at different moments in time. Thus, the client terminals should, in addition to the visualization capabilities of multimedia contents, allow the explicit interaction with the service, but also provide all implicit interactions that provide contextual information to the server. This work is part of a larger developing project that aims to create an IPTV service with automatic channel personalization and contents rating. In parallel with the development of an IPTV server, it was necessary to create a multi-platform mobile client able to fully interact with it. This client will allow users to visualize multimedia contents and interact (explicitly and implicitly) with the server. This dissertation presents the methodology and process of creating an IPTV client that can run on multiple platforms and in different types of devices.

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

API	*Application Programming Interface*
BD	Base de Dados
CDC	*Connected Device Configuration*
CLDC	*Connected Limited Device Configuration*
GUI	*Graphical user interface*
IDE	*Integrated Development Environment*
IPTV	*Internet Protocol Television*
Jarpa	*Java Packaging for Flash Lite Developers*
Java ME	*Java Platform, Micro Edition*
JRE	*Java Runtime Environment*
JSR 75	*FileConnection APIs*
JSR 135	*Mobile Media API*
JSR 177	*Security and Trust Services API*
JSR 179	*Location API*
JSR 226	*Scalable 2D Vector Graphics API*

JVM	*Java Virtual Machine*
MIDP	*Mobile Information Device Profile*
MMAPI	*Mobile Media API; JSR 135*
P2P	*Peer-to-Peer*
Pad	*Tablet Computer*
RAM	*Random Access Memory*
RIA	*Rich Internet application*
RSS	*Really Simple Syndication*
RTOS	*Real-time operating system*
SDK	*Software Development Kit*
SGBD	Sistema Gestor de Base de Dados
SO	Sistema Operativo
SOAP	*Simple Object Access Protocol*
SVG	*Scalable Vectorial Graphics*
TV	Televisão (*Television*)
UML	*Unified Modeling Language*
WRT	*Web Runtime*
XML	*eXtensible Markup Language*

Glossário

BitTorrent É um protocolo de partilha de dados que utiliza uma arquitectura *Peer-to-Peer* (P2P) e é usado para distribuição de grandes quantidades de dados por um elevado número de utilizadores. *BitTorrent* é um dos protocolos actualmente mais comuns em todo o mundo para transferência de arquivos de grande dimensão. O protocolo *BitTorrent* permite aos seus utilizadores distribuir grandes quantidades de dados sem sobrecarregar nenhum sistema central de armazenamento. Um utilizador, ao mesmo tempo que obtém determinado ficheiro, partilha os "pedaços" que já possui deste e de outros que já possua e deseje disponibilizar na rede. Deste modo, uma determinada partilha não se encontra armazenada num só ponto crítico da rede mas distribuída por múltiplos utilizadores da rede de forma indiscriminada. Com esta abordagem é possível aumentar a disponibilidade dos dados, a sua acessibilidade, a tolerância a falhas, a largura de banda dos *downloads* possibilitando a transferência simultânea a partir de várias fontes e ainda a robustez e escalabilidade da rede;

Kernel O *Kernel* de um sistema operativo é entendido como o núcleo deste ou, numa tradução literal, o seu cerne. Este representa a camada de software mais próxima do *hardware*, sendo responsável por gerir os recursos do sistema computacional como um todo;

Multicast Em *multicast* a entrega de informação (pacotes) não é feita de forma singular mas sim relativa a um grupo de destinatários que recebem os dados de forma "simultânea". Esta arquitectura usa uma estratégia muito mais eficiente de entrega de informação quando esta é igual para múltiplos destinatários (por exemplo, *streaming*

de um mesmo vídeo em tempo-real), ocupando a ligação de transmissão com um único fluxo de pacotes, criando cópias do fluxo somente quando as ligações para os destinatários se separam. Do ponto de vista da associação subjacente ao endereço de rede, esta é do tipo um-para-muitos entre o endereço de rede e os pontos de entrega finais;

Streaming Fluxo de dados multimédia distribuídos aos seus clientes através de uma rede de telecomunicações. Este conceito é frequentemente utilizado para distribuir conteúdos multimédia através da Internet. A utilização de *streaming* possibilita que o utilizador aceda ao conteúdo multimédia de uma forma rápida e faseada em que não necessita de transferir todo o conteúdo para começar a assistir ao mesmo. Tal como o nome indica, trata–se de um fluxo de dados que é usado para apresentar o conteúdo no momento ao utilizador sendo depois os dados, tipicamente, perdidos. A informação pode ser transmitida utilizando diversas arquitecturas, através de *unicast*, *multicast* ou *broadcast*.

Capítulo 1

Introdução

Quando pretendemos assistir a um conteúdo multimédia do nosso interesse na Televisão (*Television*) (TV), provavelmente necessitamos de fazer *Zapping* entre diversos canais, ou então consultar a programação dos canais, de modo a obter o horário dos programas pretendidos. Por vezes, perdemos parte do programa devido ao facto de este se ter iniciado antes de seleccionarmos o respectivo canal. Para eliminar estas situações, foi criado um servidor de *Internet Protocol Television* (IPTV) com personalização automática de conteúdos de modo a permitir ao utilizador assistir aos seus programas preferidos quando tiver condições para o fazer, não necessitando assim de se preocupar com os horários dos mesmos. Graças ao facto do sistema IPTV ter capacidade de aprender as preferências dos utilizador, é possível sugerir conteúdos do seu interesse.

1.1 Objectivos

A criação do servidor de IPTV, suscitou a necessidade de criar um cliente para o mesmo. Este cliente servirá como a interface que o utilizador vai utilizar para reproduzir os conteúdos sugeridos e fazer a interacção com o servidor. O objectivo desta dissertação foi criar um cliente com capacidade de ser executado em diversas plataformas e em dispositivos com diversos tipos de características de *hardware*. Outro dos objectivos é permitir que a aplicação possa ser executada em dispositivos móveis, o que obrigou a ter em conta as

limitações físicas destes dispositivos.

1.2 Estrutura

Esta dissertação é constituída por 5 capítulos. De seguida é apresentada uma breve descrição do conteúdo de cada capítulo.

Capítulo 1 Neste capítulo encontra-se uma introdução à dissertação e uma descrição do conteúdo da mesma. São também mencionados os objectivos principais do projecto;

Capítulo 2 Neste capítulo é efectuado um enquadramento das tecnologias utilizadas ou consideradas e também das características do servidor IPTV;

Capítulo 3 Neste capítulo encontra-se descrito todo o trabalho realizado e as características das aplicações desenvolvidas;

Capítulo 4 Neste capítulo são efectuados alguns testes à aplicação desenvolvida em Qt;

Capítulo 5 Neste capítulom que constitui a parte final deste trabalho, são apresentados alguns tópicos de trabalho futuro e as principais conclusões.

Capítulo 2

Enquadramento

2.1 Servidor IPTV com Personalização Automática de Canais

O cliente IPTV a desenvolver tem como objectivo interagir com um servidor, sendo este o responsável por disponibilizar os *links* para os conteúdos multimédia. Nesta secção irá ser efectuada uma descrição do servidor em questão.

O servidor IPTV[33], para interacção com o cliente, disponibiliza uma série de *web services Simple Object Access Protocol* (SOAP) que retornam uma *String* com informações no formato *eXtensible Markup Language* (XML). Esta resposta tanto pode ser um *link* de um conteúdo multimédia com as suas respectivas informações, uma mensagem de erro, ou a confirmação de uma acção executada com sucesso. Na tabela 2.1 é possível encontrar uma lista dos *web services* disponibilizados e as suas respectivas funções.

Relativamente à estrutura funcional do servidor, esta é constituída por quatro módulos distintos que podem ser distribuídos por diversas máquinas: o *IPTV Server Core*, que é responsável por fazer a interacção com o cliente; o *Content Link Sources*, cujo objectivo é a gestão dos conteúdos multimédia; o *User Profile Learning*, que tem como função sugerir os conteúdos aos utilizadores e aprender as suas preferências e, por fim, a *IPTV Database*, que consiste na gestão da base de dados onde estão armazenadas as informações dos conteúdos multimédia e dos utilizadores. Uma descrição mais pormenorizada dos respectivos módulos

e das suas funcionalidades será efectuada posteriormente.

Nome do *web service*	**Descrição da função**
User Authentication	Permite que um utilizador se autentique de modo a iniciar uma nova sessão no servidor
User Deauthentication	Permite que um utilizador termine a sua sessão
New User	Permite um utilizador criar uma nova conta
New User Validation	Permite que um utilizador previamente registado valide o seu registo
New Link Request	Permite ao cliente pedir o próximo *link* de um conteúdo sugerido e em simultâneo classificar o conteúdo previamente assistido
Link List Request	Possibilita que um cliente obtenha uma lista de vários *links* para conteúdos, segundo alguns critérios de procura e selecção
Link Feedback Request	Permite que um cliente envie o *feedback* de um conteúdo
keep Alive Session Request	Permite que um cliente mantenha aberta sessão com o servidor, dado que esta termina automaticamente após algum tempo de inactividade
Command Request	Permite que um cliente envie um pedido de comando ou operação que não exija autenticação

Tabela 2.1: Diferentes tipos de serviços disponibilizados

2.1.1 *IPTV Server Core*

Nesta secção vai ser descrito com mais pormenor o módulo *IPTV Server Core* cuja principal função é a interacção cliente-servidor.

Este módulo segue uma política de comunicação cliente-servidor típica, onde o servidor tem uma acção passiva devido ao facto de ser sempre o cliente a tomar a iniciativa relativamente ao envio dos pedidos.

Uma das funcionalidades deste módulo é enviar, em determinadas ocasiões, mensagens electrónicas de *e-mail* para os utilizadores.

Algumas das situações que causam/provocam o envio de (*e-mails*) para os utilizadores

são:

- Criação de uma nova conta de utilizador (conta ainda por validar);
- *Feedback* relativo a uma validação com sucesso da nova conta (conta validada);
- Excesso de tentativas de *login* falhadas;
- *Broadcast* de informações/notícias gerais sobre o sistema, enviadas para um grupo de utilizadores. Possibilidade de criação de *mailing lists* segundo diferentes critérios relativos às características dos utilizadores.

Vamos passar a uma descrição simples do que ocorre quando efectuamos cada uma das acções disponibilizadas pelo servidor, tendo estas acções já sido referidas na tabela 2.1.

User Authentication A autenticação é efectuada por comparação das credenciais definidas no momento da validação da conta com as fornecidas na tentativa de *login*. Estas têm de coincidir de forma exacta. Caso o utilizador falhe a autenticação três vezes, fica impossibilitado de aceder ao sistema durante um curto período de tempo, sendo notificado via *e-mail* do ocorrido;

User Deauthentication A finalização da sessão no sistema pode ser manual ou automática, dependendo se é o utilizador a efectuá-la voluntariamente ou se é o próprio sistema a despoletá-la. Caso o utilizador efectue *login* no servidor utilizando outro sistema computacional, a sessão anterior é encerrada e a nova é naturalmente iniciada. Caso a sessão esteja inactiva durante um longo período de tempo, este módulo efectua também o fecho automático da sessão;

New User Esta tarefa permite criar um novo utilizador no sistema, bem como iniciar a construção do seu perfil de preferências multimédia. Caso as credenciais e/ou os dados relativos ao perfil já existam no sistema, é devolvida uma resposta que indica que a criação do registo falhou, sendo também incluída uma descrição dos motivos da falha. Um novo utilizador do sistema necessita sempre de validar a sua nova conta.

Para isso, o servidor envia um *e-mail* para o utilizador com o *user name* criado e uma *password* que deverá alterar ou confirmar;

New User Validation Depois de criada uma nova conta no servidor, é necessário a validação da mesma. Este requisito é executado por uma chamada ao *web service New User Validation*. Após a conta ser validada, é enviado um *e-mail* para o utilizador a notificar que a validação foi efectuada com sucesso;

Session Keep Alive Request A chamada deste *web service* leva o servidor a actualizar a data e hora da última actividade do utilizador no sistema, de modo a evitar o *logout* automático do utilizador;

New Link Request Relativamente à resposta da chamada deste *web service*, os *links* dos conteúdos multimédia são enviados ao utilizador por ordem decrescente da data de sugestão. Estas sugestões continuam presentes na respectiva tabela da Base de Dados (BD), até ser recebido pelo servidor algum *feedback* do utilizador respectivo relativamente ao conteúdo multimédia assistido. O cliente, na chamada deste *web service*, pode optar por enviar também o *feedback* de um conteúdo multimédia;

Link List Request Ao contrário da chamada do *web service New Link Request*, uma chamada do *web service Link List Request* permite obter uma lista de tamanho variável de *links* de conteúdos multimédia e não somente um *link* para um conteúdo multimédia. Para além disso, é possível especificar critérios de procura, critérios de ordenação e a origem do conteúdo multimédia em termos de organização interna do servidor. Por outras palavras, o conteúdo multimédia ou pertence à lista de conteúdos sugeridos ao utilizador respectivo ou então pode ser qualquer conteúdo multimédia presente na BD;

Link Feedback Request Dado que o sistema pretende aprender dinamicamente o perfil do utilizador, este deverá especificar sempre que possível um *feedback* relativo ao conteúdo multimédia assistido. Esta tarefa actualiza a classificação global do

conteúdo, assim como insere uma nova entrada no histórico de *feedbacks* do utilizador respectivo, para posterior tratamento pelo módulo *User Profile Learning* (secção 2.1.3);

Command Request A tarefa realizada com recurso à chamada deste *web service* pode ser muito variada, desde um comando de verificação de disponibilidade do servidor (*Ping Request*) até a um pedido de uma configuração específica ou execução de uma tarefa que não necessite da autenticação no sistema.

2.1.2 *Content Link Sources*

Nesta secção irá ser efectuada uma descrição das funcionalidades do módulo *Content Link Sources*.

O objectivo deste módulo prende-se com a ligação entre os conteúdos multimédia externos e o servidor IPTV. Para a obtenção dos dados e *links* dos conteúdos multimédia de forma automática, utilizam-se os *Really Simple Syndication* (RSS) ou *Application Programming Interface*s (APIs) dos provedores de conteúdos. É de notar que o sistema não armazena os conteúdos multimédia propriamente ditos, armazenando apenas um *link* para o conteúdo multimédia.

Outro dos objectivos deste módulo é a gestão dos conteúdos inseridos, ou seja, actualizar a sua informação, eliminando-os caso estejam desactivados.

Este módulo permite também a inserção de conteúdos multimédia manualmente e a gestão dos mesmos.

2.1.3 *User Profile Learning*

Nesta secção irão ser descritas as funções do módulo *User Profile Learning*.

