基于 J2ME 的手机名片备份系统设计与实现

Design and Implementation of Mobile Phone Contacts Backup System Based on J2ME

刘云鹏 （浙江万里学院 计算机与信息学院 浙江 宁波 315100）
章中引 （浙江万里学院 商学院 浙江 宁波 315100）
李 瑾 （浙江万里学院 商学院 浙江 宁波 315100）

摘 要： 互联网与手机应用相结合逐步成为了网络发展的新趋势，在这种新趋势下，人们希望利用网络来进行备份手机中的名片信息(即联系人信息)。本文在分析了当前市场上手机名片备份软件的弊端后，提出一种客户端基于 J2ME、整个体系架构为轻型 J2EE 结构的手机名片备份系统，经过对手机模拟器和 N70、N73、N95、MOTO E6 等真实机型的测试，可以方便有效的对手机名片进行备份和恢复，有很强的实用性。

关键词： J2ME J2EE 手机名片 备份系统

1 引言

当今时代是个信息时代，也是一个手机的时代，现代生活对手机的依赖相当严重，人们很多时候都希望通过在手机上的各种系列操作来完成某项必要的功能。同时伴随着互联网的日益更新和发展以及 3G 技术应用的日益逼近，越来越多基于手机平台的应用更是在飞速发展，互联网与手机应用相结合逐步成为了如今互联网发展的新趋势。

对于手机用户来讲，名片信息(即联系人信息)的丢失和重新录入是最为严重的问题之一，需要尽量避免，用户希望借助于无线网络和 Internet，可以在用户与服务者之间，用户与用户之间，更快捷、更安全、更方便的交换信息，保证名片等重要、私有的信息在网络上进行有效的备份的恢复。目前这样的国内外手机软件有很多，比如 SmartvCard、ContactsTransfer、BackupContacts、ContactsManager、MICAT 等等。

SmartvCard 这一类软件，界面优美，管理功能强大，很方便的支持名片夹导出导入功能，支持批量操作，支持对分组的操作，支持多种传输方式，支持 cvp 格式、VCard 格式的文件，支持红外、蓝牙等方式发送 VCard 文件，而且可以通过 VCard 格式和其他的应用平台里交换联系人信息。但是这类软件网络功能支持比较差，仅仅支持 N72 等很少量的机型，而且试用期结束后，需要用 PC 机在网络上注册后才可以使用，这对于一些经常使用智能手机的中老年用户不方便，这类用户对计算机的使用不很熟练，但是由于长期使用手机，所以对手机的操作比较熟悉，他们希望可以通过手机来完成所有的相关操作。ContactsTransfer 这一类国外软件，效率较高，主要支持本地备份和短距离无线传输，机型更加受限，只能用于诺基亚 S60 第三版手机。值得一提的是北京美科互动科技有限公司的 MICAT 软件，真正做到了为手机和互联网平台之间搭建无缝对接桥梁的作用，其中的备份手机名片功能，可以将名片信息传到专属的私有空间(比如百度空间，搜狐博客等)中保存，非常有效、安全、私密。备份后可以通过手机在线查询、取回和进行多种管理，但是该软件为 SIS 文件格式，仅适用于 Symbian 操作系统的智能手机。

本文所提出的手机名片备份系统，采用 Java 技术开发，它的优势是良好的跨平台性和扩展性，不仅仅适用于 Symbian 操作系统，只要是任何支持 Java 虚拟机的手机平台都可以应用，而且对于目前流行的诺基亚 S60 第三版机型还避免了证书问题，即不必担心需要证书的数字签名限制，可以直接在手机上安装使用。

在论文下面的描述中将首先介绍系统的整体架构和功能流程,然后讲述关键技术的实现方法和相关的讨论,接着给出实验测试结果和性能分析,最后说明结论和提出论文的进一步研究开发工作。

2 系统设计与实现

2.1 系统架构与流程

整个软件系统的主要功能包括手机名片的本机备份和恢复、网络备份和恢复,开发重点是在客户端的处理,采用 J2ME 平台,服务端使用 Tomcat 服务器配合 SQL Server 数据库,主要采用 Servlet 和 Java bean 技术,所以整个系统的体系结构是一个轻型的 J2EE 架构。系统架构如图 1 所示。

