富媒体技术在手机广播社交中的研究与应用①

于喜清[1,2], 孙建伟[2], 于 波[2], 李培军[1,2]

[1](中国科学院大学, 北京 100049)
[2](中国科学院沈阳计算技术研究所, 沈阳 110168)

摘 要: 3G 网络与智能机的普及, 带来了移动媒体应用的大变革. 本文旨在通过对富媒体技术的研究与分析, 论述其在基于 iOS 的社交平台——沈阳手机广播系统中的应用, 并通过音频队列服务和线程队列服务及缓存策略对富媒体的传输、呈现机制进行优化. 目前该手机广播系统已通过测试, 并在 AppStore 上线, 经实验与实践证明, 该设计方案在性能与用户体验方面能满足用户的移动社交需求.

关键词: 富媒体; iOS; 手机广播; 移动社交

Research and Application of Rich Media Technology in the Mobile Radio Social

YU Xi-Qing[1,2], SUN Jian-Wei[2], YU Bo[2], LI Pei-Jun[1,2]

[1](University of Chinese Academy of Sciences, Beijing 100049, China)
[2](Shenyang Institute of Computing Technology of Chinese Academy of Sciences, Shenyang 110168, China)

Abstract: The advent of 3G network and popularity of intelligent phones has brought about great changes in mobile media applications. Through the study of rich media technology, the paper aims at discussing the application of rich media on iOS social platform---Shenyang mobile phone radio system. We use audio queue services and thread queue services and caching policy to optimize the transmission and presentation mechanisms of the rich media resource. At present, the mobile phone radio system has passed testing and launched on the app store. As experiments and practice show, the design scheme can meet the user's mobile social needs in terms of performance and user experience.

Key words: rich media; iOS; mobile phone radio; MSNS

随着智能手机和移动互联网的发展, 第三方基于手机平台的媒体应用也进入发展的黄金阶段. 由于手机终端携带方便, 可操控性强, 观看或收听节目时受约束条件少, 满足了现代人快节奏生活方式的需要, “富媒体”概念也随之兴起[1].

富媒体将文本、图形、图片、音视频等多种媒体对象在时间和空间上进行有机结合, 提供了丰富的表现形式和交互能力[2]. 在客户需求多样化、操作要求简易化、交互设计人性化的今天, 富媒体技术已逐渐成为下一代手机媒体应用的必然发展趋势.

沈阳手机广播系统采用 Xcode 开发环境及运行 iOS 操作系统的 iPhone4s 作为测试平台, 将社交网络中常见的个人主页、微博交友、在线互动、私信聊天及收听广播等功能集于一体, 并结合富媒体技术, 为用户提供了一款界面美观、切换流畅、表现力丰富、交互性强的社交平台应用.

1 社交网络模型

本系统采用 C/S 框架模型, 其基本结构主要包括 iPhone 手机客户端、媒体服务器、ELGG 电台服务器、MQTT 服务器四部分, 如图 1 所示.

iPhone 手机用户通过 3G 或 WiFi 网络将包含多种媒体资源的信息与服务器交互[3]步骤如下:

① iPhone 客户端通过 MQTT 协议向 MQTT 服务器订阅话题并进行话题监听. 之后将包含音视频、图片、表情等的富媒体资源传送至 Rich Media 服务器,

① 收稿时间:2013-08-20;收到修改稿时间:2013-09-24

逻辑存储后分别转化成相应的 URL 返回给客户端. 其中富媒体图片会先纵横比缩放, 裁剪为适合 iPhone 手机显示的尺寸, 再通过 UIImageJPEG Representation 方式进行比例压缩后再传送; 音视频也会进行相应的编码压缩后传送, 富媒体语音传送方法会在 2.2 节中说明.

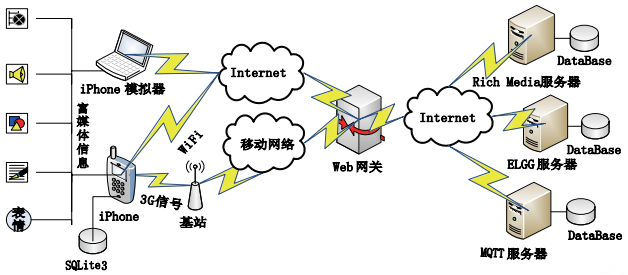

图 1 社交网络模型

② 客户端将 Rich Media 服务器返回的音视频、图片的 URL 及文本、表情等数据封装在相应的参数中, 并通过相应的 API 接口发送至 ELGG 电台服务器.