Este módulo possui duas funções: a sugestão de conteúdos e a aprendizagem dinâmica do perfil do Utilizador. A sugestão de conteúdos, como o nome indica, é a sugestão dos conteúdos multimédia aos utilizadores que segundo o seu perfil estejam interessados nos

mesmos. A aprendizagem dinâmica do perfil corresponde à actualização das características e preferências do utilizador em questão tendo em conta o feedback enviado pelo utilizador.

Este módulo, devido à sua complexidade, ainda se encontra com uma implementação muito simples.

2.1.4 *IPTV Database*

Em relação a esta secção irá ser feita uma descrição do módulo *IPTV Database*.

Este módulo é um conjunto formado pela base de dados (Sistema Gestor de Base de Dados (SGBD)) e a API de acesso à base de dados. Esta API é utilizada em todos os módulos que necessitem de aceder à base de dados. Como todos os módulos vão ter acesso à base de dados devido ao facto de ser nesta que são armazenadas todas as informações relativas aos conteúdos multimédia e aos utilizadores, é necessário utilizar esta API de modo a manter a persistência na base de dados. Esta API, além de complementar as entidades e métodos de introdução, acesso e manipulação dos dados armazenados na base de dados, também possui métodos de inicialização da base de dados de modo a reduzir o tempo de instalação do sistema.

2.2 Sistemas Operativos/Plataformas Móveis

A aplicação desenvolvida em Qt pode ser executada em Sistema Operativos (SOs) onde é possível executar o SMPlayer, o MPlayer e aplicações que utilizem gSOAP. Actualmente os SOs que suportam estes requisitos são diversos, sendo os mais importantes o Windows e o Linux. Nos sistemas baseados en Linux é de destacar o Maemo e o MeeGo. Mas além destas plataformas também existem projectos para portar Qt para plataformas que já permitem executar o MPlayer, por exemplo o Android. Sendo assim, possivelmente esta aplicação também poderá no futuro ser executada nessas plataformas.

2.2.1 Maemo

Esta plataforma de software é uma plataforma desenvolvida pela *Nokia* para *smartphones* e *Internet Tablets* sendo baseada numa distribuição de *Linux Debian*.

O Maemo[6] é na sua maior parte baseada em código *open source* sendo desenvolvida em colaboração com diversos projectos *open source*, tais como *Linux Kernel*, *Debian* e *GNOME*. A maioria das suas GUI *frameworks* e livrarias têm origens no projecto *GNOME*.

A sua última versão estável foi lançada a 25 de Maio de 2010 e é a versão 5.0 PR1.2.

Esta plataforma permite desenvolver aplicações utilizando diversas linguagens de programação, entre elas *C/C++*, *Java* usando o *Jalimo JVM*, *Python*, *Ruby* e *Mono*.

Na figura 2.1 encontra-se a GUI *home* do *Maemo 5*.

Figura 2.1: GUI *home* do *Maemo 5*.

.

2.2.2 Symbian

O *SymbianOS*[9] foi originalmente desenvolvido pela *Symbian Software Limited* que em 2008 foi adquirida pela *Nokia* e uma organização independente chamada *Symbian Foundation*. Em Fevereiro de 2010, o código fonte desta plataforma passou a código aberto.[32] Actualmente esta é a plataforma para dispositivos móveis mais utilizada a nível mundial,

com 41.2 por cento[29] do mercado dos *smartphones*.

A versão mais recente desta plataforma é a *Symbianˆ3*. Mas a plataforma considerada para este projecto foi a *S60* devido ao facto de ser a plataforma do *Symbian* mais utilizada e a mais recente na altura do início do desenvolvimento.

Esta plataforma permite o desenvolvimento em diversas linguagens de programação, como *C/C++, Java Me, Python, Flash Lite, Ruby, .NET, Web Runtime (WRT)*.

Na figura 2.2 encontra-se a GUI *home* do *Symbian S60 5th edition*.

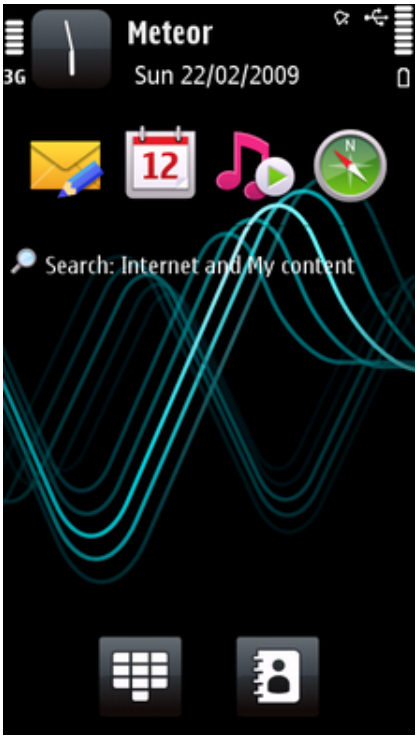

Figura 2.2: GUI *home* do *Symbian S60 5th edition*.

2.2.3 MeeGo

O projecto *MeeGo*[16][31] é um projecto *open source* baseado em *Linux* que foi anunciado em Fevereiro de 2010, sendo fruto da união de esforços entre o SO *Moblin*[18] da *Intel* e o SO *Maemo* da *Nokia*. Mais recentemente, em Novembro de 2010,[34] a *AMD* também se juntou ao projecto. O primeiro lançamento do *MeeGo* foi a 26 de Maio de 2010.

Este SO está projectado para ser utilizado em diversos tipos de dispositivos, entre os quais *Netbooks* (figura 2.3), *Handsets* (figura 2.4), *Tablets* (figura 2.5), *In-Vehicle* figura (2.6), *Connected TVs* e *Media phones*. A principal linguagem de desenvolvimento para este SO é *QT/C++*, embora também suporte *GTK*.

Os primeiros dispositivos móveis que irão utilizar o MeeGo têm lançamento projectado para a primeira metade de 2011.[35] Apesar disto, já estão disponíveis este ano alguns *netbooks*, *Connected TVs* e um *Tablet Computer* (Pad) o *WeTab*[27] com esta plataforma.

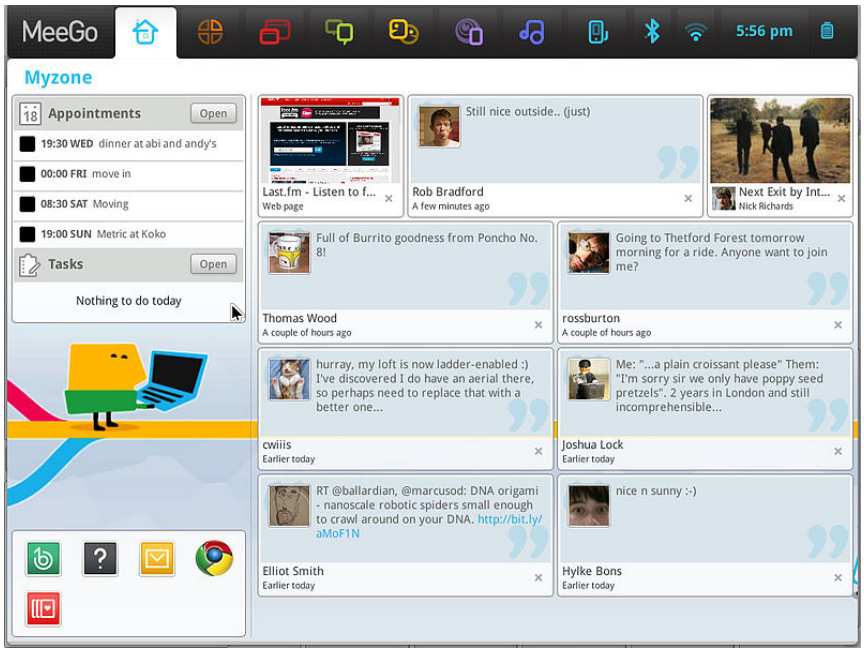

Figura 2.3: GUI *home* do *MeeGo Netbook*.

2.2.4 Android

O Android[1] é um SO baseado numa versão alterada do *Linux Kernel* sendo um participante na *Open Handset Alliance*[8]. Foi inicialmente desenvolvido pela *Android Inc*, uma firma que foi comprada pela *Google* em 2005. Este SO possui uma grande comunidade de programadores, que programam em Java controlando o dispositivo usando livrarias desenvolvidas pela *Google*[10].

Este SO pode ser utilizado em diversos tipos de dispositivos, entre os quais telemóveis,

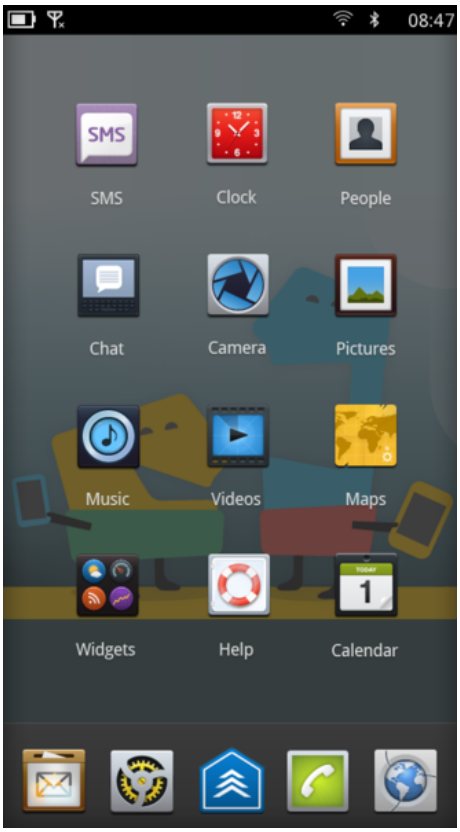

Figura 2.4: GUI *home* do *MeeGo Handset*.

netbooks, *Connected TVs* e Pads. O primeiro dispositivo móvel a ser lançado com o Android foi o HTC Dream, lançado a 22 Outubro de 2008, e a primeira televisão a ser lançada com Android é a Scandinavia[20].

A primeira versão deste SO foi lançada a 21 de Outubro de 2008 e a última versão estável é a versão 2.2 (Froyo). Na figura 2.7 encontra-se a GUI *home* do *Android 2.2*.

2.2.5 iOS

O iOS[2] é o SO para dispositivos móveis da Apple. Este SO foi inicialmente desenvolvido para o *iPhone* mas também é utilizado no *iPod Touch*, no *iPad* e na *Apple TV*. Este SO é derivado do *Mac OS X*, sendo por isso um SO do tipo *Unix*.

A interface do iOS é baseada no conceito de manipulação directa utilizando toque múltiplo e em algumas aplicações a interacção é complementada através de acelerómetros

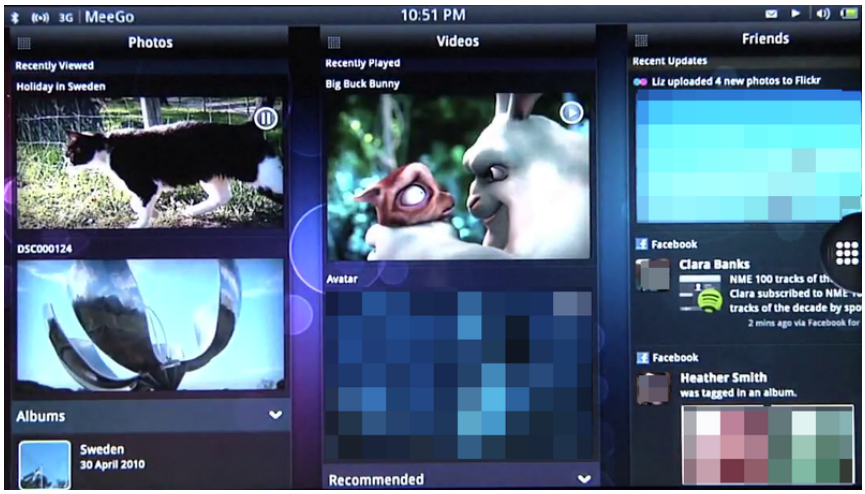

Figura 2.5: GUI *home* do *MeeGo Tab*.

internos. Na figura 2.8 encontra-se a GUI *home* do *iOS 4.2*.

2.3 Tecnologias de Desenvolvimento de Aplicações

2.3.1 Linguagens de Programação

2.3.1.1 JavaFX

JavaFX[4] é uma plataforma *Java* para criar e disponibilizar *Rich Internet application* (RIA). A versão actual 1.3 lançada a 22 de Abril de 2010 permite criar aplicações para *Desktops*, dispositivos móveis, *TV set-top boxes*, consolas de jogos e leitores de *Blu-ray*. Esta plataforma corre em qualquer *desktop* e *browser* que corra o *Java Runtime Environment* (JRE) ou em dispositivos móveis que corram Java ME.

Actualmente *JavaFX* é suportado em *Desktops* com os seguintes SOs, *Windows XP, Windows Vista, Windows 7, Mac OS X, Linux e OpenSolaris*. Alguns SOs de dispositivos móveis suportados são, *Symbian OS, Windows Mobile, Android e Real-time operating system (RTOS)*. Os dispositivos *Nokia* com o SO *Symbian OS* não correm aplicações que utilizem *JavaFX* devido ao facto de possuírem a sua própria variante de Java ME.

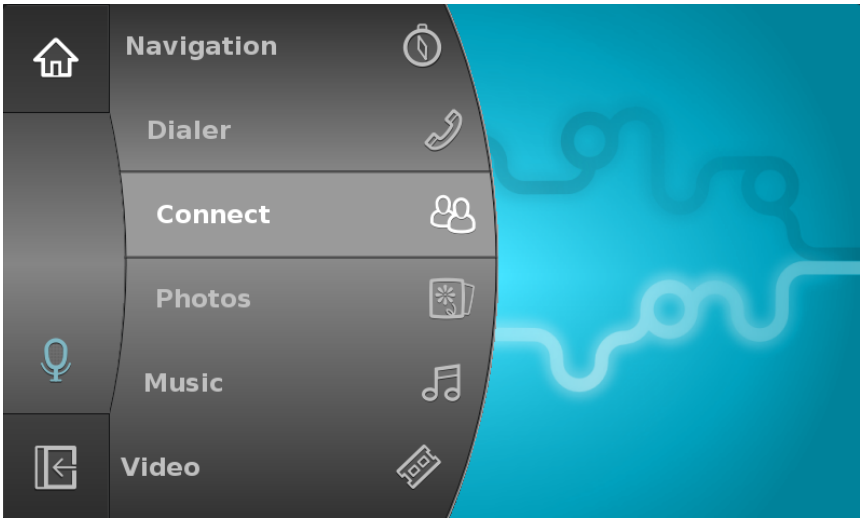

Figura 2.6: GUI *home* do *MeeGo In-Vehicle*.

