

嵌入式系统 MMS 发送简化过程研究与实现

Research and Implementation of Simplified MMS Sending Procedure for Embedded Systems

陈 里 焦继乐 邓拯宇 兰 清 (中南大学 信息科学与工程学院 湖南 长沙 410083)

摘要: 通过对 WAP 协议的分析,提出了建立连接、发送数据、断开连接组成的 MMS 简化发送方法。并提出了针对该过程的一种直接 PDU 发送简化实现。为需要 MMS 发送的低成本嵌入式系统提供了一个较好的解决方案。

关键词: 多媒体信息服务 无线应用协议 无线会话协议 无线事务协议 嵌入式系统

1 引言

多媒体信息(Multimedia Messaging Service, 以下简称 MMS)的工业标准是由 WAP 论坛和 3GPP 制定的^[1]。MMS 最大的特色就是支持多媒体功能, WAP 无线应用协议为载体传送视频短片、图片、声音和文字。现有的 WAP 协议版本功能十分复杂,本项目针对 MMS 发送,需要对未使用的现有 WAP 协议功能进行简化。

MMS、WAP 相关的开源项目主要有: MMSLib、MMSCClient 和 Kannel。它们的相关信息在表 1 中。

表 1 MMS 开源项目

项目名称	语言	栈类型	内容	相关项目	主页
MMS Lib	Java	客户端	MMS 客户端	JWAP	sourceforge.net/projects/mmslib
MMS Client	C	客户端	MMS 客户端	Kannel	sourceforge.net/projects/mmsclient
Kannel	C	服务器	WAP 网关	无	www.kannel.org

MMSLib: 该项目使用 JAVA 实现了一个 MMS 接收和发送的应用层,协议栈方面直接使用了著名开源项目 JWAP;不过 JAVA 语言效率较低,系统需要 JAVA1.2 以上环境支持,不利于在嵌入式系统下移植

与应用开发。

Kannel: 该项目为 WAP 网关实现,实现中包含了 WAP 服务器端协议栈。

MMSCClient: 该项目试图实现一个 MMS 发送与接收的客户端,尝试修改 Kannel 项目的 WAP 协议栈实现一个完整的 WAP 客户端协议栈。该项目从 2003 年 4 月以来,一直为 alpha 0.1 测试版,实际已经死亡。

综合上述几个开源项目的特点,本项目组在此之前选择了 MMSCClient 为基础进行开发,在此构架上添加了 MMS 应用层、UDP 访问层,去掉了 WTLS 层,修正项目的状态转换代码,更新了全部相关 PDU 错误^[2]。但是,MMSCClient 的构架是针对并发处理,多线程的服务器端构架,对于那些“小容量”嵌入式系统来说负担太重,移植的话需要 Linux 操作系统支持。

为进一步简化实现方法我们分析发现,发送过程可简化为:建立连接,发送 MMS 内容,断开连接三个步骤。更为重要的是这三个步骤当中的一些 PDU 相对固定,因此可以通过直接发送 PDU 的方法,实现单线程 MMS 发送。该简化实现可以摆脱对多线程的依赖,以及无关操作所需的存储消耗,更方便移植到各种嵌入式系统。

2 MMS发送时序图

协议中各状态之间的转换规则非常复杂,根据项

① 基金项目:国家自然科学基金(60634020)

收稿时间:2008-11-04

目的简化要求, 我们仅仅提取了一条捷径路线, 该路线不考虑暂停、继续等其它特殊情况。简化后的时序图如图 1-3 所示。

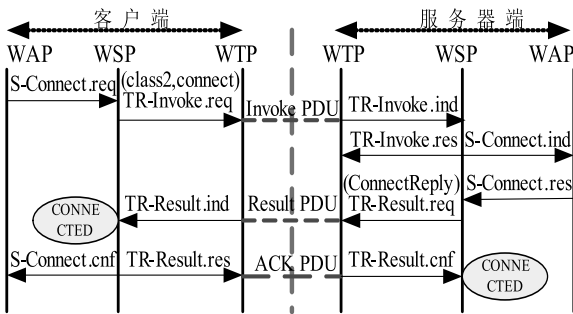


图 1 客户端与服务器建立连接时序图

协议栈的基本情况, 以及各层的协议数据单元 (Protocol Data Unit, 以下简称 PDU)的基本格式。如图 4 所示是各层 PDU 的总体组装原则, 各层都以紧邻的上一层数据作为服务数据单元 (Service Data Unit, 以下简称 SDU)。