③ ELGG 电台服务器通过 KEY 认证机制, 接收授权用户发送的富媒体微博.

④ MQTT 服务器向订阅该话题的用户推送富媒体微博信息, 终端将富媒体微博的 Json 数据进行解析并刷新显示. 其中, 图片的呈现机制在 2.3 节中说明. 音视频信息只显示按钮图标及时长, 当用户点击音视频按钮, 触发 Rich Media 服务器 API 接口, 通过对应的 URL 向 Rich Media 服务器请求压缩后的音视频文件, 再通过音视频解码技术, 进而实现音视频播放.

2 手机广播系统客户端设计

2.1 系统客户端架构图

本系统使用基于 JavaScript 的轻量级数据交换语言 Json 进行信息的传送和接收; 同时结合 IMB 公司的 MQTT 即时通信协议, 完成消息的即时推送. 在实现了社交网络平台中收听广播、聊天交友、在线互动等功能的同时, 富媒体技术的嵌入使之更具动态性和交互性. 系统客户端的架构图如下所示:

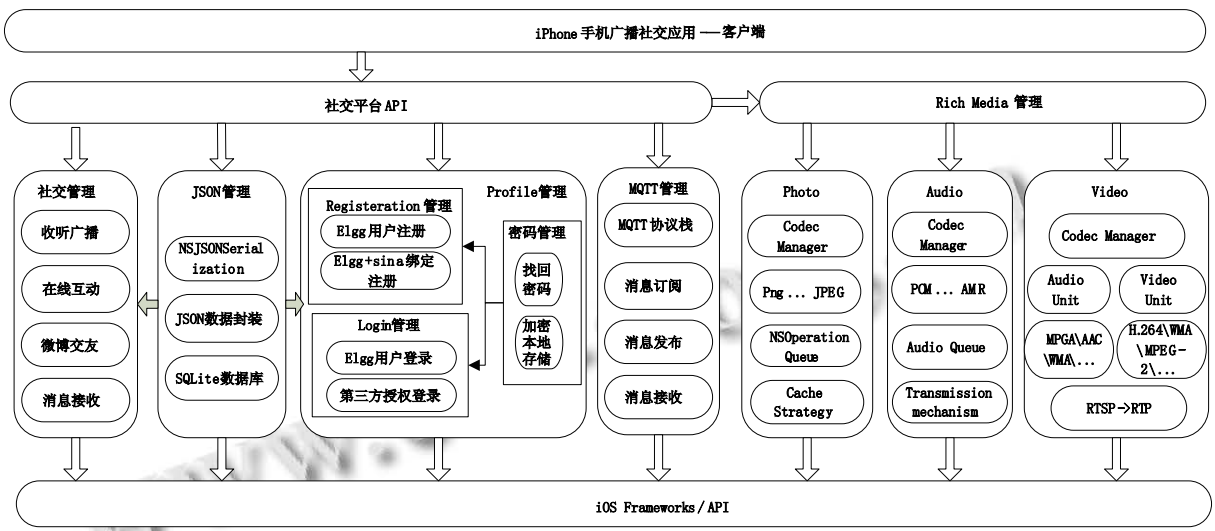

图 2 系统客户端架构图

2.2 富媒体语音模块设计

本系统在设计语音的录制和传送流程图如图 3 所示.

2.2.1 语音传送机制

在录制语音后的传送模式上, 本系统分为边录边传和边录边存方式. 分别是为适应语音发送和富媒体微博的传送而设计.

边录边传模式下, 当用户长按录音按钮超过 0.3 秒触发录音函数 UIGestureRecognizerStateBegan, 通过音频队列服务将语音编码并实时发送到 Rich Media 服务器, 当松开录音键时触发录音结束函数.

UIGestureRecognizerStateEnded, 设置结束标志. 随后调用 sendAudio API, 将 Rich Media 服务器返回的语音 URL 发送至 Elgg 电台服务器, 完成语音传送.