2.3.1.2 Java ME

Java ME[3] é uma plataforma *Java* desenhada para dispositivos móveis e sistemas embutidos. Esses dispositivos implementam um perfil que consiste num conjunto de classes que possibilita aos programadores implementarem as aplicações consoante as características das aplicações dos pequenos dispositivos computacionais, sendo o mais comum deles o *Mobile Information Device Profile* (MIDP).

O Java ME possui configurações que são basicamente um conjunto de classes bases denominadas classes *core* que definem a *Java Virtual Machine* (JVM) de um pequeno dispositivo computacional. Existem duas configurações distintas para os dispositivos: para os com maior capacidade computacional o *Connected Device Configuration* (CDC) e para os com menor capacidade o *Connected Limited Device Configuration* (CLDC).

2.3.1.3 Adobe Flash Lite/Jarpa

A tecnologia *Java Packaging for Flash Lite Developers* (Jarpa)[15] permite aos programadores de Java ME expandir as suas aplicações de modo a embutir conteúdos *Adobe Flash Lite*.

Adobe Flash Lite[11] é uma versão mais leve do *Adobe Flash Player*, desenhada prin-

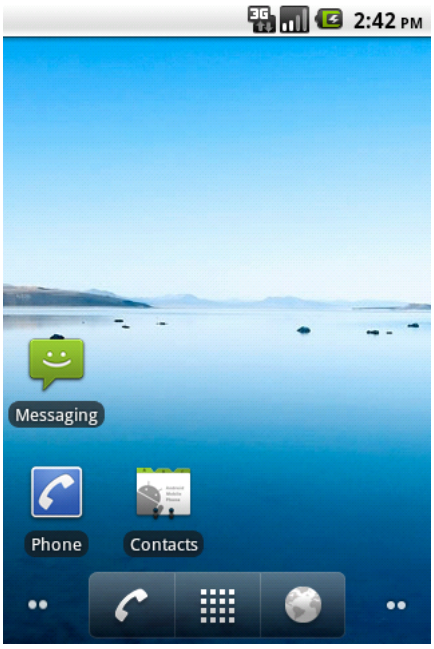

Figura 2.7: GUI *home* do *Android 2.2*.

cipalmente para dispositivos móveis, permitindo assim os utilizadores destes dispositivos assistirem a conteúdos multimédia e aplicações desenvolvidas usando *Adobe's Flash tools*.

2.3.1.4 Qt/C++

Qt[21] é uma *framework* multi-plataforma para desenvolvimento em C++, amplamente utilizada para o desenvolvimento de programas com GUI, mas também utilizada para o desenvolvimento de programas sem GUI, tais como ferramentas de consolas e servidores.

Esta *framework* permite executar os programas desenvolvidos em diversas plataformas sem ser necessário alterar o código fonte, sendo apenas necessário compilar o código para a plataforma pretendida. A *framework* é distribuída pela *Nokia* para as seguintes plataformas: *Linux/X11, Embedded Linux, Mac OS X, Microsoft Windows, Windows CE, Symbian, Maemo, MeeGo*. Desde que a *Nokia* a tornou código aberto, têm aparecido projectos externos com o objectivo de portar esta *framework* para outras plataformas, sendo alguns desses projectos *Qt for OpenSolaris, Qt for Haiku, Qt for OS/2, Qt-iPhone*[23], *Android-Lighthouse*[22], *Qt for webOS, Qt for Amazon Kindle DX*. Actualmente esta *framework* é

Figura 2.8: GUI *home* do *iOS 4.2*.

amplamente utilizada no ambiente de *desktop KDE* e nas plataformas *Maemo* e *MeeGo* dos dispositivos *Nokia*.

Esta *framework* é distribuída sob a licença *GNU Lesser General Public License* sendo livre e de código aberto.

2.3.2 Mecanismos de Processamento Multimédia

2.3.2.1 Java Mobile Media API

A *Java Mobile Media API* (*Mobile Media API; JSR 135* (MMAPI))[7] é uma API para a plataforma Java ME, que dependendo de como for implementada permite que as aplicações reproduzam e gravem sons, vídeos e capturem imagens. Esta API foi desenvolvida sobre o *Java Community Process* como JSR 135. Os formatos de conteúdos e protocolos suportados variam consoante o dispositivo.

2.3.2.2 *MPlayer*

O *MPlayer*[19] é um reprodutor multimédia livre e de código aberto que está disponível em vários SOs tais como, *Linux, Microsoft Windows, Mac OS X, OS/2, Syllable, AmigaOS* e *MorphOS*.

Este reprodutor multimédia suporta vários formatos de media sendo uma aplicação de linha de comandos que tem várias GUIs *front-ends* dependendo do SO, sendo algumas delas o *gMplayer* escrito em GTK+, o *KMPlayer* escrito em Qt, o *MPlayer OS X Extended* (para *Mac Os X*), o *MPUI-hcb* (para *Windows*) e claro o *SMPlayer* que é multi-plataforma e escrito em Qt.

Na seguinte lista estão os formatos de media suportados:

Media fixa: CDs, DVDs, Vídeo CDs

Formatos de *container*: 3GP, AVI, ASF, FLV, Matroska, MOV (QuickTime), MP4, NUT, Ogg, OGM, RealMedia, Bink

Formatos de vídeo: Cinepak, DV, H.263, H.264/MPEG-4 AVC, HuffYUV, Indeo, MJPEG, MPEG-1, MPEG-2, MPEG-4 Part 2, RealVideo, Sorenson, Theora, WMV, Bink

Formatos de áudio: AAC, AC3, ALAC, AMR, DTS, FLAC, Intel Music Coder, Monkey's Áudio, MP3, Musepack, RealAudio, Shorten, Speex, Vorbis, WMA, Bink

Formatos de legendas: AQTitle, ASS/SSA, CC, JACOsub, MicroDVD, MPsub, OGM, PJS, RT, Sami, SRT, SubViewer, VOBsub, VPlayer

Formatos de imagem: BMP, JPEG, MNG, PCX, PTX, TGA, TIFF, SGI, Sun Raster

Protocolos: RTP, RTSP, HTTP, FTP, MMS, Netstream (mpst://), SMB, ffmpeg:// (*Uses FFmpeg's protocol implementations*)

2.3.3 Comunicação Cliente/Servidor

O KXML[5] é um *parcer* de XML para Java ME que consome poucos recursos. É do tipo *pull parser*, ou seja, lê o documento um bloco de cada vez e a aplicação conduz o *parser* pedindo o próximo bloco.

O gSOAP [13] é um conjunto de ferramentas (*toolkit*) de software, com código aberto e multi-plataforma que gera código em C e C++ para SOAP/XML *Web services*, sendo livre para uso não comercial. Este *toolkit* gera C/C++ RPC codigo , XML *data bindings* e eficientes *schema-specific parsers* para SOAP *Web services*[25].

Capítulo 3

Arquitectura e Desenvolvimento

Durante o desenvolvimento deste projecto foram seguidas duas abordagens: A primeira abordagem foi desenvolver uma aplicação em *JavaME* para dispositivos móveis. Esta foi posteriormente abandonada devido a problemas com a API Multimédia e ao facto dos ecrãs dos dispositivos móveis serem de reduzidas dimensões e não muito práticos para assistir a conteúdos multimédia.

Foi então necessário adoptar outra abordagem, que consistiu em apostar nos Pad, que no momento estavam a ser muito divulgados devido ao lançamento do *iPad*. Estes dispositivos já possuem um ecrã com dimensões mais propícias para assistir a conteúdos multimédia. O próximo passo foi decidir o principal SO para o qual desenvolver esta aplicação. Depois de alguma análise, foi escolhido o *MeeGo* que tinha sido recentemente lançado, sendo resultado de uma parceria entre a Intel e a Nokia e baseado em *Linux*. Os principais oponentes, nomeadamente o *Android* e o *iOS* foram descartados devido ao mercado ser dominado pela Nokia e pela Intel nas respectivas áreas. Esta aplicação foi desenvolvida tendo em conta a possibilidade de ser utilizada em várias plataformas e diversos tipos de dispositivos.

3.1 Requisitos do Sistema

Antes de dar início ao desenvolvimento da aplicação foi necessário analisar os requisitos que a aplicação a desenvolver teria de suportar. Na lista que se segue estão referidos os requisitos que a aplicação terá que cumprir:

- Permitir criar um novo utilizador no servidor;
- Permitir efectuar o *Login/Logout* no servidor;
- Obter o conteúdo multimédia sugerido pelo servidor;
- Enviar ao servidor a classificação(*feedback*) do conteúdo assistido;
- Suportar a reprodução de diversos tipos de conteúdos multimédia;
- Suportar protocolos de *streaming*;
- Ser multi-plataforma.

3.2 Tecnologias e Plataformas

3.2.1 Sistemas Operativos/Plataformas Móveis

Inicialmente foi considerado o *Maemo* devido ao interesse no mercado *Nokia*, tendo sido instalado o *Software Development Kit* (SDK) desta plataforma para avaliar as suas características. Infelizmente foi descartada devido a serem poucos os dispositivos que utilizam esta plataforma e estes serem pouco difundidos no mercado devido ao seu elevado custo. Os dispositivos que usam esta plataforma são o *Nokia 770*, o *Nokia N800*, o *Nokia N810* e o *Nokia N900*, sendo este último o único que utiliza a versão *Maemo 5*. Durante o desenvolvimento da aplicação Qt, foram tidas em conta as características desta plataforma para que esta aplicação possa ser utilizada. Posteriormente, foi considerado o *Symbian* devido ao interesse no mercado *Nokia* e também devido ao facto de ser o SO de *smartphones* com

maior implementação no mercado, sendo utilizado em 41.2 por cento[29] dos *smartphones*. A aplicação Java ME foi desenvolvida principalmente para esta plataforma.

Quando se notou que a aplicação Java ME, devido à API multimédia MMAPI, não reproduzir vídeos codificados usando os *codecs* mais difundidos, foi necessário analisar outras opções. Como o *MeeGo* tinha sido lançado recentemente e devido ao facto de estar projectada a sua utilização em diversos tipos de dispositivos, esta opção foi imediatamente considerada. Outra das razões que levou a esta escolha foi o facto de se tratar de uma parceria entre duas empresas líderes na sua respectiva área, respectivamente a *Nokia* e a *Intel*. A aplicação Qt foi fundamentalmente desenvolvida com o objectivo de ser utilizada em dispositivos que utilizem esta plataforma.

3.2.2 Linguagens de Programação

Inicialmente, devido ao facto da aplicação a desenvolver ter como um dos principais requisitos a utilização em diversos SOs e a existência de uma interface gráfica moderna e agradável, foi considerado o desenvolvimento da aplicação em *JavaFX*. Esta plataforma tinha sido lançada recentemente para dispositivos móveis. Esta plataforma foi descartada devido ao facto dos dispositivos da marca *Nokia*, que possuem uma grande fatia do mercado, não a suportarem.

Depois de se descartar a possibilidade de desenvolver a aplicação utilizando *JavaFX*, foi considerado o desenvolvimento em Java ME devido a existirem muitos dispositivos que permitem executar aplicações desenvolvidas nesta linguagem de programação. A aplicação Java ME, como o próprio nome indica, foi desenvolvida para esta plataforma.

Tendo em conta a falta de capacidade da API multimédia do Java ME, foi necessário pensar em alternativas. Uma das alternativas seria criar uma aplicação em *Adobe Flash Lite* para reproduzir os conteúdos multimédia e aproveitar a parte da lógica e da comunicação com o servidor IPTV da aplicação Java ME. A comunicação e os pacotes de software seriam criados usando uma tecnologia chamada Jarpa[15], criando assim uma aplicação híbrida. Para criar a aplicação foi considerado utilizar como base um reprodutor

de conteúdos multimédia de código aberto. Este reprodutor criado em *Adobe Flash Lite* chama-se Dandelion Player[12]. Mas depois de se efectuar alguns testes com o código exemplo da tecnologia Jarpa, notaram-se alguns problemas no emulador do *Nokia N97*, dado que as aplicações em *Adobe Flash Lite* não executavam correctamente. Este facto, aliado à necessidade de licença para desenvolver utilizando *Adobe Flash Lite*, levou a considerar outras opções, acabando assim por se descartar o uso desta opção.

Depois de se verificar a falta de capacidades da API multimédia do Java ME e de se ter descartado a possibilidade de desenvolver em *Adobe Flash Lite* foi necessário pensar em alternativas. A alternativa encontrada foi desenvolver uma aplicação usando a *framework* Qt e utilizar o *MPlayer* para reproduzir os conteúdos multimédia. Desenvolver a aplicação utilizando esta *framework* foi a alternativa escolhida devido a ser a *framework* principal do SO *MeeGo* e também devido ao facto de ser suportada por vários SOs.

3.2.3 Mecanismos de Processamento Multimédia

No decorrer do desenvolvimento da aplicação *Java ME*, foi necessário pesquisar um mecanismo para reproduzir os conteúdos multimédia, encontrando-se a (Java Mobile Media API)MMAPI. Foi então necessário pesquisar como se utilizava esta API, analisando-se uma *demo* da utilização da mesma disponibilizada no *NetBeans*. Esta *demo* está disponível se efectuarmos os seguintes passos no *NetBeans 6.7.1*: Abrir o menu *File*, seleccionar *New Project...*, depois na janela que aparecer, seleccionar o directório *Samples/Java ME(MIDP)* e por fim seleccionar o projecto *MMAPI Demos*. Depois de analisar esta demo ficou claro que parte do código poderia ser utilizado para o projecto a desenvolver, dado que a licença desta demo o permitia.

Depois de se ter notado a falta de capacidades da API multimédia do Java ME, foi necessário pensar em alternativas. Como foi notado que a maior dificuldade na aplicação a desenvolver seria a existência de um mecanismo de processamento multimédia que suportasse vários tipos de conteúdos multimédia e também suportasse *streaming* de conteúdos sobre a rede, procurou-se um mecanismo com essas características, escolhendo-se por fim o

MPlayer. Esta escolha deve-se ao facto do *MPlayer* cumprir todos os requisitos pretendidos. Depois desta escolha procurou-se uma aplicação de código aberto que utilizasse este mecanismo e que fosse desenvolvida usando Qt para servir de base à aplicação a desenvolver. Procurou-se um reprodutor multimédia com estas características para se explorar melhor todas as capacidades do *MPlayer*, sendo o reprodutor multimédia encontrado o *SMPlayer*.

O *SMPlayer*[24] é um reprodutor multimédia multi-plataforma livre e de código aberto, sendo um *front-end* para o *MPlayer*, ou seja, faz a interacção entre o *MPlayer*[19] e o utilizador. Como o *SMPlayer* utiliza a *framework* Qt e é baseado no *MPlayer*, é bastante portável devido ao facto de tanto o Qt como o *MPlayer* estarem disponíveis em vários SOs.