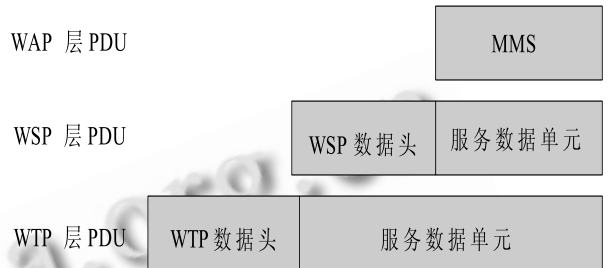


图 4 PDU 基本格式

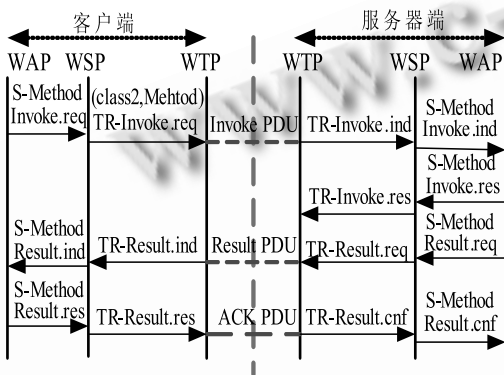


图 2 客户端向服务器发送 MMS 内容时序图

3.1 PDU 细节

多媒体信息服务 (Multimedia Message Service, 以下简称 MMS)也就是常说的彩信, 也就是 WAP 应用层的 PDU。关于 MMS 的制作与本文所关注的简化发送过程无关, 因此不作说明, 相关 PDU 制作可以参看文献[1][3]。

WSP 层 PDU 由两部分组成: WSP 数据头、上层的 SDU, 在 MMS 发送的应用中, 此处的 SDU 就是 MMS。在表 2 中列出了 WSP 层的数据格式。在我们的应用当中使用了: Connect、ConnectReply、Disconnect、Reply、Post, 其中 Connect、Post、Disconnect 是我们需要组装的, 其它的是需要解析的。

表 2 WSP 数据头格式

字节	1
名称	PDU类型	类型相关段

WTP 层 PDU 由两部分组成: WTP 数据头、上层 SDU, 此处的 SDU 就是 WSP 层 PDU。WTP 数据头可见表 3, 它主要由两个部分组成: 固定部分、变长部分。为简化起见, 我们的 MMS 发送应用不考虑变长部分。固定部分由: WTP 导引头、TID、类型相关段组成。WTP 导引头的信息很重要关系到: PDU 的方向、PDU 类型、是否为包组 (packets group) 传输、是否

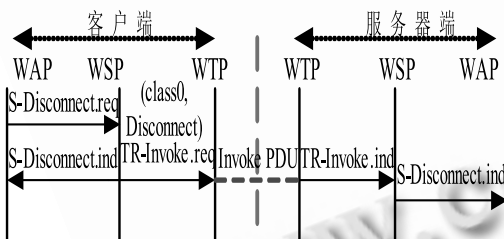


图 3 客户端断开连接时序图

我们可以直接组装在上面图中所需的数据包, 直接发往服务器端。由于所有的包可以直接组装、解析, 因此图 1-3 中客户端的所有 .ind、.cnf、.res 原语都可以不考虑, 此处只是为了让读者了解整个过程的完整性将这些原语绘制在了图中。

3 WAP 协议栈 PDU

在进行数据包组装之前, 我们先了解一下 WAP

重发包，具体格式见表 4，需要更详细的内容可以参考文献[5]。WTP 层中我们使用的 PDU 类型有：Invoke、Ack、Segmented Invoke、Result，其中前三种是我们需要组装，剩下的需要解析。

表 3 WTP 数据头格式

	固定部分				变长部分
字节	0	1	2
名称	WTP 导引头	TID	TID	类型相关段

表 4 WTP 导引头格式

位	7	6	5	4	3	2	1	0
名称	CON	PDU 类型		GTR	TTR	RID		

在发送 MMS 的过程有多种 PDU 需要组装、解析，这些都在下面的发送过程中详细介绍。

4 PDU 组装与发送过程

4.1 建立连接

在建立连接过程中，可以与服务器端进行能力协商，为简化程序我们仅仅对 Client-SDU-Size、Server-SDU-Size 两个能力值进行协商，这两个能力值关系到可以发送的彩信大小。保险起见我们将协商的值设置为 60K。建立连接过程：