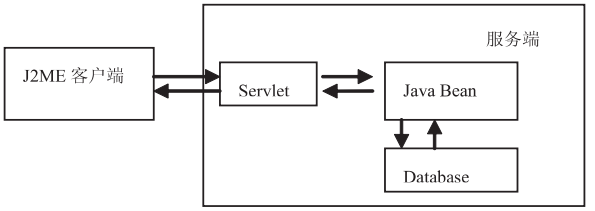

图 1　系统架构图

系统的两大主要处理流程是备份和恢复,如图 2 和图 3 所示。在图 2 中,对于网络备份还需要分段上传的步骤,在图 3 中,对于网络恢复还需要前面从服务端取数据的两个步骤。

2.2 关键技术分析与实现

2.2.1 手机名片信息的获取

名片信息,有时称为联系人信息、电话本信息或通讯录信息,属于手机用户的私有个人信息,在 J2ME 中访问名片数据是比较困难的事情,特别是中低端机型。通常,各终端厂商都有自己的 API 用于访问电话本。例如,MOTO 的 Phonebook 包。在 J2ME 的 jsr75 规范中,SUN 推出了 PIM 包,可访问名片信息等本地数据,目前大多数支持 JAVA 虚拟机的手机都支持 jsr75 规范。

PIM 的意思是 Personal Information Management(个人信息管理),主要是针对用户的个人重要信息,例如名片、提醒、日程等进行管理。PIM OP 定义了一系列的 API,提供了访问这些重要数据的方法和途径。

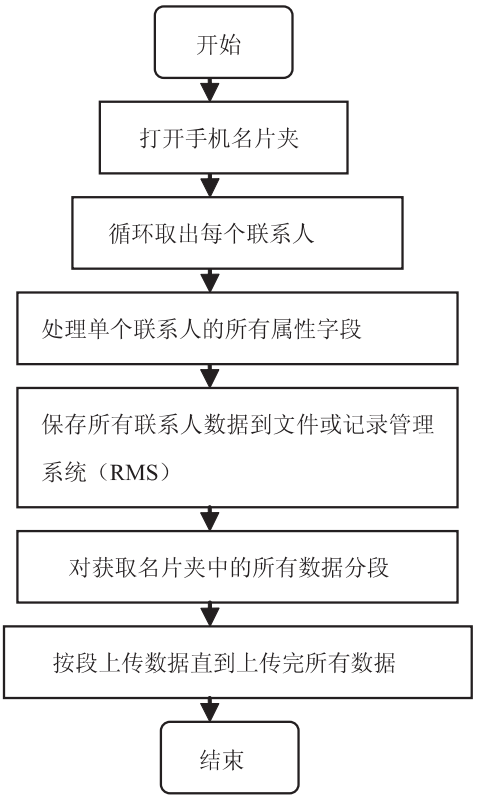

图 2　备份流程

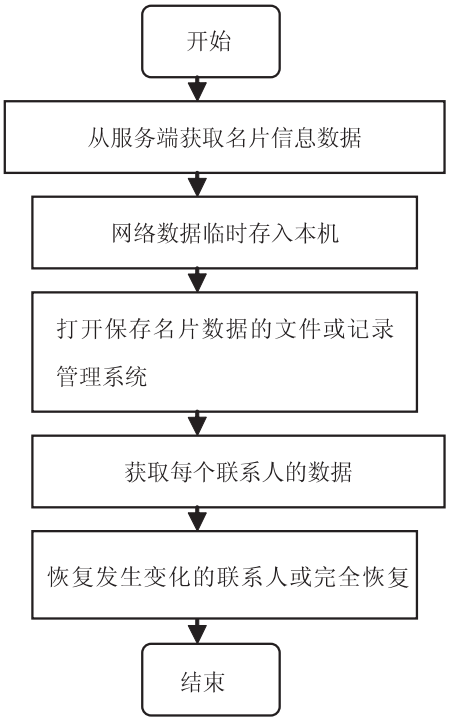

图 3　恢复流程

OP 的意思就是 Optional Package(可选包),它并不能提供一个整套的运行环境比如 MIDP,而是对 MIDP 的扩展,同时需要设备的支持。在使用 PIM OP 之前必须判断它是否可用,方法很简单,只是检查 microedition.pim.version 的属性值是不是 null。PIM 定义了三种信息类型,分别是 Contact list,Event list 和 ToDo list,名片的处理是针对 Contact list。