Na seguinte lista encontram-se as principais características desta aplicação:

- Legendas configuráveis;
- Possibilidade de seleccionar a pista de áudio do conteúdo multimédia;
- Possibilidade de avançar ou retroceder o vídeo utilizando a roda do rato;
- Possui um equalizador de vídeo;
- Permite reproduzir os conteúdos em diversas velocidades;
- Permite aplicar filtros ao conteúdo multimédia;
- Permite ajustar o atraso das legendas e do áudio;
- Possui opções avançadas, tais como seleccionar um *demuxer* ou os *codecs* de vídeo e áudio;
- Possui lista de reprodução;
- Possui uma interface de preferências para facilmente permitir a configuração da aplicação;
- Possibilidade de automaticamente pesquisar legendas em *opensubtitles.org*;

- Possui traduções em mais de 20 linguagens;
- É multi-plataforma possuindo binários para Windows e Linux.

Esta aplicação foi utilizada como base para a aplicação Qt e mantiveram-se todas as suas características principais.

3.2.4 Comunicação Cliente/Servidor

Durante o desenvolvimento da aplicação Java ME, foi necessário utilizar um *parser* de XML para processar as mensagens recebidas do servidor, sendo utilizado o KXML devido a ser um *parser* de XML que consome poucos recursos.

Para desenvolver a aplicação Qt era necessário que esta aplicação fosse cliente dos *Web services* disponibilizados pelo servidor. Para criar o código para essa comunicação utilizou-se o gSOAP[13].

3.3 Aplicações Desenvolvidas

3.3.1 Aplicação Java ME

3.3.1.1 Requisitos Técnicos

As características de *hardware* necessárias para executar esta aplicação tendem a variar consoante o tipo de conteúdo multimédia que se pretende reproduzir e a qualidade do mesmo. O dispositivo necessita de acesso à rede onde se encontra o servidor IPTV e também necessita de acesso à rede onde se encontra o conteúdo multimédia a reproduzir.

Os requisitos técnicos de software necessários para se executar esta aplicação são o suporte da plataforma Java ME com as características referidas na seguinte lista:

- Utilizar a configuração CLDC-1.0 ou superior;
- Utilizar o perfil MIDP-2.1 ou superior;

- Suportar a API *FileConnection APIs* (JSR 75);
- Suportar a API *Mobile Media API* (JSR 135);
- Suportar a API *Security and Trust Services API* (JSR 177);
- Suportar a API *Location API* (JSR 179);
- Suportar a API *Scalable 2D Vector Graphics API* (JSR 226).

3.3.1.2 Tecnologias Utilizadas

Nesta aplicação foram usadas as seguintes tecnologias: *JavaMe MMAPI Sample* devido ao facto de ser de código aberto e servir como uma boa base para o desenvolvimento da parte responsável por reproduzir os conteúdos multimédia e o KXML, que foi utilizado para fazer o *parsing* das respostas do servidor. Esta aplicação foi desenvolvida usando o NetBeans 6.7.1 para a plataforma Java ME.

3.3.1.3 Arquitectura

Relativamente à arquitectura interna da aplicação Java ME, esta é constituída por três pacotes, estando estes representados no diagrama *Unified Modeling Language* (UML) da figura 3.1 que mostra o encapsulamento interno dos pacotes criados. No pacote *DataManager* é efectuada a lógica e a interacção cliente servidor e também são armazenadas todas as informações dos conteúdos multimédia e do utilizador. O pacote Interface é onde se encontra o código relativo à *MIDlet* da interface de registo de um novo utilizador. No pacote *MMAPIInterface* residem todas as classes relativas à demo da MMAPI que serviu de base para a interface onde é efectuada a autenticação do utilizador e são reproduzidos os conteúdos multimédia.

Na figura 3.2 encontra-se o diagrama de classes do pacote *MMAPIInterface*, onde as principais classes são a *MyData* e a *ManageData*. Na *MyData* são armazenados todos os dados do utilizador e efectuadas a maioria das operações que envolvem o contacto com o servidor. Na outra classe principal, *ManageData*, são armazenadas as informações relativas

à sessão, tais como, se o utilizador está autenticado, os *links* dos conteúdos multimédia e a *thread* relativa ao *keepalive* cuja função é evitar que o utilizador seja desautenticado no servidor caso este fique inactivo por muito tempo.

O diagrama de classes do pacote Interface está ilustrado na figura 3.3, onde existe apenas uma classe denominada *Register*, é a *MIDlet* da interface de registo.

No diagrama de classes do pacote *MMAPIInterface*, ilustrado na figura 3.4, são apresentadas as classes necessárias para a interface de autenticação e para a interface de reprodução dos conteúdos multimédia. Estas classes são uma cópia das respectivas classes do projecto *MMAPI Demos* que serviu de base para este projecto. Estas classes sofreram alterações de modo a incluírem a interface de autenticação e permitirem a reprodução dos conteúdos sugeridos pelo servidor.

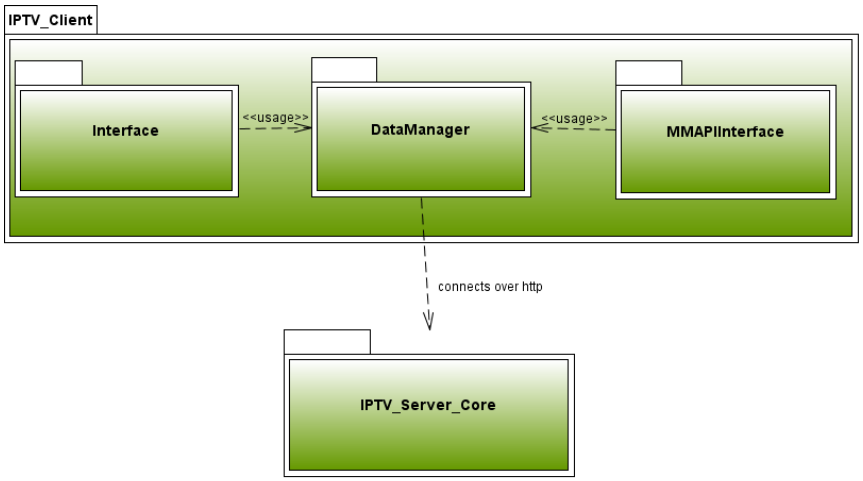

Figura 3.1: Diagrama de pacotes da aplicação desenvolvida em Java ME e algumas relações.

3.3.1.4 Desenvolvimento

A fase inicial do desenvolvimento desta aplicação passou pela análise da possibilidade da implementação de todos os requisitos usando esta plataforma. Infelizmente apenas foi possível concluir que as capacidades da MMAPI eram insuficientes já depois da aplicação estar quase concluída.

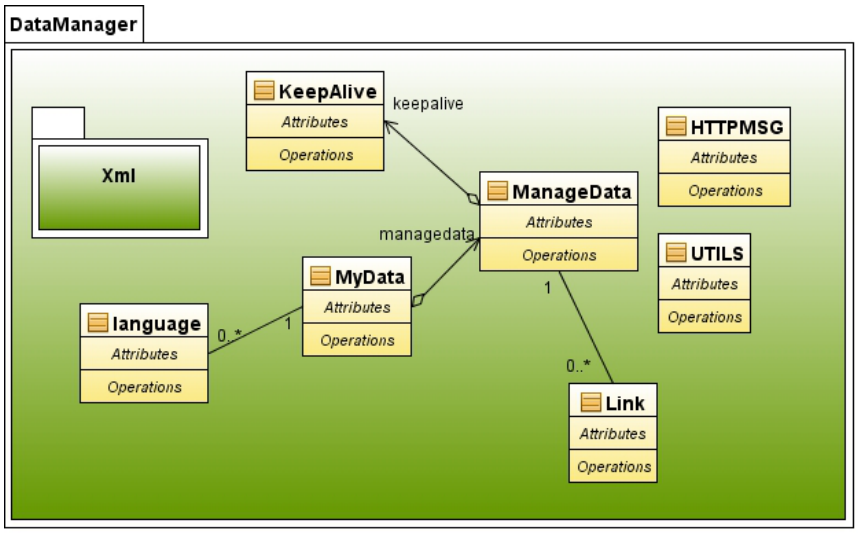

Figura 3.2: Diagrama de classes do pacote *DataManager*.

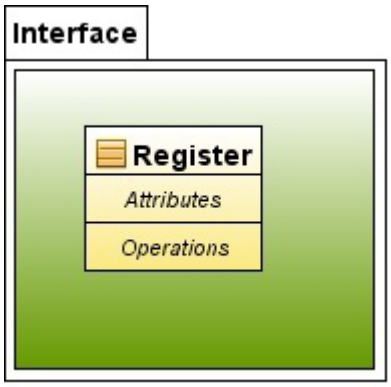

Figura 3.3: Diagrama de classes do pacote Interface.

Após a análise referida anteriormente, seguiu-se o desenvolvimento da comunicação com o servidor. Podemos observar o diagrama de classes da figura 3.2 onde estão representadas as classes responsáveis pela lógica e comunicação com o servidor.

De seguida vai ser dado início à explicação da função das classes principais. A *MyData* é onde são armazenados os dados do utilizador e efectuadas a maioria das operações que envolvem o contacto com o servidor. Outra classe principal é a *ManageData* onde são armazenadas as informações relativas à sessão, tais como, se o utilizador está autenticado, os *links* dos conteúdos multimédia e a *thread* relativa ao *keepalive*. A classe *keepalive* é

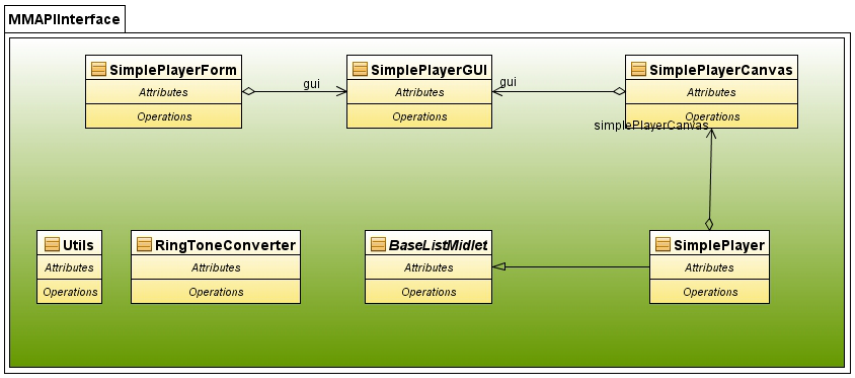

Figura 3.4: Diagrama de classes do pacote *MMAPIInterface*.

responsável por evitar que o utilizador seja desautenticado no servidor caso fique inactivo por muito tempo. A classe *HTTPMSG* possui métodos estáticos cuja função é enviar as mensagens para o servidor e retornar a respectiva resposta. A classe *UTILS* é onde estão armazenadas informações de configuração, tais como o endereço do servidor e o *time out* para manter a sessão activa. Por fim, temos o pacote *Xml* onde se encontram todas as classes relativas ao *parsing*(leitura) das mensagens XML recebidas.

Depois de criada a comunicação com o servidor, foram criados alguns GUIs usando SVG; o único implementado encontra-se no pacote Interface mostrado na figura 3.3. O GUI implementado nesta fase é a interface de registo. Na figura 3.5 encontra-se a interface de registo desenvolvida, sendo de notar que esta interface ainda iria sofrer melhorias. Devido ao facto de esta aplicação ter sido abandonada, estas não foram efectuadas.

Por fim, iniciou-se a integração com o projecto *MMAPI Demos* que serviu de base para a utilização da MMAPI, cujo objectivo é reproduzir os conteúdos sugeridos pelo servidor. O código desta integração está no pacote *MMAPIInterface*, correspondente à figura 3.4. As classes presentes neste pacote são uma cópia das respectivas classes do projecto *MMAPI Demos*. Estas classes sofreram alterações de modo a permitir a interacção com o servidor e a inclusão da interface de autenticação. Na figura 3.6 encontra-se a interface de autenticação e nas figuras 3.7, 3.8 e 3.9 encontram-se a interface de reprodução e algumas das suas opções.

Quando a integração estava quase completa efectuaram-se alguns testes e verificou-se

que quando se tentava reproduzir algum conteúdo que não estivesse alojado no dispositivo, a MMAPI transferia primeiro o conteúdo para o dispositivo e só no fim é que o reproduzia, o que introduzia um tempo de espera inaceitável antes da reprodução do conteúdo. Depois de ter sido efectuada uma pesquisa por soluções para resolver o problema, não foi encontrada nenhuma solução viável. Isto aliado ao facto da MMAPI não reproduzir diversos dos *codecs* mais utilizados actualmente, tais como o *h.264*, levou ao abandono do desenvolvimento desta aplicação e ao início da procura de alternativas.

É preciso ter em conta que esta aplicação está integrada com uma versão mais antiga do servidor. Nesta versão a interacção com o servidor não é efectuada através do acesso a *web services*, mas sim através de *http* onde se envia a variável iptvxml com o XML relativo à operação que se deseja efectuar e recebe-se a resposta em XML na mesma variável.

Para desenvolver esta aplicação utilizou-se o NetBeans devido à experiência anterior com este software, como também se utilizou o emulador do Nokia N97 para testar a aplicação.

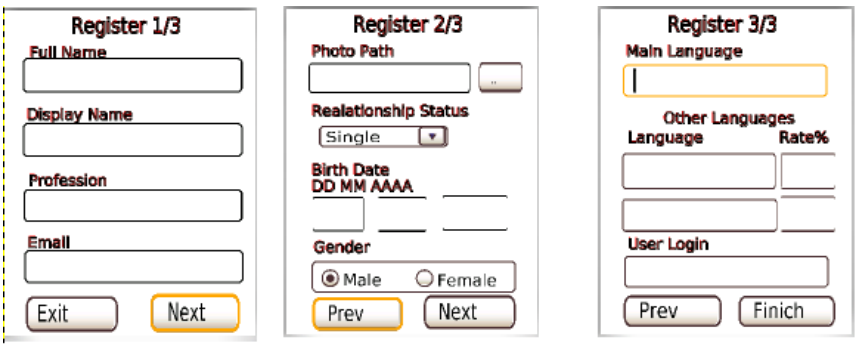

Figura 3.5: GUIs de registo da aplicação Java ME, criadas usando SVG.

3.3.2 Aplicação Qt

3.3.2.1 Requisitos Técnicos

A aplicação desenvolvida usando Qt teve como base o SMPlayer e a única biblioteca externa usada foi o gSOAP, logo esta aplicação pode ser executada em todos os sistemas onde tanto o SMPlayer como o gsoap possam ser utilizados. Alguns dos SOs onde esta

Figura 3.6: Interface de autenticação da aplicação Java ME.