- ① 建立网络环境，连接 MMS 网关 10.0.0.172，端口 9201；
- ② 发送连接包，进行建立连接、能力协商；
- ③ 接收服务器返回数据，并进行类型检验，提取服务器会话 ID；
- ④ 发送 ACK 包。

WSP 层需要组装一个 Connect 类型、版本为 1.0(本程序目前支持 WAP1.0，更高版本尚未测试)、Client-SDU-Size 与 Server-SDU-Size 能力为 60K 的 WSP 数据头。WTP 层应该组装一个 Invoke 型、非分段重组包、版本 1.0、Class 2 型事务。下面是一个参考数据头(为清晰起见在每一段参考数据头之前进

行了注释): unsigned char connect_pdu[] = {(WTP 数据头)0x0e,0x00,0x00,0x02,(WSP 数据头)0x01,0x10,0x0a,0x00,0x04,0x80,0x83,0xd4,0x60,0x04,0x81,0x83,0xd4,0x60 }。

在接收到了服务器返回的数据包之后，首先需要验证这是一个 Result PDU，之后判断是否是一个 ConnectReply PDU，如果无误则提取服务器会话 ID，该 ID 在断开连接时使用，否则，说明建立连接失败。此处没有检验服务器是否支持我们请求的能力，因为我们的实验已经验证过这个能力值。

无论发送成功与否，最后都要发送 ACK 包，因为这是一个 Class 2 型事务，下面是一个参考的 ACK PDU：

unsigned char ack_pdu[] = { 0x18,0x00,0x00 }。

4.2 断开连接

断开连接的过程非常简单，我们只需要组装一个 PDU，发送给 MMS 网关即可。WSP 层需要组装一个 Disconnect 型、会话 ID(建立连接中获得的会话 ID)数据头。WTP 层组装一个 Invoke 型、非分段重组包、版本 1.0、Class 0 型数据头。下面是一个参考数据头：unsigned char disconnect_pdu[] = {(WTP) 0x0e, 0x00,0x01,0x00,(WSP)0x05,0x00}。

4.3 MMS 数据发送

4.3.1 分段重组

MMS 的发送过程会相对复杂一些，这是因为 MMS 的数据量较大造成的。不但是我们需要组装复杂的数据包，而且必须对需要发送的数据进行分段重组处理。当成功建立连接之后，便可以开始进行 MMS 的发送了。如图 5 所示，简化发送过程的底层是 UDP 服务，而 UDP 服务的最大发送数据单元(Maximum Transmission Unit，以下简称 MTU)的缺省值大小在大多数情况下要小于 MMS 的值(Linux 系统)，因此不可能一次性将最后的 PDU 全部发送给 UDP 处理。所以我们需要实现 WTP 层的分段重组功能。在我们的系统中使用的缓冲大小是 1400，虽然该值比较保守但是符合很多嵌入式系统节省内存的要求。

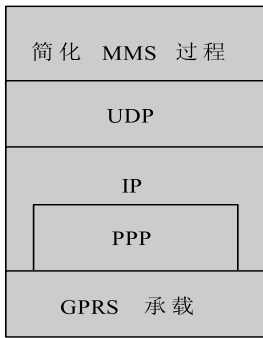


图 5 简化 WAP 与底层连接示意图

在描述 MMS 发送过程之前，我们需要先了解一下 WTP 层的分段重组相关概念。包组 (packets group)，顾名思义包组由数据包(packet)构成，图 6 就是一个由多个包组组成的消息的一般情况。图 6 中的包组由 L 个包组组成，每个包组中由不同数量的数据包构成。由此可见，分段重组的策略是很多的，而且可以实现选择性重发机制以提高传输效率。在我们的 MMS 简化发送过程中，使用一种最简单的策略：将整个 WSP 层 PDU 当成一个包组，收到错误信息后重新发送全部数据包。

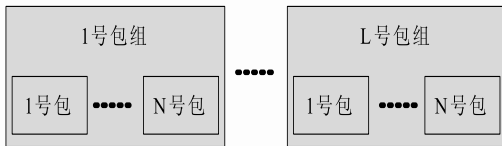


图 6 包组

4.3.1 分段发送过程

一个大数据 MMS 的典型发送过程如下：

- ① 首数据包组装，发送；
- ② [2, N-1]数据包组装，发送；
- ③ 最后数据包组装，发送；
- ④ 接收服务器数据，并进行类型检验；
- ⑤ 发送 ACK 包。