所有的 API 都在 javax.microedition.pim 包里面,可以使用提供的 API,来完成对名片信息的获取、分组处理、添加、删除、修改等各种处理。在 PIM list 中的数据称作 PIM item,可以把 PIM list 看作是容器,把 PIM item 看作是实体。本文要处理的实体是名片夹中的每个联系人,即 Contact 的信息,注意"联系人"中可用的字段跟设备实现是相关的,比如在某个机型中支持联系人的"传真"字段,可能在另一个机型中就不支持。因此,在使用某个字段的时候有必要调用它的 isSupportedField 来判断一下。PIM 规范中同样对字段中的基本数据类型进行了定义,例如在 Contact 里面的电话字段就是 String,生日字段就是 long 类型。

即便是同一个手机中的不同联系人,其保存的字段个数也可能是不一样的,比如张三仅仅保存了他的手机号码一个字段,而李四不但保存了手机号码还保存了家庭地址等字段。而且,不同类型的字段需要区别对待,经过测试分析,笔者发现在处理的时候可以将字段分成三种类型:第一种是姓名、地址等这种复合型字段,所谓复合型,就是一个大字段是由多个小字段组合而成的,比如地址字段是由国家名字段、省名字段、城市名字段、街道名字段等组成的,这种字段在保存时要循环处理每个小字段。第二种是可以重复多次的非复合字段,比如电话字段,一个联系人就可以记录多个电话值,再比如职位字段,可以记录多个不同的职位名称,这种字段在保存时要循环处理完同一个字段类型下所有的数据值。第三种是只能有一个的不能重复的非复合字段,比如备注字段等,如果支持该字段直接保存即可。

根据上面的分析,对于每个联系人的所有字段数据采用不定长保存策略,即对于本机支持且存在数据的字段进行保存,保存时根据上面分析的三种不同类型进行不同处理,对于本机支持但没有数据的字段,为了恢复时方便,可以采用若干个特殊符号(比如星号,井号等)来代替,这样的保存策略对于支持 jsr75 的手机可以直接适用,如果机型更换,代码在此方面不必做任何修改。

2.2.2 手机名片信息的恢复

在恢复的时候,会遇到这么几个问题。第一个问题是不论是从网络的服务端还是文件中获取到所有联系人数据,都是以字节流的形式存在,所有的联系人数据都存在于这个流中。我们需要从这个流中分离出每一个联系人的数据,由于在保存的时候,每个联系人数据按行保存,而且要求必须以换行符结束,所以循环处理字节流的时候,以换行符为标记就可以取出每一行。在本机备份中,如果数据是直接保存在记录管理系统(RMS)中,此问题不会出现,因为在 RMS 中就是按每个联系人分别保存的。

第二个问题是恢复的策略。策略一是将手机原有的联系人全部删除,使用备份数据进行完全恢复,如果联系人数量较大的话,一方面恢复时间会比较长,另一方面需要附加的操作也很繁琐,因为对于其他公司或个人开发的诺基亚手机软件,在未得到诺基亚公司授权之前(具体的授权需要经过一定的手续和费用),在对手机私有信息进行某些操作的时候,会进行询问,要求操作用户反馈后才可以继续执行。比如读所有联系人信息的时候,需要询问一次,用户确认允许后会全部读出,而对于添加联系人信息的时候,每添加一个人,就要询问一次,也就是说如果添加 100 个联系人,就要有询问提示 100 次,用户就要操作确认 100 次,会非常麻烦,而策略二可以将这种麻烦减少到最小的程度,但效率上会有一定的损失。具体方案是循环处理当前手机中联系人中的每一个,将联系人与恢复记录集中的每个进行比较,如果有相同的,表明不需恢复,因为目前就存在,同时将恢复记录集中的该记录做上标记;如果与要恢复的数据都不同,则表明该联系人还未备份,无需任何处理;最后恢复记录集中剩下的就是目前名片夹中不存在的联系人记录,全部恢复即可,除了重新安装系统和名片夹完全丢失的情况下,变化的联系人数据还是比较少的,方便了用户的动态恢复。