Figura 3.7: Interface de reprodução da aplicação Java ME.

aplicação pode ser utilizada são o Windows e o Linux. Como o Maemo e o MeeGo são baseados em Linux, esta aplicação também pode ser utilizada nestas plataformas.

As características de *hardware* necessárias para executar esta aplicação tendem a variar consoante o tipo de conteúdo multimédia que se pretende reproduzir e a qualidade do mesmo. O dispositivo necessita de acesso à rede onde se encontra o servidor IPTV e também necessita de acesso à rede onde se encontra o conteúdo multimédia a reproduzir.

Figura 3.8: Algumas opções da interface de reprodução da aplicação Java ME.

Figura 3.9: Interface de reprodução da aplicação Java ME em ecrã inteiro.

3.3.2.2 Tecnologias Utilizadas

Esta aplicação foi desenvolvida em C++ usando as ferramentas Qt Creator, Qt Linguist e o *toolkit* do gSOAP.

O *MPlayer*[19] foi o mecanismo de reprodução utilizado devido a ser multi-plataforma, à sua constante evolução, às suas capacidades de reproduzir a maioria dos formatos e *codecs* e também pelo facto de suportar diversos protocolos de *streaming*.

O SMPlayer foi utilizado como base para possibilitar o desenvolvimento de uma aplicação que explore a maioria das capacidades do MPlayer. Este reprodutor de conteúdos mul-

timédia serviu como base devido ao facto de ser de código aberto, multi-plataforma, altamente configurável, possuir muitas funcionalidades, ser estável, largamente difundido e o motor que utiliza para reproduzir os conteúdos multimédia ser o *MPlayer*.

Para criar a comunicação com o servidor foi utilizado o gSOAP *toolkit*. Estas ferramentas foram utilizadas para facilitar a integração da aplicação com os *Web Services* SOAP disponibilizados pelo servidor e devido ao facto de serem suportadas em diversas plataformas, para além de também serem sugeridas no fórum da *Nokia*[14]. As ferramentas utilizadas foram os comandos wsdl2h e soapcpp2.

3.3.2.3 Arquitectura

Na figura 3.10 encontra-se um diagrama que representa a arquitectura geral da aplicação. É de notar que o pacote SMPlayer representa todas as classes e pacotes relacionados com o mesmo. A classe e os pacotes inseridos no pacote *SMPlayer* foram criadas no âmbito desta dissertação. A classe *iptvclientgui* foi criada com o objectivo de inserir mais uma interface à aplicação, sendo esta mais vocacionada para dispositivos com *Thouchscreen* e ecrãs de pequenas e médias dimensões. Esta classe teve como base a classe *DefaultGuii* do SMPlayer. O pacote *serverconnectioni* contém as classes relacionadas com a interacção cliente-servidor e a lógica associada a esta interacção. Por fim, o pacote *interfaceclientserveri* contém as classes relacionadas com as interfaces de autenticação, estado, classificação e registos criados.

Na figura 3.11 encontra-se um diagrama de classes do pacote *serverconnectioni*. Este pacote contém as classes relacionadas com a interacção cliente-servidor e a lógica associada a esta interacção. Entre ambas as classes é de destacar a *clientlogic* que gere toda a informação e lógica necessárias para a interacção com o servidor. A classe *keepAlive* é responsável por evitar que o utilizador seja desautenticado no servidor caso fique inactivo por muito tempo. A classe *WsClient* é responsável pela abstracção na chamada aos *Web Services* e a transição dos tipos de dados usados no gSOAP para os usados no Qt. As restantes classes neste pacote foram geradas automaticamente pelo gSOAP.

Na figura 3.12 encontra-se o diagrama de classes do pacote *interfaceclientserveri*. A classe *autenticationdialog* está relacionada com a interface e lógica da GUI de autenticação mostrada na figura 3.13. A classe *state_iptv* está relacionada com a interface e lógica da GUI do estado, sendo apresentada na figura 3.15. A classe *rate_iptv* está relacionada com a interface e lógica da GUI de rate mostrada na figura 3.16 e por fim a classe *regist* está relacionada com a interface e lógica da GUI do registo e é mostrada na figura 3.14.

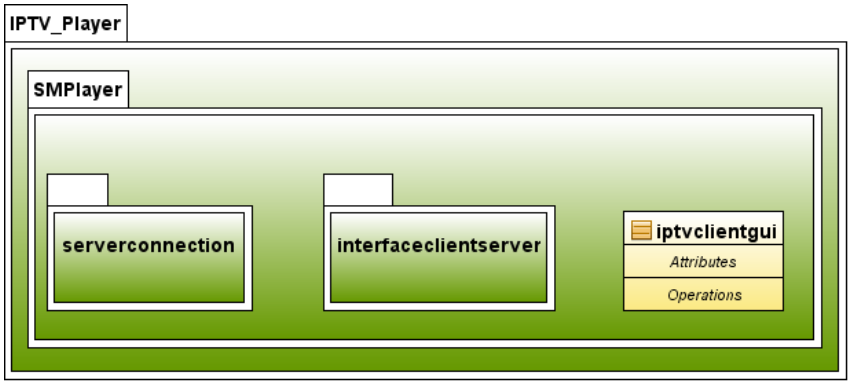

Figura 3.10: Diagrama de pacotes da aplicação desenvolvida em Qt.

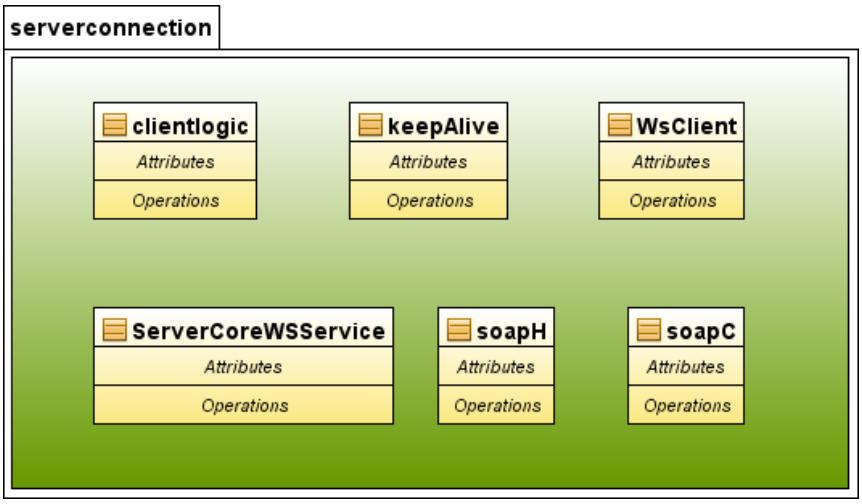

Figura 3.11: Diagrama de classes do pacote *serverconnectioni*.

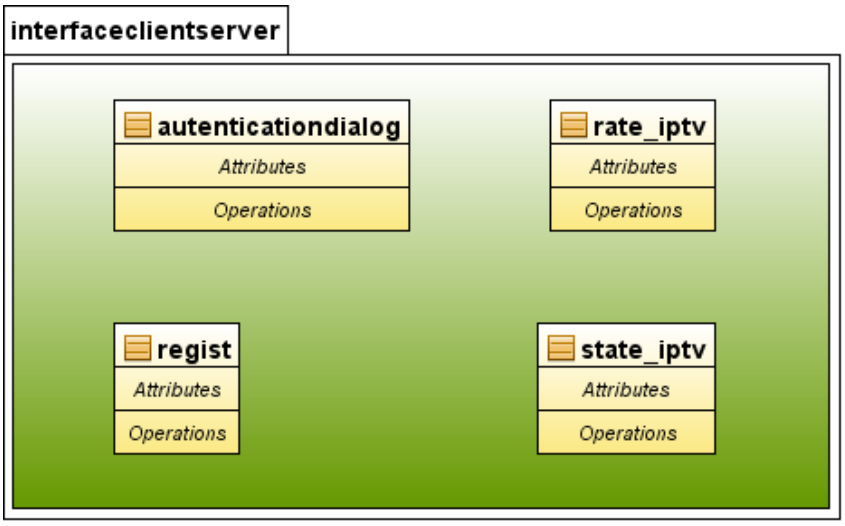

Figura 3.12: Diagrama de classes do pacote *interfaceclientserveri*.

3.3.2.4 Desenvolvimento

Para o desenvolvimento desta aplicação foi inicialmente verificado se era possível implementar todos os requisitos usando esta plataforma. Devido aos problemas com a aplicação Java ME, testou-se o mecanismo de processamento multimédia mais profundamente para ver se cumpria os requisitos.

Depois desta análise, começou por se desenvolver a comunicação com o servidor. Podemos observar o diagrama de classes da figura 3.11, onde estão representadas as classes responsáveis pela lógica e comunicação com o servidor. Inicio agora a explicação da função das classes. A classe *clientlogic* é a que gere toda a informação e lógica necessárias para a interacção com o servidor. A classe *WsClient* é responsável por uma abstracção na chamada aos *Web Services* e a transição dos tipos de dados usados no gSOAP para os usados no Qt. A classe *keepAlive* é responsável por evitar que o utilizador seja desautenticado no servidor caso fique inactivo por muito tempo. As outras classes foram geradas automaticamente usando o gSOAP Toolkit e a sua função é a comunicação com os *Web Services* disponibilizados pelo servidor.

Depois de criada a comunicação com o servidor, iniciou-se a integração do código do SMPlayer de modo a permitir reproduzir os conteúdos sugeridos pelo servidor. Depois

Figura 3.13: Interface de Autenticação da aplicação Qt.

de se conseguir reproduzir os conteúdos fornecidos e classificá-los, iniciou-se a criação de mais uma interface principal para o SMPlayer, ilustrada na figura 3.17. Para a criação desta interface criou-se a classe *iptvclientgui* que teve como base a classe *DefaultGuii* do SMPlayer. É de notar que esta interface foi sofrendo melhorias sucessivas durante o desenvolvimento da aplicação. Durante o desenvolvimento da aplicação também se teve o cuidado de manter todas as características e capacidades do SMPlayer.

Depois de se ter criado mais uma interface principal, criaram-se e implementaram-se as seguintes interfaces: a interface de autenticação mostrada na figura 3.13, a interface de Estado mostrada na figura 3.15 , a interface de classificação mostrada na figura 3.16 e, por fim, a interface de registo mostrada na figura 3.14.

Depois da criação de todas as interfaces, iniciou-se a melhoria de certos detalhes da aplicação, nomeadamente melhoria na gestão de erros e adição da capacidade de criar traduções facilmente usando a ferramenta Qt Linguist. De momento, já foi criada a tradução em Português e tendo em conta que a língua base da aplicação é Inglês, a aplicação já está disponível em Inglês e Português. Além das línguas também foi criado um novo e mais intuitivo método de classificar os conteúdos multimédia, sendo este as estrelas pre-

Figura 3.14: Interface de Registo da aplicação Qt.

sentes na interface da figura 3.17. Também se adicionou a possibilidade do utilizador se autenticar automaticamente, de modo a ficar totalmente abstraído das comunicações com o servidor.

Para desenvolver esta aplicação utilizou-se o Qt Linguist e o Qt Creator devido ao facto de serem as ferramentas sugeridas para desenvolver aplicações para a plataforma MeeGo. Também se utilizou o gSOAP Toolkit para o desenvolvimento da comunicação com os *Web Services* servidor.

3.3.2.5 Características e Configurações

Esta aplicação foi desenvolvida tendo em conta a capacidade de ser executada em diversos dispositivos com características diferentes. Esta aplicação, além de possuir quatro interfaces principais, também se integra com a plataforma em que está a ser executada, tanto no esquema de cores como no estilo e na linguagem, caso possua uma tradução para a respectiva linguagem. Além de possuir todas estas integrações também é possível, através da alteração do ficheiro de configuração, alterar o aspecto da interface principal de diversas

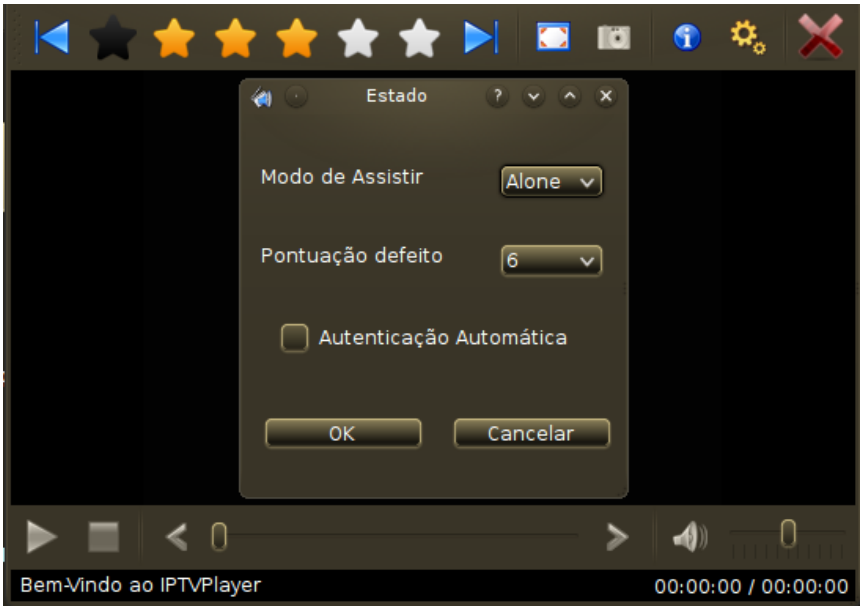

Figura 3.15: Interface de Estado da aplicação Qt.

formas.

O ficheiro de configurações possui muitos critérios a configurar, sendo a maioria deles herdados do SMPlayer e os outros criados para a integração com o servidor ou para aumentar as capacidades de configuração das interfaces. Este ficheiro no SO Linux encontra-se na seguinte localização /.conf/iptvplayer/iptvplayer.ini, onde à pasta (/) corresponde a pasta local do utilizador, o que significa que cada utilizador vai ter a sua própria configuração. No anexo A encontram-se um exemplo de um ficheiro de configuração e a tabela A.1 que contem informações relativas a cada grupo de configuração e a sua posição no ficheiro previamente referido.

Os grupos que foram adicionados a este ficheiro de configuração no âmbito desta dissertação foram o *iptvSettings* e o *iptv_gui*. Este último é muito semelhante ao *default_gui* devido ao facto de a classe que cria este grupo ser baseada na classe que cria o grupo *default_gui*. Além destes grupos, também foi necessário fazer uma pequena alteração no código que cria o grupo *playlist_contentsi*, de modo a permitir o correcto funcionamento do sistema sendo também adicionadas algumas acções ao grupo *actions*.