首发数据包组装：WSP 层需要组装一个类 HTTP/1.1 的 POST 请求将 MMS PDU 发送到 http://mmcs.monternet.com 也就是彩信中心。由于实现了类 HTTP/1.1 的机制，且客户发送数据到 MMS 网关是一个代理，不需要指定内容的长度。由于 WAP 协议栈对

常用 HTTP Header 使用紧凑格式描述^[4]，所以将该请求转换为字符串形式如下：

```
POST http://mmcs.monternet.com
Content-type:application/vnd.wap.mms-message
.....(MMS PDU 前部)
```

WTP 层需要组装一个 Invoke 类型、非结尾分段重组包、版本 1.0、Class 2 类型、0 号包数据头。参考数据头如下所示：unsigned char mms_1st_pdu[] = {(WTP)0x08, 0x00,0x01, 0x02, (WSP)0x60, 0x19,0x01,0x68,0x74,0x74,0x70,0x3a,0x2f,0x2f,0x6d,0x6d,0x73,0x63,0x2e,0x6d,0x6f,0x6e, 0x74,0x65,0x72,0x6e,0x65,0x74,0x2e,0x63,0x6f,0x6d,0xbe }。

第[2,N-1]数据包的格式可见表 5。由于分段重组功能属于 WTP 层，因此，仅需 WTP 层组装 Segmented Invoke 类型、非结尾分段重组包、Class 2 类型、累加包号的数据头。参考数据头如下所示：unsigned char mms_mid_pdu[]={0x28,0x00,0x01, 0x 05, }。

表 5 第[2,N-1]数据包格式

字节	0	1	2	3	4	1399
名称	WTP 导引头	TID	TID	包号	紧接 139 6 字节		
层	WTP				WSP (MMS)		

最后数据包的结构与表 5 的结构相似，不过 WTP 层组装的包应该指示是最后一个分段重组包。

发送完所有数据之后，接收来自服务器的响应信息。首先检验是否为 Result PDU，若是则检验是否为 Reply 包，然后检测状态码是否为成功。若满足上述要求则说明发送成功。

不论发送成功与否均应发送 ACK 包，内容同 4.1 中 ACK 包(TID 除外)。

5 实验与结论

实验将预先制作的 MMS 发送给简化 MMS 发送程 (下转第 171 页)

序,程序先启动 PPP 拨号过程,成功登录后,发起简化发送过程,利用手机终端作为接收端,检查是否能接收到之前发送的 MMS。通过在北京、上海、深圳、长沙的实验,所有测试终端均能接收到预定 MMS。实验环境如下:

嵌入式平台:三星 S3C44B0(ARM7), 16M SDRAM, 16M NAND Flash

软件:uClinux 2.4.23、PPP、Yaff 文件系统

GPRS Modem: CWT2000 深圳市盈科互动科技有限公司

MMS 制作:NokiaMMS 制作程序(开源软件 JAVA 语言实现)

接收终端:Nokia 6020, Samsung SGH-X458

通过实验,进一步验证了本项目的简化理论和实现方法的可行性。实验采用 C 语言实现,全部代码不到四百行,利用了 Socket 编程技术,可以很方便的在各种 Linux 平台上移植。较之现有完整的 WAP 协

议栈实现,对于那些仅需要 MMS 发送的嵌入式系统,本简化发送过程降低了系统的复杂度,占用资源少,效率要高于 Java 实现版本,以及多线程构架的 MMS Client 实现。

参考文献

- 1 Gwenaël Le Bodic. Multimedia Messaging Service. England:John Wiley & Sons Ltd, 2003.
- 2 李脉,陈里.用于 MMS 发送的简化 WAP 客户端协议栈设计与实现.长沙理工大学学报(自然科学版), 2008,5(1):84 - 89.
- 3 WAP Forum. MMS Encapsulation Protocol Specification.<http://www.wapforum.org/>, 05-Jan-2002.
- 4 WAP Forum. Wireless Session Protocol Specification. <http://www.wapforum.org/>,30-April - 1998.
- 5 WAP Forum. Wireless Transaction Protocol Specification.<http://www.wapforum.org/>,30-April - 1998.