边录边存模式下, 考虑到用户操作的随机性, 语音录制过程中, 先将编码后的 AMR 格式的语音数据追加至本地 NSData 内存中缓存, 之后等待用户其他操

作，包括录音的撤销、重录以及视频录制、图片加载等行为. 操作完成后调用 sendRichMedia API，将富媒体信息发送至 Elgg 电台服务器完成富媒体微博传送.

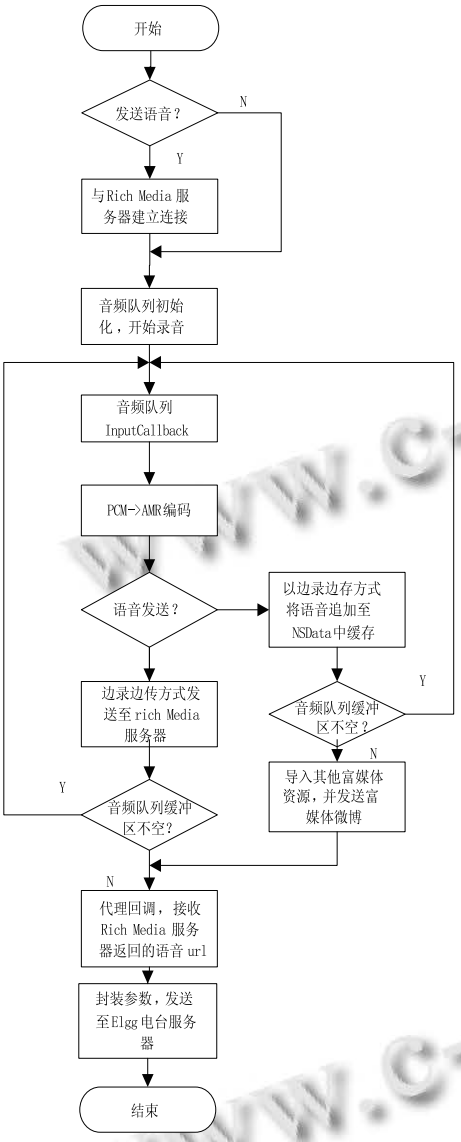

图 3　语音录制传送流程图

2.2.2 音频队列服务

在 iOS 设备上播放和录制音频，AVFoundation 框架中的 AVAudioPlayer 和 AVAudioRecorder 类用法简单，是苹果官方首推技术方案. 但由于其不支持流式，即在播放音频前，必须等到整个音频加载完成后，才能开始播放音频；录音时也必须等到录音结束后，才能获取到录音数据，这给应用造成了很大的局限性. 为了解决这个问题，本系统中使用 Audio Queue Services 来播放和录制音频[4]；并通过 Audio File Services 简化音频文件的处理. 其录制音频文件的原理框架图如下:

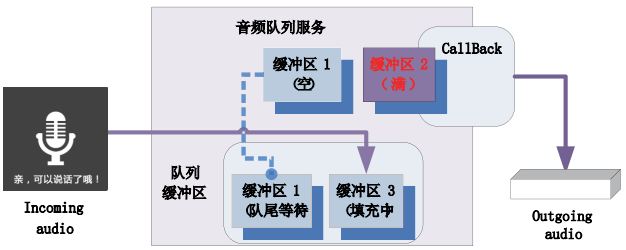

图 4　音频队列服务工作原理

如图 4 所示，其录制音频文件的主要步骤如下:

① 音频队列初始化，设置缓冲区数目为 3，大小为 16000，通过 AudioQueueEnqueueBuffer 对缓冲区进行排队. 调用 AudioQueueStart 函数开始录音并将音频数据填入第一个缓冲区中.

② 第一缓冲区填满时，会自动填充下一缓冲区. 同时，触发回调函数 AudioInput Callback.

③ 在回调函数中，首先调用音频编码库，对第一缓冲区的音频数据进行编码，由原始的线性 PCM 格式转码成 AMR 格式. 实验表明，通过编码可将录制的语音大小压缩至原来的 1/10 左右. 之后根据音频相关属性来判断是边录边传还是边录边存模式.