De seguida vai ser feita uma descrição da função das configurações adicionadas ao

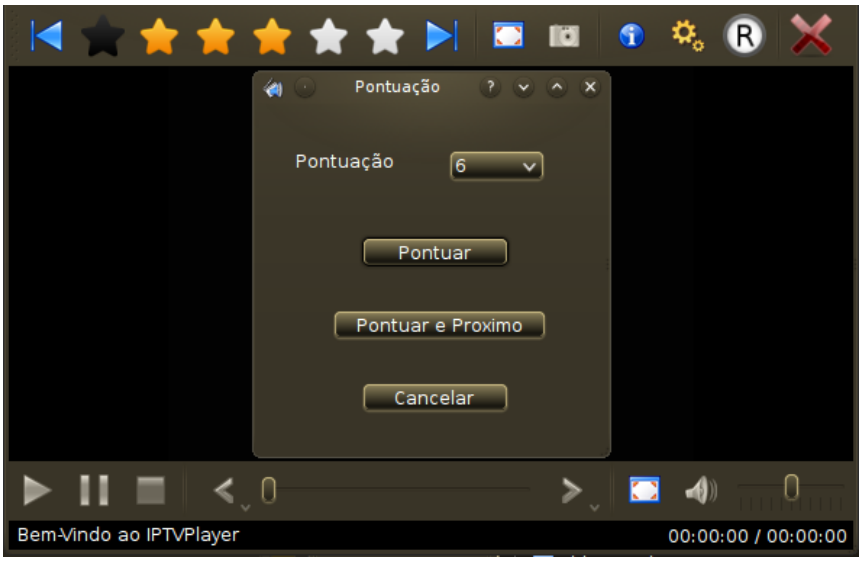

Figura 3.16: Interface de Classificação da aplicação Qt.

ficheiro de configuração.

Começando pelo grupo *iptvSettings*, na linha 269 temos a configuração do *watchmode*, que é basicamente o modo em que se estava a assistir ao conteúdo multimédia quando se fechou a aplicação; na linha 270 temos a configuração do *defaultFeedback*, que é a cotação dada por defeito ao conteúdo multimédia; na linha 271 temos a configuração do *serverURL*, que é o endereço necessário para se aceder aos *web services* disponibilizados pelo servidor; nas linhas 272 e 273 temos a configuração do *login* e da *pass*, o nome do utilizador e a respectiva senha secreta, que são armazenadas caso a configuração da linha 274, o *remember* seja igual a *true*. Esta opção está relacionada com a capacidade do sistema armazenar as informações relacionadas com o utilizador. Na linha 270 temos a configuração do *autoautenticate*, que está relacionada com a capacidade do utilizador se autenticar automaticamente sem necessitar de utilizar a interface de autenticação, permitindo assim criar-se uma abstracção do utilizador relativamente à comunicação com o servidor.

No grupo *actions*, que associa acções a atalhos de teclados, foram adicionadas as seguintes acções: a acção *state_iptv*, linha 515, que mostra a interface de estado ilustrada na figura 3.15; a acção *rate_iptv*, linha 516, que mostra a interface de classificação ilustrada na figura 3.16 e as acções das linhas 517 a 522 que são acções relacionadas com a classificação

Figura 3.17: Interface de Principal da aplicação Qt criado (IPTVGUI).

do conteúdo por exemplo, se se efectuar a acção *star5_iptv* o conteúdo é classificado com 5 estrelas.

O grupo *playlist_contentsi* teve de sofrer algumas alterações para permitir a correcta integração com o servidor. Uma dessas alterações teve em conta que ao armazenar um item também terá de ser armazenado o seu respectivo *id* e *author*. Esta alteração teve de ser efectuada devido ao facto de quando se termina a aplicação o item que está a ser reproduzido não é classificado, logo, se a opção de armazenar os conteúdos da lista de reprodução estiver activa este conteúdo irá ser armazenado e depois carregado quando a aplicação iniciar novamente. O problema surge no momento em que o cliente solicita ao servidor um novo conteúdo. O servidor pode voltar a sugerir o conteúdo armazenado, mas como a aplicação não permite adicionar duas vezes o mesmo conteúdo este não é adicionado, por isso para o cliente posteriormente conseguir classificar o conteúdo armazenado necessita também de armazenar o seu *id*.

De seguida vai ser efectuada uma descrição das configurações do grupo *iptv_gui* que, como referido anteriormente, é muito semelhante ao grupo *default_gui* devido ao facto de a classe que cria este grupo ser baseada na classe que cria o grupo *default_gui*. Além

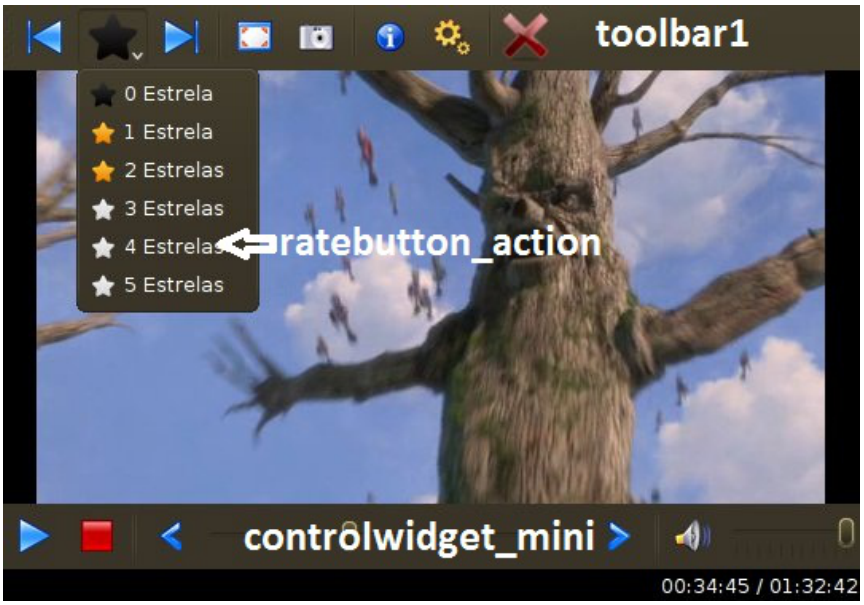

Figura 3.18: Interface principal com legendas.

das configurações comuns que funcionam de modo semelhante, também foram adicionadas configurações para permitir alterar o tamanho dos elementos das barras da interface. Se essas configurações de tamanho forem zero ou inferiores, será utilizado o tamanho padrão. Nas linhas 538 e 539 encontra-se a configuração do tamanho dos elementos da *toolbar1*, nas linhas 540 e 541 encontra-se a configuração do tamanho dos elementos da *controlwidget* e da *controlwidget_mini* e por fim nas linhas 542 e 543 encontra-se a configuração do tamanho dos elementos do *floating_control* sendo esta a barra que aparece quando se está no modo de ecrã inteiro. Relativamente às configurações similares ao grupo *default_gui*, podemos destacar: a *actions\toolbar1* na linha 534 que configura os elementos a aparecerem na *toolbar1*; a *actions\controlwidget* na linha 535 que configura os elementos a aparecerem na *controlwidget*; a *actions\controlwidget_mini* na linha 536 que configura os elementos a aparecerem na *controlwidget_mini* e a *actions\floating_control* na linha 537 que configura os elementos a aparecerem na *floating_control*. Relativamente aos elementos que podem ser adicionados em cada barra, estes são todas as acções presentes no grupo *actions* e mais alguns que foram criados propositadamente para esse efeito, sendo estes por exemplo o *rewindbutton_action*, o *forwardbutton_action*, o *timeslider_action*, o *volumeslider_action* e o

ratebutton_action. Este último foi criado no âmbito desta dissertação para que o mecanismo de classificação ocupe menos espaço na barra. Na figura 3.18 encontra-se uma imagem com a interface principal com legendas a indicar o nome de cada barra, sendo de notar que a barra *controlwidget* aparece quando o tamanho da janela proporcionar o seu aparecimento; caso contrário, no lugar desta aparece a *controlwidget_mini*.

Depois de analisar as capacidades de configuração desta aplicação, podemos concluir que apenas editando o ficheiro de configuração é possível adaptar a aplicação não apenas a diversos tipos de dispositivos com características e capacidades muito distintas mas também a diversos tipos de utilizadores.

Capítulo 4

Testes de Desempenho

A maioria dos testes efectuados à aplicação Qt foram executados durante o desenvolvimento da mesma, de modo a verificar se as alterações introduzidas não afectaram o correcto funcionamento da aplicação. No entanto, após o desenvolvimento estar completado, efectuaram-se alguns testes para avaliar o desempenho da aplicação, nomeadamente o tempo que a aplicação demora a iniciar e o tempo que demora desde que se faz um pedido de um conteúdo multimédia até o mesmo começar a ser reproduzido.

Os testes foram efectuados num sistema computacional com as seguintes características gerais:

SO *Kubuntu 10.04 Desktop edition*;

Processador Um processador *Intel Centrino Duo T2300 Dual Core 1.66 GHZ, 2MB cache*;

RAM 3 GBytes;

Disco(s) Um disco de 100 GBytes IDE;

Fornecedor *Asus*;

Placa gráfica *ATI X1600* com 256MB de memoria dedicada.

Os testes à aplicação Qt não foram executados directamente no SDK do MeeGo porque o tempo de resposta deste era exageradamente elevado, devido ao facto de a aceleração 3D

não funcionar correctamente neste sistema computacional. Esta situação ocorreu devido ao facto da placa gráfica já se inserir na classe *legacy*, não existindo assim *drivers* proprietários para a versão do SO que o sistema computacional utilizava.

Para medir os valores presentes nas seguintes tabelas recorreu-se à interface de *log* do SMPlayer cujo objectivo é o *debug* da aplicação, esta interface representa uma lista de acções e o tempo em que foram efectuadas. Nestas medições removeram-se os tempos de acesso aos *Web Services* do servidor, para que estes testes apenas avaliassem as características da aplicação desenvolvida.

Na tabela 4.1 encontram-se várias medições que correspondem ao tempo que a aplicação leva desde o instante em que é despoletada a ordem de iniciar até à apresentação da interface de autenticação.

Descrição	**Tempo (Segundos)**
Primeira medição	1,172
Segunda medição	0,963
Terceira medição	1,087
Quarta medição	1,068
Quinta medição	0,992
Sexta medição	0,981
Sétima medição	1,002
Oitava medição	0,859
Nona medição	0,872
Décima medição	0,931
Média	0,9927

Tabela 4.1: Tempos de inicialização da aplicação

Relativamente ao tempo que a aplicação demora a iniciar a reprodução de um novo conteúdo multimédia, este varia não só consoante o tipo, localização e características do conteúdo multimédia, mas também consoante as configurações da aplicação.

Na tabela 4.2 encontram-se diversos valores médios do tempo que a aplicação demora desde que é feito o pedido de um novo conteúdo multimédia até se iniciar a reprodução do mesmo. O novo conteúdo multimédia tem as seguintes características:

Tamanho 700MB;

Duração 01:32:42;

Demuxer avi;

Resolução 640 x 352;

Taxa de bits do vídeo 912 kbps;

Taxa de bits do aúdio 128 kbps;

Frames por segundo 23.976;

Localização file:///media/FAT32/Filmes/teste1.avi;

Descrição	**Tempo (Segundos)**
Primeira medição	5,303
Segunda medição	5,093
Terceira medição	4,964
Quarta medição	5,008
Quinta medição	5,125
Sexta medição	4,841
Sétima medição	5,192
Oitava medição	5,146
Nona medição	4,872
Décima medição	4,931
Média	5,0475

Tabela 4.2: Tempos de inicialização do primeiro conteúdo

Na tabela 4.3 encontram-se diversos valores médios dos tempos que a aplicação demora desde que é feito o pedido de um novo conteúdo multimédia até se iniciar a reprodução do mesmo. O novo conteúdo multimédia tem as seguintes características:

Tamanho 174 MB;

Duração 00:07:48;

Demuxer avi;

Resolução 1280 x 528;

Taxa de bits do vídeo 2897 kbps;

Taxa de bits do aúdio 224 kbps;

Frames por segundo 23.976;

Localização file:///media/FAT32/Filmes/teste2.avi;

Descrição	**Tempo (Segundos)**
Primeira medição	3,879
Segunda medição	3,793
Terceira medição	3,664
Quarta medição	3,582
Quinta medição	3,548
Sexta medição	3,841
Sétima medição	3,692
Oitava medição	3,633
Nona medição	3,872
Décima medição	3,631
Média	3,7135

Tabela 4.3: Tempos de inicialização do segundo conteúdo

Como podemos verificar, os tempos desde o pedido até ao início da reprodução de um novo conteúdo multimédia variam consoante as características do mesmo. Relativamente a estes resultados, os valores de espera são aceitáveis. Além disso, durante este processo, a aplicação vai dando algum *feedback* das funções que está a executar.

Capítulo 5

Conclusões e Trabalho Futuro

Relativamente ao trabalho futuro relacionado com o cliente, este vai variar consoante o dispositivo onde a aplicação vai ser utilizada. Isto deve-se ao facto de a aplicação Qt ser multi-plataforma e estar preparada para dispositivos com características muito distintas. Esta aplicação pode sofrer melhorias de modo a integrar-se mais facilmente com as características específicas de cada dispositivo, tais como a integração com os sensores do dispositivo, adaptação ao modo como são geridos os conteúdos multimédia, melhor integração das interfaces com as características e capacidades de cada tipo de dispositivo e criação de interfaces para a pesquisa de conteúdos. Além destas melhorias, também pode ser necessário adaptar a aplicação às alterações que possam ser efectuadas no servidor.

Como a aplicação vai estar relacionada com a evolução do servidor, esta também pode ter que sofrer alterações relacionadas com o modelo de negócio que possa ser aplicado ao servidor e as alterações que este terá que sofrer para implementar este mesmo modelo de negócio. Por exemplo, um modelo de negócio que possivelmente seria um bom ponto de partida e aliciante para os clientes seria uma adaptação do modelo actualmente utilizado nas televisões livres actuais, ou seja, a fonte de rendimentos provém da reprodução de conteúdos publicitários. Nesta adaptação, o servidor, além de sugerir o conteúdo multimédia, também sugeria os conteúdos publicitários a reproduzir e a altura do conteúdo multimédia em que estes seriam reproduzidos. Os conteúdos publicitários também seriam sugeridos consoante o perfil do utilizador. Além deste método, o utilizador também poderia

ter à sua disposição a possibilidade de subscrever uma assinatura mensal que o permitiria diminuir a quantidade de conteúdos publicitários apresentados, e/ou facultar acesso a mais conteúdos. Este modelo de negócio poderia ser aliciante para o cliente porque para aderir ao serviço apenas necessitava de possuir uma ligação à Internet com largura de banda suficiente para a reprodução dos conteúdos, não sendo assim necessário subscrever nenhuma assinatura para obter o serviço básico. Do lado do provedor do serviço, este tinha a sua fonte de rendimentos vinda dos conteúdos publicitários e dos clientes que optassem pela assinatura mensal. Durante a reprodução dos conteúdos publicitários, também poderia ser adicionado um serviço que permitisse ao utilizador adquirir o artigo publicitado através de um clique de botão do seu controlo remoto. Neste caso, apareceria uma interface onde seria requisitada a validação da compra através de uma senha secreta e seleccionado o modo de pagamento. O pagamento poderia ser efectuado posteriormente na factura da assinatura do serviço, e caso o cliente não mostrasse interesse contrário, o produto seria enviado para a sua morada.