2.2.3 网络通信

(1)上传数据:采用 http 协议上传,有 post 和 get 两种方法。post 方法可以传输大量数据,而 get 方法由于受 url 长度的限制,一般不超过 1024 字节,无法传输大量数据,所以选用 post 方法。每次传输需要建立一个 HTTPConnection,使用字节流的方式。

经测试,post 每次传输的最大字节数为 2016,如果需要传递的数据超过 2016 字节,需要按 2016 分段,循环发送,数据分段到达服务器端后,由 Servlet 中的程序来处理重新组装。由于 http 连接是不保存的,所以每一次传送都要建立一次 http 连接。在此处的传输处理中,没有支持断点续传,但是由于一次所有的备份数据量并不是很大,按照名片夹中有 500 个联系人(联系人较多的情况)计算,一般才 10k 到 20k 之间的数据量,所以任何一次传输失败都要全部重新传输,服务端在清理残留数据后重新接收保存。对于超过 2016 字节数据量的传输,需要区分是首次传输还是首次以后的传输,因为当服务端知道是首次传输后,需要来删除老的备份数据或残留数据,从而将新的数据来进行缓冲保存。如何区分呢? 目前策略是首次传输数据前先发送用户名信息,因为系统要求成功登录后才可以访问服务端,也就是说此时用户已经通过了服务端验证,只是把用户名重新发送一下,并不需要密码,如果服务端接收并验证通过后,就知道这是一次新的数据传输了。

(2)下载数据:可以使用两种方法来下载数据。方法 1 还是使用 http 连接的方式,但是和上传数据有很大的不同,上传数据可以采用 post 或 get 方法, 对于下载数据,在获取了数据文件的具体路径和文件名后,可以直接使用输入流的方式连续下载数据文件的数据,具体代码如下:

```
HttpConnection conn = null;
conn = (HttpConnection) Connector.open("http://*.*.*.*.net/info.dat");
InputStream in = conn.openInputStream();
ByteArrayOutputStreambaos = new ByteArrayOutputStream();
int oneByte;
while ((oneByte = in.read()) != -1) {
    baos.write(oneByte);
}
```

此处假设"http://*.*.*.*.net/info.dat"为保存数据的完整文件路径名,下载完成后,最后数据就下载到了输出流变量 baos 流中。

下载数据前需要做的一点工作是,用户上传验证信息到服务端,通过验证后服务端根据用户名从数据库获取要下载的备份数据,并保存到一个确定路径下的临时数据文件中,以供下载。

方法 2 和上传数据的模式是类似的。首先通过 post 或 get 方法向服务端发送一个下载数据的请求,验证通过后,Servlet 将数据缓冲到内存,并以响应 post 或 get 的方式将数据直接传输到客户端,不需要中间的临时文件。

2.2.4 编码问题

UTF-8 是 UNICODE 八位交换格式的简称,是 UNICODE 传送格式,即把 UNICODE 文件转换成 BYTE 的传送流。UTF-8 是世界通用的语言编码,而且 UTF-8 编码在逐步成为主流,对于移动设备来说,它的字符串一般都使用 UTF-8 编码格式,所以在处理的时候要在适当的位置进行编码转换,否则可能导致不同的存储位置的编码格式是不一致的。

在本软件编码中,通过文件进行备份恢复数据或在上传下载数据的情况下,写入和读出数据一定要通过相关的方法将数据流或串转为 UTF-8 编码再处理;如果是对本机的记录管理系统(RMS)进行处理,则不需要人为的编码转换,系统会自动处理。转换代码如下所示。