④ 将第一缓冲区重新放入音频队列队尾，以便重复使用. 重复步骤②.

2.3 富媒体图片模块设计

由于图片数据量较大且传输处理困难，在本地取图、网络图片加载等情况下，采用传统方式难以提供较好的用户体验. 本文针对网络图片加载、内存分配、释放及缓存策略等方面进行设计优化，使基于富媒体技术的手机广播系统的性能得到提升，健壮性得到加强，并具有更好的用户体验.

2.3.1 相机/本地图库取图

iPhone 提供的 UIImagePickerController 类，通过设定 sourceType 属性和委托对象来实现获取图片功能，其实例会以模态形式从窗口底部滑入，并全屏显示. 当取图后关闭 UIImagePickerController 视图并返回父视图时，会存在 UIImageView 显示图片概率失败问题.

以相机取图为例，当拍照结束后，照片会保存在内存中. 若此时内存占用率过高，会导致系统发生内存过低警告，此时若视图控制对象的视图不可见，则相应的

视图会被释放. 即父视图的视图控制对象会释放他的视图, 包括呈现照片的 imageView, 使 imageView=nil, 这时再为 imageView 设置图片是无效的.

本文通过为图片创建一个独立的存储对象, 而并非直接赋值给 imageView 来解决此问题. 当从 UIImagePickerController 对象返回父视图对象时, 在 viewWillAppear 函数中获取独立存储区的数据重置 imageView. 这样, 即使发生内存过低警告, 图片依然可以正常显示.

2.3.2 富媒体图片在 UITableView 中显示

本系统使用 UITableView 视图来呈现富媒体语音、图片等资源. 苹果公司利用 dequeueReusable-CellWithIdentifier:identifier 函数来标识 UITableView-Cell, 在不开销内存情况下实现了的 cell 重用机制, 做到显示和数据的分离. 但 UITableViewCell 的重用机制可能造成子视图重叠错乱; 同时 UITableView-Cell 下载图片过程中会阻滞系统 UI 主线程; 并且当图片过多导致占用的内存过大, 在快速滑动界面出现新的 cell 时, 绘制图片占有系统的资源过高, 阻滞滚动速度, 造成的内存过低甚至程序崩溃的情况依然不可避免.

1) 通过对 cell 重用机制稍作修改, 解决 cell 填充内容错乱的可能.

方法 1:

```
if (loadImageTableViewCell != nil)
{
    [oldImageview removeFrom: Superview];
[loadImageTableViewCell addSubview: newImageView ];
 }
```

方法 2: 自定义 CustomLoadImageViewcell 类, 将标签、图片等视图在该类中声明, 例如 IBOutlet UIImage *loadImage; 在显示图片的 UITableView 类所对应 storyboard 中, 拖拽 Table View Cell, 设置 Class 为 CustomLoadImageViewcell 类, 并将 identifier 设为 customLoadImageIdentity; 再把相应的视图添在该 Cell 上, 之后与自定义的 CustomLoad ImageViewcell 类内变量进行连接. 这样如果 loadImageTableView Cell = = nil, 则调用 loadImageTableViewCell = [[CustomLoadImageViewCell alloc] initWithStyle : UITableViewCellStyleDefault reuseIdentifier : @"customLoadImageIdentity"]; 并在之后操作中设置该 cell 相应的属性即可[5].

2) 针对加载网络图片阻滞系统 UI 主线程问题, 使用线程队列和缓存技术来优化[6].