Para a implementação do modelo de negócio sugerido acima, seria necessário criar do lado do servidor um módulo para a gestão dos conteúdos multimédia e o método utilizado para os fazer chegar ao cliente. Um dos problemas da implementação em larga escala deste serviço seria a sobrecarga a que a rede estaria sujeita. Para aliviar essa carga, poderiam ser tomadas diversas medidas e tirar-se proveito do modo de operar do sistema onde o cliente estivesse implementado. Por exemplo, se o cliente fosse uma Box da Meo, poderíamos tirar proveito do facto da mesma estar geralmente activa todo o dia, para transferir os conteúdos multimédia em horas em que a rede estivesse menos sobrecarregada. Além disso, também podíamos implementar um sistema similar aos torrents[26], onde a Box iria descarregar o conteúdo da Box mais próxima que possuísse esse conteúdo. Além destas medidas, poderia ainda ser agendada a distribuição de conteúdos mais requisitados através de sessões *multicast*. Neste caso, as Boxs dos utilizadores que pudessem estar interessados nos conteúdos a serem transmitidos por *multicast*, seriam notificadas da hora de início da sessão e, caso tivessem possibilidade, adeririam à sessão e gravavam o conteúdo no seu disco. Estas medidas iriam minimizar o número de sistemas computacionais necessários

para implementar o sistema e também os recursos de rede necessários.

Outra abordagem relativa à implementação de um serviço de sugestão automática de conteúdos num serviço do tipo Meo seria a possibilidade do servidor em vez de enviar a localização de um conteúdo multimédia, enviar um canal e o intervalo de tempo em que o conteúdo iria ser reproduzido. A Box, se na altura tivesse condições, iria gravar o conteúdo no seu disco, sendo este posteriormente reproduzido quando o utilizador mostrasse interesse em assistir aos conteúdos sugeridos automaticamente. Neste caso, o servidor seria uma espécie de sistema automático de agendamento de gravações. Um sistema deste tipo iria tirar proveito das programações dos canais de TV a que o utilizador tem acesso para obter diversos conteúdos que pudesse sugerir.

Relativamente a melhorias que o cliente poderia receber em relação à gestão dos conteúdos, caso o dispositivo destino possuísse as capacidades de *hardware* necessárias, podiam ser criadas integrações com diversos gestores de transferências usando *metalinks*[17], permitindo assim pré-transferir os conteúdos de diversos tipos de fontes.

Para melhorar a interface de registo, também poderia ser criada uma aplicação separada que se adaptasse melhor ao dispositivo em questão, dado que é necessário aceder à conta de correio electrónico para validar uma nova conta de utilizador. Assim, não seria necessário efectuar o registo na aplicação responsável por reproduzir os conteúdos. Esta aplicação poderia ser executada durante a instalação da aplicação principal, quando o utilizador executasse pela primeira vez a aplicação principal ou quando o utilizador mostrasse interesse em criar uma nova conta. Esta aplicação poderia utilizar métodos para facilitar a criação de um novo utilizador e completar automaticamente o perfil do mesmo. Estes métodos poderiam ser, por exemplo, o uso de serviços disponibilizados pelas redes sociais para obter informações relacionadas com o utilizador e as suas características. Também se poderiam ir procurar informações no sistema, tais como o histórico de páginas de Internet visitadas e alguns dados do utilizador, informações estas que seriam posteriormente processadas e utilizadas para facilitar o registo e completar as informações sobre as características do utilizador.

Tendo em conta as propostas referidas neste capítulo, podemos concluir que as melhorias

e adaptações a efectuar estão condicionadas à evolução do servidor e o tipo de dispositivo onde a aplicação irá ser executada.

Durante o desenvolvimento deste projecto, adquiriu-se um conhecimento das vantagens e desvantagens de utilizar como base aplicações código aberto para atingir um determinado objectivo. Adquiriu-se ainda conhecimento das limitações de desenvolver aplicações para dispositivos de pequenas dimensões e, além disso, também se adquiriram conhecimentos sobre diversas tecnologias e aprofundaram-se conhecimentos sobre outras.

Relativamente às vantagens dos sistemas de sugestão automática de conteúdos, tópico em que esta dissertação se insere, podemos concluir que provavelmente no futuro irá ter uma grande difusão devido à sua flexibilidade, às suas vantagens, à inovação que trará ao mercado e à sua capacidade de ir ao encontro das características e preferências do utilizador.

Anexo A

Ficheiro de Configuração da Aplicação Qt

Neste anexo encontram-se um exemplo do ficheiro de configuração da aplicação Qt e a tabela A.1 onde estão descritos os grupos deste ficheiro e suas principais configurações. Na secção 3.3.2.5 encontram-se descritas as principais configurações e as suas funções.

```
[%General]
mplayer_bin=mplayer
driver\vo=xv
driver\audio_output=pulse
use_screenshot=true
screenshot_directory=/home/fonseca/.config/smplayer/screenshots
dont_remember_media_settings=true
dont_remember_time_pos=true
audio_lang=
subtitle_lang=
use_direct_rendering=false
use_double_buffer=true
use_soft_video_eq=false
use_slices=true
autoq=6
add_blackborders_on_fullscreen=false
disable_screensaver=true
disable_video_filters_with_vdpau=true
use_soft_vol=true
softvol_max=110
use_scaletempo=-1
use_hwac3=false
use_audio_equalizer=true
global_volume=true
volume=100
```

```
26 mute=false
27 autosync=false
28 autosync_factor=100
29 use_mc=false
30 mc_value=0
31 osd=1
32 osd_delay=2200
33 file_settings_method=hash
34
35 [drives]
36 dvd_device=/dev/dvd
37 cdrom_device=/dev/cdrom
38 vcd_initial_title=2
39 use_dvdnav=false
40
41 [performance]
42 priority=2
43 frame_drop=true
44 hard_frame_drop=false
45 coreavc=false
46 h264_skip_loop_filter=1
47 HD_height=720
48 fast_audio_change=-1
49 threads=1
50 cache_for_files=0
51 cache_for_streams=1000
52 cache_for_dvds=0
53 cache_for_vcds=1000
54 cache_for_audiocds=1000
55 cache_for_tv=3000
56
57 [subtitles]
58 font_file=
59 font_name=
60 use_fontconfig=false
61 subcp=ISO-8859-1
62 use_enca=false
63 enca_lang=pt
64 font_autoscale=1
65 subfuzziness=1
66 autoload_sub=true
67 use_ass_subtitles=true
68 ass_line_spacing=0
69 use_closed_caption_subs=false
70 use_forced_subs_only=false
71 sub_visibility=true
72 subtitles_on_screenshots=false
73 use_new_sub_commands=-1
74 change_sub_scale_should_restart=-1
75 fast_load_sub=true
76 styles\fontname=Arial
```

```
77  styles\fontsize=20
78  styles\primarycolor=4294967295
79  styles\backcolor=4278190080
80  styles\outlinecolor=4278190080
81  styles\bold=false
82  styles\italic=false
83  styles\halignment=2
84  styles\valignment=0
85  styles\borderstyle=1
86  styles\outline=1
87  styles\shadow=2
88  styles\marginl=20
89  styles\marginr=20
90  styles\marginv=8
91  force_ass_styles=false
92  user_forced_ass_style=
93  freetype_support=true
94
95  [advanced]
96  color_key=20202
97  use_mplayer_window=false
98  monitor_aspect=
99  use_idx=false
100 mplayer_additional_options=
101 mplayer_additional_video_filters=
102 mplayer_additional_audio_filters=
103 log_mplayer=true
104 log_smplayer=true
105 log_filter=.*
106 verbose_log=false
107 save_smplayer_log=false
108 autosave_mplayer_log=false
109 mplayer_log_saveto=
110 repaint_video_background=false
111 use_edl_files=true
112 prefer_ipv4=true
113 use_short_pathnames=false
114 change_video_equalizer_on_startup=true
115 use_pausing_keep_force=true
116 correct_pts=-1
117 actions_to_run=
118
119 [gui]
120 fullscreen=false
121 start_in_fullscreen=false
122 compact_mode=false
123 stay_on_top=0
124 size_factor=100
125 resize_method=0
126 style=
127 show_motion_vectors=false
```

```
128 mouse_left_click_function=fullscreen
129 mouse_right_click_function=show_context_menu
130 mouse_double_click_function=fullscreen
131 mouse_middle_click_function=mute
132 mouse_xbutton1_click_function=
133 mouse_xbutton2_click_function=
134 mouse_wheel_function=2
135 wheel_function_cycle=30
136 wheel_function_seeking_reverse=false
137 seeking1=10
138 seeking2=60
139 seeking3=600
140 seeking4=30
141 update_while_seeking=false
142 time_slider_drag_delay=100
143 relative_seeking=true
144 language=
145 iconset=Nuvola_fsf
146 balloon_count=5
147 restore_pos_after_fullscreen=false
148 save_window_size_on_exit=true
149 close_on_finish=false
150 default_font=
151 pause_when_hidden=false
152 allow_video_movement=false
153 gui=DefaultGUI
154 gui_minimum_width=0
155 default_size=@Size(580 440)
156 hide_video_window_on_audio_files=true
157 report_mplayer_crashes=true
158 reported_mplayer_is_old=true
159 auto_add_to_playlist=true
160 add_to_playlist_consecutive_files=false
161
162 [tv]
163 check_channels_conf_on_startup=true
164 initial_tv_deinterlace=4
165 last_dvb_channel=
166 last_tv_channel=
167
168 [directories]
169 latest_dir=/home/fonseca
170 last_dvd_directory=
171
172 [defaults]
173 initial_sub_scale=5
174 initial_sub_scale_ass=1
175 initial_volume=40
176 initial_contrast=0
177 initial_brightness=0
178 initial_hue=0
```

```
179 initial_saturation=0
180 initial_gamma=0
181 initial_audio_equalizer=0, 0, 0, 0, 0, 0, 0, 0, 0, 0
182 initial_zoom_factor=1
183 initial_sub_pos=100
184 initial_volnorm=false
185 initial_postprocessing=false
186 initial_deinterlace=0
187 initial_audio_channels=2
188 initial_stereo_mode=0
189 initial_audio_track=1
190 initial_subtitle_track=1
191
192 [mplayer_info]
193 mplayer_detected_version=1
194 mplayer_user_supplied_version=-1
195
196 [instances]
197 use_single_instance=true
198 connection_port=8000
199 use_autoport=true
200 temp\autoport=40380
201
202 [floating_control]
203 margin=0
204 width=100
205 animated=true
206 display_in_compact_mode=false
207 bypass_window_manager=true
208
209 [history]
210 recents=
211 recents\max_items=10
212 urls=@Invalid()
213 urls\max_items=50
214
215 [filter_options]
216 deblock=vb/hb
217 denoise_normal=
218 denoise_soft=2:1:2
219 noise=9ah:5ah
220 volnorm=1
221
222 [default_gui]
223 video_info=false
224 frame_counter=false
225 fullscreen_toolbar1_was_visible=true
226 fullscreen_toolbar2_was_visible=true
227 compact_toolbar1_was_visible=false
228 compact_toolbar2_was_visible=false
229 pos=@Point(0 0)
```

```
size=@Size(706 459)
toolbars_state=@ByteArray(\0\0\0\xff\0\0\x11\xfa\xfd\0\0\0\x1\0\0\0\x3\0\0\0\0\0\0\0\xfc\x1\0\0\0\x1\xfb
actions\toolbar1=play_prev, nostar_iptv, star1_iptv, star2_iptv, star3_iptv, star4_iptv,
    star5_iptv, play_next, separator, compact, fullscreen, separator, screenshot,
    separator, show_file_properties, show_playlist, show_preferences, state_iptv,
    separator
actions\controlwidget=play, pause_and_frame_step, stop, separator, rewindbutton_action,
    timeslider_action, forwardbutton_action, separator, fullscreen, mute,
    volumeslider_action
actions\controlwidget_mini=play_or_pause, stop, separator, rewind1, timeslider_action,
    forward1, separator, mute, volumeslider_action
actions\floating_control=play, pause, stop, separator, rewindbutton_action,
    timeslider_action, forwardbutton_action, separator, fullscreen, mute,
    volumeslider_action, separator, timelabel_action

[base_gui_plus]
show_tray_icon=false
mainwindow_visible=true
trayicon_playlist_was_visible=false
widgets_size=102
fullscreen_playlist_was_visible=false
fullscreen_playlist_was_floating=true
compact_playlist_was_visible=false
ignore_playlist_events=false

[playlist]
repeat=false
shuffle=false
auto_get_info=true
recursive_add_directory=false
save_playlist_in_config=false
play_files_from_start=true
automatically_play_next=true
row_spacing=-1
latest_dir=/media/FAT32/Filmes

[playlist_contents]
count=1
item_0_id=1
item_0_author=
item_0_filename=file:///media/FAT32/Filmes/teste.avi
item_0_duration=5562.77
item_0_name=file:///media/FAT32/Filmes/teste.avi
current_item=0
modified=false

[iptvSettings]
watchmode=Alone
defaultFeedback=6
serverURL=http://localhost:8080/IPTVServerCoreWebService/ServerCoreWSService
login=fsf
```

```
273 pass=1234
274 rememberme=true
275 autoautenticate=true
276
277 [actions]
278 open_file=Ctrl+F
279 open_directory=
280 open_playlist=
281 open_vcd=
282 open_audio_cd=
283 open_dvd=
284 open_dvd_folder=
285 open_url=Ctrl+U
286 close=Ctrl+X
287 clear_recents=
288 edit_tv_list=
289 jump_tv_list=
290 next_tv=H
291 previous_tv=L
292 tv_menu=
293 edit_radio_list=
294 jump_radio_list=
295 next_radio=Shift+H
296 previous_radio=Shift+L
297 radio_menu=
298 play=
299 play_or_pause=Media Play
300 pause=Space
301 pause_and_frame_step=
302 stop=Media Stop
303 frame_step=.
304 rewind1=Left
305 rewind2=Down
306 rewind3=PgDown
307 forward1=Right
308 forward2=Up
309 forward3=PgUp
310 set_a_marker=
311 set_b_marker=
312 clear_ab_markers=
313 repeat=
314 jump_to=Ctrl+J
315 normal_speed=Backspace
316 halve_speed={
317 double_speed=}
318 dec_speed=[
319 inc_speed=]
320 dec_speed_4=
321 inc_speed_4=
322 dec_speed_1=
323 inc_speed_1=
```

```
fullscreen=F
compact=Ctrl+C
video_equalizer=Ctrl+E
screenshot=S
multiple_screenshots=Shift+D
video_preview=
flip=
mirror=
motion_vectors=
postprocessing=
autodetect_phase=
deblock=
dering=
add_noise=
add_letterbox=
upscaling=
audio_equalizer=
mute=M
decrease_volume="9, /"
increase_volume="0, *"
dec_audio_delay=-
inc_audio_delay=+
audio_delay=
load_audio_file=
unload_audio_file=
extrastereo_filter=
karaoke_filter=
volnorm_filter=
load_subs=
unload_subs=
dec_sub_delay=Z
inc_sub_delay=X
sub_delay=
dec_sub_pos=R
inc_sub_pos=T
dec_sub_scale=Shift+R
inc_sub_scale=Shift+T
dec_sub_step=G
inc_sub_step=Y
use_ass_lib=
use_closed_caption=
use_forced_subs_only=
subtitle_visibility=V
show_find_sub_dialog=
upload_subtitles=
show_playlist=Ctrl+L
show_file_properties=Ctrl+I
show_preferences=Ctrl+P
show_mplayer_log=Ctrl+M
show_smplayer_log=Ctrl+S
faq=
```

```
375 cl_options=
376 tips=
377 about_qt=
378 about_smplayer=
379 play_next=>
380 play_prev=<
381 move_up=Alt+Up
382 move_down=Alt+Down
383 move_left=Alt+Left
384 move_right=Alt+Right
385 inc_zoom=E
386 dec_zoom=W
387 reset_zoom=Shift+E
388 auto_zoom=Shift+W
389 zoom_169=Shift+A
390 zoom_235=Shift+S
391 exit_fullscreen=Esc
392 next_osd=O
393 dec_contrast=1
394 inc_contrast=2
395 dec_brightness=3
396 inc_brightness=4
397 dec_hue=5
398 inc_hue=6
399 dec_saturation=7
400 inc_saturation=8
401 dec_gamma=
402 inc_gamma=
403 next_video=
404 next_audio=K
405 next_subtitle=J
406 next_chapter=@@
407 prev_chapter=!
408 toggle_double_size=Ctrl+D
409 reset_video_equalizer=
410 reset_audio_equalizer=
411 show_context_menu=
412 next_aspect=A
413 next_wheel_function=
414 show_filename=Shift+I
415 toggle_deinterlacing=D
416 osd_none=
417 osd_seek=
418 osd_timer=
419 osd_total=
420 denoise_none=
421 denoise_normal=
422 denoise_soft=
423 size_50=
424 size_75=
425 size_100=Ctrl+1
```

```
426 size_125=
427 size_150=
428 size_175=
429 size_200=Ctrl+2
430 size_300=
431 size_400=
432 deinterlace_none=
433 deinterlace_l5=
434 deinterlace_yadif0=
435 deinterlace_yadif1=
436 deinterlace_lb=
437 deinterlace_kern=
438 channels_stereo=
439 channels_surround=
440 channels_ful51=
441 stereo=
442 left_channel=
443 right_channel=
444 aspect_detect=
445 aspect_1%3A1=
446 aspect_3%3A2=
447 aspect_4%3A3=
448 aspect_5%3A4=
449 aspect_14%3A9=
450 aspect_14%3A10=
451 aspect_16%3A9=
452 aspect_16%3A10=
453 aspect_2.35%3A1=
454 aspect_none=
455 rotate_none=
456 rotate_clockwise_flip=
457 rotate_clockwise=
458 rotate_counterclockwise=
459 rotate_counterclockwise_flip=
460 on_top_always=
461 on_top_never=
462 on_top_playing=
463 toggle_stay_on_top=
464 dvdnav_up=Shift+Up
465 dvdnav_down=Shift+Down
466 dvdnav_left=Shift+Left
467 dvdnav_right=Shift+Right
468 dvdnav_menu=Shift+Return
469 dvdnav_select=Return
470 dvdnav_prev=Shift+Esc
471 dvdnav_mouse=
472 speed_menu=
473 ab_menu=
474 videotrack_menu=
475 videosize_menu=
476 zoom_menu=
```

```
477 aspect_menu=
478 deinterlace_menu=
479 videofilter_menu=
480 rotate_menu=
481 ontop_menu=
482 audiotrack_menu=
483 audiofilter_menu=
484 audiochannels_menu=
485 stereomode_menu=
486 subtitlestrack_menu=
487 titles_menu=
488 chapters_menu=
489 angles_menu=
490 programtrack_menu=
491 osd_menu=
492 quit=
493 show_tray_icon=
494 restore\hide=
495 pl_open=
496 pl_save=
497 pl_play=
498 pl_next=N
499 pl_prev=P
500 pl_move_up=
501 pl_move_down=
502 pl_repeat=
503 pl_shuffle=
504 pl_preferences=
505 pl_add_current=
506 pl_add_files=
507 pl_add_directory=
508 pl_remove_selected=
509 pl_remove_all=
510 pl_edit=
511 toggle_video_info=
512 toggle_frame_counter=
513 show_main_toolbar=F5
514 show_language_toolbar=F6
515 state_iptv=Shift+V
516 rate_iptv=Shift+B
517 nostar_iptv=
518 star1_iptv=
519 star2_iptv=
520 star3_iptv=
521 star4_iptv=
522 star5_iptv=
523
524 [iptv_gui]
525 video_info=false
526 frame_counter=false
527 fullscreen_toolbar1_was_visible=true
```

```
528 fullscreen_toolbar2_was_visible=false
529 compact_toolbar1_was_visible=true
530 compact_toolbar2_was_visible=false
531 pos=@Point(46 20)
532 size=@Size(537 378)
533 toolbars_state=@ByteArray(\0\0\0\xff\0\0\x11\xfa\xfd\0\0\0\x1\0\0\0\x3\0\0\0\0\0\0\0\xfc\x1\0\0\0\x1\xfb
534 actions\toolbar1=play_prev, nostar_iptv, star1_iptv, star2_iptv, star3_iptv, star4_iptv,
        star5_iptv, play_next, separator, fullscreen, screenshot, separator,
        show_file_properties, state_iptv, separator, close
535 actions\controlwidget=play, pause_and_frame_step, stop, separator, rewindbutton_action,
        timeslider_action, forwardbutton_action, separator, fullscreen, mute,
        volumeslider_action
536 actions\controlwidget_mini=play_or_pause, stop, separator, rewind1, timeslider_action,
        forward1, separator, mute, volumeslider_action
537 actions\floating_control=play, pause, stop, separator, rewindbutton_action,
        timeslider_action, forwardbutton_action, separator, fullscreen, mute,
        volumeslider_action, separator, timelabel_action
538 toolbar1iconh=30
539 toolbar1iconw=30
540 controlwidgeticonh=30
541 controlwidgeticonw=30
542 fullscreeniconh=30
543 fullscreeniconw=30
```