备份到文件或上传数据前处理:str 为 String 类型,是要处理的数据。

```
byte[] utf_b = str.getBytes("UTF-8"));
```

从文件恢复或下载后数据处理:dis 是 InputStream 类型,是数据输入流,num 是 int 类型,是从流中读出的字节数。

```
byte word_utf[] = new byte[2048];
dis.read(word_utf, 0, num);
String str = new String(word_utf, "UTF-8");
```

3 系统测试

为了方便测试,避免每次添加联系人时进行的询问和确认,提高测试效率,可以先使用 WTK 中的手机模拟器进行测试。注意在设置中把存储空间和堆空间设置稍大一些,比如 5M 左右,防止缓冲区不够。模拟器上测试通过后,还必须经过真机测试,操作界面基本是一致的,主要测试功能,此处选择了当前流行度较高的 N70,N73,N95,MOTO E6 分别进行了测试,即有诺基亚机型,又有摩托罗拉机型,而且诺基亚机型还包括了 S60 第二版和第三版两个流行版本。

对于真机测试,首先需要对安装后的程序进行一些设置,对于网络连接一般设置为首次询问,也就是说首次连接无线网络时候需要用户确认,对于读取私有信息设置为首次询问,写私有信息只能设置为每次询问(软件未得到诺基亚授权前)。

测试的名片夹中联系人的数量分别选择 100 个、300 个和 600 个,因为根据一个初步的调研,一般普通用户的联系人数量大概在 100 到 300 左右,对于商人等特殊用户,联系人数量大概在 300 到 600 左右。大多数情况下,用户在保存联系人信息的时候,仅仅保存 1–2 个电话号码,而且保存 1 个电话号码的还占大多数,其他字段保存数据的情况很少,

但是随着手机使用的进一步发展,用户保存联系人的信息应该会越来越多。经过研究当前流行的智能机型:诺基亚、摩托罗拉、索爱,一个联系人的所有字段信息,在都存在有数据而且是最大数据量的情况下,一般不会超过 2Kbytes。

由于对本软件来讲,各个测试机型在处理速度上没有显著差异,所以在结果分析中取消了不同机型的比较,仅仅以 N70 的结果为例,选择 50% 的联系人只有 1 个电话号码,50% 联系人含有 2 个电话号码,网络服务是中国移动最普通的按流量包月服务,20Kbps 左右的上下行速度。性能测试结果如表 1 所示。

表 1 性能测试结果表

联系人数量	本机备份	本机恢复	网络备份	网络恢复
100	12s	25s	41s	35s
300	50s	132s	70s	170s
600	91s	600s	124s	673s

4 结论与进一步工作

从性能测试数据可以看得出,对于 300 以内联系人数量的备份和恢复,效率还是比较高的。如果联系人数量再增多,由于系统在设计的时候要求当前每个联系人都需要循环与备份数据中的所有联系人进行比较的原因,恢复时的效率有较明显的降低,可能会超过用户的忍耐度。但是对于大多数一般手机用户来讲,有很强的实用性,对于大数据量联系人的备份和恢复,可以将程序放在手机后台处理。

本系统的进一步研究开发工作是继续利用 Java 的轻便、安全和可移植性的优势在手机业务开发中的体现,做到不仅仅可以上传名片信息,而且可以将手机里产生的各种数据,如照片、短信、视/音频等文件上传至互联网。这种上传不单单是解决手机存储空间的问题,同时也实现了手机数据内容的交流与分享,创建了手机与互联网对话的功能,从而促进网站与手机无缝对接的新产业链。

参考文献

1 刘斌. Java ME 实用详解,北京:电子工业出版社,2007.
2 施铮,等,J2ME 技术参考手册,北京:电子工业出版社,2004.
3 胡虚怀,等,J2ME 移动设备程序设计, 北京:清华大学出版社, 2005.
4 京京工作室译. Java2 核心技术卷 1,北京:机械工业出版社, 2000.