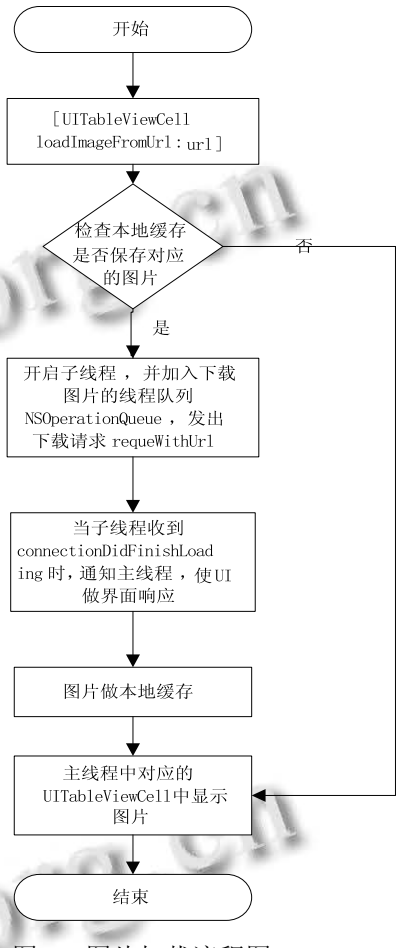

图 5 图片加载流程图

对于每个 cell 中下载图片的任务, 使用图片的 URL 作为标识, 新开线程来实现, 并通过 NSOperationQueue 来对各个线程任务进行管理和调度, 设置并发线程数、调整下载任务优先级, 比如对当前可见 Cell 的下载任务设置高优先级. 当图片加载成功, 首先通过 NSMutableDictionary 的键值对数据结构将图片存储到本地缓存. 同时通过 performSelectorOnMainThread 通知主线程, 触发 [UITableView reloadData], 将下载成功的图片显示在对应的 cell 中. 加载图片的流程图如图 5 所示.

其中, 当应用程序遇到 respondToMemory Warning 内存过低警告时, 只需通过[self.imageCache removeAllObjects]方式来清除本地图片缓存即可.

3 系统验证

基于 iOS 社交平台的沈阳手机广播系统目前包含 6 套直播节目. iPhone 手机用户可通过节目单了解当前收听的栏目, 还可以通过电台微博或新浪微博进行互动, 并支持听友间的实时在线交流. 通过测试, 在 iPhone 用户处于 WiFi 和 3G 环境下, 该系统具有良好的可靠性和 QoS 性能. 目前该手机广播系统已通过测试, 并在 AppStore 上线.

系统中有关语音发送和收听广播电台的展示界面如下所示:

图 6 界面展示图

4 结语

本文通过对富媒体技术在手机广播社交平台应用的研究, 实现了收听广播电台、与主持人在线交流互动等功能. 此外, 还允许用户建立自己的社交圈, 同感兴趣的听友进行微博、私信交友聊天, 并支持第三方授权登录功能. 本系统在语音传输、富媒体信息呈现、内存管理、通信超时及异常处理等方面进行的优化, 使其具备运行的流畅性和良好的用户体验, 为 iPhone 手机用户提供了更具个性化的社交服务. 随着移动互联网时代的到来, 该社交应用将会有更广阔的发展前景.

参考文献

1 杨鑫诚.基于 iOS 的多媒体播放系统设计与实现.电脑知识与技术,2012 ,(36):8784–8791.

2 张骥先.面向移动终端的富媒体技术研究[硕士学位论文].西安:电子科技大学,2010.

3 徐明,陈广宇.iOS 平台多媒体短信系统开发的设计模式研究.微电子学与计算机,2012,29(11):112–115.

4 唐杰. 移动终端的录音功能的研究与实现[学位论文].西安:西安科技大学,2006.

5 Vo K. Increase and Optimize UITableView Performance. Pro iOS Apps Performance Optimization. Apress, 2011: 39–58.

6 黄天柱,涂时亮.iOS 开发 UITableView 加载图片的内存管理.计算机系统应用,2012,(9):113–118.

(上接第 33 页)

10 Candelas J. Real-time collaboration of virtual laboratories through the Internet. Computers and Education, 2009, 13(7): 217–223.

11 Hoyer H, Andreas. A multi-user virtual-reality environment for a tele-operated laboratory. Education and Information Technologies, 2004, 7(10): 289–301.

12 Jacobson A. Virtual physics lab close to reality. Computing in Science and Engineering, 2003, 12(8): 145–156.

13 Chen BM. A web-based virtual laboratory on a frequency modulation experiment. Man and Cybernetics, 2001, 18(9): 121–130.

14 Desmeulles G. The virtual reality applied to biology understanding. Expert Systemswith Applications, 2006, 17(6): 29–35.

15 Dalgarno B, Bishop AG. Effectiveness of a virtual laboratory as a preparatory resource for distance education chemistry students. Computers and Education, 2009, 13(8): 71–82.