Nome do grupo	**Linhas no ficheiro**	**Descrição**
General	1-33	Neste grupo encontram-se configurações gerais relacionadas com o reprodutor de conteúdos.
drives	35-39	Neste grupo encontram-se as configurações relacionadas com os leitores de DVDs e CDs e métodos de reprodução dos mesmos.
performance	41-55	Neste grupo encontram-se configurações relacionadas com o desempenho da aplicação.
subtitles	57-93	Neste grupo encontram-se configurações relacionadas com as legendas da aplicação.
advanced	95-117	Neste grupo encontram-se configurações avançadas da aplicação.

continua na página seguinte

continuação da página anterior

Nome do grupo	**Linhas no ficheiro**	**Descrição**
gui	119-160	Neste grupo encontram-se configurações gerais relacionadas com a interface principal e com as funções que o rato irá efectuar.
tv	162-166	Neste grupo encontram-se configurações relacionadas com a funcionalidade do reprodutor de conteúdos para que este possa possuir uma lista de canais de TV e rádio.
directories	168-170	Neste grupo encontram-se configurações relacionadas com as últimas pastas utilizadas.
defaults	172-190	Neste grupo encontram-se definidos os valores iniciais que diversos campos tomam quando a aplicação inicia, tais como volume, o *zoom* e filtros.
mplayer_info	192-195	Neste grupo encontram-se informações sobre a versão do MPlayer.
instances	196-200	Neste grupo encontram-se configurações relacionadas com a capacidade de apenas ser possível executar uma instância desta aplicação.
floating_control	202-207	Neste grupo encontram-se configurações relacionadas com a barra que aparece quando se está no modo de ecrã inteiro.
history	209-213	Neste grupo encontram-se informações e configurações relacionadas com os últimos conteúdos reproduzidos.
filter_options	215-220	Neste grupo encontram-se configurações relacionadas com os filtros que se podem aplicar.
default_gui	222-235	Neste grupo encontram-se configurações relacionadas com aspecto da interface Default GUI.
base_gui_plus	237-245	Neste grupo encontram-se configurações relacionadas com aspecto de todas as interfaces principais.

continua na página seguinte

continuação da página anterior		
Nome do grupo	**Linhas no ficheiro**	**Descrição**
playlist	247-256	Neste grupo encontram-se configurações relacionadas com a lista de reprodução.
playlist_contents	258-266	Neste grupo são armazenadas as informações relacionadas com os conteúdos que estavam na lista de reprodução quando a aplicação foi terminada e a opção de guardar os conteúdos da lista estiver activa.
iptvSettings	268-275	Neste grupo encontram-se as configurações relacionadas com a comunicação com o servidor.
actions	277-522	Neste grupo encontram-se configurações dos atalhos de teclado associados às acções.
iptv_gui	524-543	Neste grupo encontram-se as configurações relacionadas com aspecto da interface principal criada.

Tabela A.1: Diferentes grupos do ficheiro de configuração da aplicação Qt.

Referências

[1] Android, [accessed] September 2009. [online] http://www.android.com/. 11

[2] ios, [accessed] September 2009. [online] http://www.apple.com/ios/. 12

[3] Java me, [accessed] October 2009. [online] http://www.oracle.com/technetwork/java/javame/overview/index.html. 14

[4] Javafx, [accessed] September 2009. [online] http://javafx.com/. 13

[5] Kxml, [accessed] November 2009. [online] http://kxml.sourceforge.net/about.shtml. 18

[6] Maemo, [accessed] September 2009. [online] http://maemo.org/. 9

[7] Mmapi, [accessed] October 2009. [online] http://developers.sun.com/mobility/apis/articles/mmapi_overview/index.html. 16

[8] Open handset alliance, [accessed] November 2009. [online] http://www.openhandsetalliance.com/press_110507.html. 11

[9] Symbian, [accessed] September 2009. [online] http://www.symbian.org/. 9

[10] What is android?, [accessed] September 2009. [online] http://developer.android.com/guide/basics/what-is-android.html. 11

[11] Adobe flash lite, [accessed] May 2010. [online] http://www.adobe.com/products/flashlite/. 14

[12] Dandelion player, [accessed] March 2010. [online] `http://wiki.forum.nokia.com/index.php/Open_source_Media_Player_-_Flash_and_Flash_Lite3_compatible`. 22

[13] gsoap, [accessed] June 2010. [online] `http://www.cs.fsu.edu/~engelen/soap.html`. 18, 24

[14] gsoap nokia, [accessed] June 2010. [online] `http://wiki.forum.nokia.com/index.php/Using_gsoap_for_web_services`. 32

[15] Jarpa, [accessed] May 2010. [online] `http://www.i2tecnologia.com.br/jarpa/`. 14, 21

[16] Meego, [accessed] June 2010. [online] `http://meego.com/`. 10

[17] Metalink, [accessed] August 2010. [online] `http://www.metalinker.org/`. 49

[18] moblin, [accessed] June 2010. [online] `http://www.moblin.org/`. 10

[19] Mplayer, [accessed] June 2010. [online] `http://www.mplayerhq.hu/design7/info.html`. 17, 23, 31

[20] People of lava - scandinavia, the world's first android tv, [accessed] November 2010. [online] `http://www.peopleoflava.com/television/scandinavia/`. 12

[21] Qt, [accessed] June 2010. [online] `http://qt.nokia.com/`. 15

[22] Qt android port, [accessed] June 2010. [online] `http://qt.gitorious.org/qt/android-lighthouse`. 15

[23] Qt-iphone project, [accessed] June 2010. [online] `http://www.qt-iphone.com/Introduction.html`. 15

[24] Smplayer, [accessed] June 2010. [online] `http://sourceforge.net/apps/mediawiki/smplayer/index.php?title=Main_Page`. 23

[25] Soap version 1.2 part 1: Messaging framework (second edition), [accessed] June 2010. [online] `http://www.w3.org/TR/soap12-part1/`. 18

[26] Torrents, [accessed] September 2010. [online] `http://www.bittorrent.org/beps/bep_0003.html`. 48

[27] Wetab, [accessed] October 2010. [online] `http://wetab.mobi/en`. 11

[28] Jasmin Blanchette and Mark Summerfield. *C++ GUI Programming with Qt 4*. Prentice Hall, 2 edition, February 2008.

[29] Jacqui Cheng. Smartphones lead mobile sales, android moves into no. 3 spot, August 2010. [online] `http://arstechnica.com/gadgets/news/2010/08/smartphones-lead-mobile-sales-android-moves-into-no-3-spot.ars`. 10, 21

[30] Alan Ezust and Paul Ezust. *Introduction to Design Patterns in C++ with Qt 4*. Prentice Hall, 1 edition, August 2006.

[31] Ibrahim Haddad. Introduction to the meego project, June 2010. [online] `http://wiki.meego.com/images/MeeGo_Introduction.pdf`. 10

[32] Gary Menezes. Symbian os, now fully open source, February 2010. [online] `http://www.watblog.com/2010/02/06/symbian-os-now-fully-open-source/`. 9

[33] João Rodrigues. IPTV Server with Automatic Channels Personalization. Master's thesis, University of Aveiro, Campus Universitário de Santiago, Aveiro, December 2009. 3

[34] Vlad Savov. Amd will contribute 'engineering expertise' to meego development project, November 2010. [online] `http://www.engadget.com/2010/11/15/amd-will-contribute-engineering-expertise-to-meego-development/`. 10

[35] Alex Zaharov-Reutt. Intel/nokia meego phones are no go until 1st half of 2011, October 2010. [online] `http://www.itwire.com/your-it-news/home-it/42365-intelnokia-meego-phones-are-no-go-until-1st-half-of-2011`. 11