第一章　序言

1.1 课题来源

目前流行的各大邮件客户端软件除了最重要的收发信件之外，功能越来越复杂，不过人们平常真正用到的功能很少，诸多功能尤其对于那些计算机知识相对缺乏的人来说，愈加显得太过于华丽而不太实用。有鉴于此，在理解 RFC 底层协议的基础上，本人开发了这个多种功能相对简朴实用的邮件客户端程序，简化了诸多不必要的功能。

1.2 电子邮件简介

电子邮件（简称 E-mail）又称电子信箱、电子邮政，它是一种用电子手段提供信息互换的通信方式。它是全球多种网络上使用最普遍的一项服务。这种非交互式的通信，加速了信息的交流及数据传送，它是一种简易、迅速的措施。通过连接全世界的 Internet，实现各类信号的传送、接受、存储等处理，将邮件送到世界的各个角落。到目前为止，可以说电子邮件是 Internet 资源使用最多的一种服务，E-mail 不只局限于信件的传递，还可用来传递文献、声音及图形、图像等不一样类型的信息。

电子邮件不是一种“终端到终端”的服务，是被称为“存储转发式”服务。这正是电子信箱系统的关键，运用存储转发可进行非实用时通信，属异步通信方式。即信件发送者可随时随地发送邮件，不规定接受者同步在场，虽然对方目前不在，仍可将邮件读取信件，不受时空限制。在这里，“发送”邮件意味着将邮件放到收件人的信箱中，而“接受”邮件则意味着从自己的信箱中读取信件，信箱实际上是由文献管理系统支持是一种实体。由于电子邮件是通过邮件服务器（mail server）来传递的。一般 mail server 是执行多任务操作系统 UNIX 的计算机，它提供 24 小时的电子邮件服务，顾客只要向 mail server 管理人员申请一种信箱账号，就可使用这项快递的邮件服务。

1.3 电子邮件的工作原理：

电子邮件的发送是通过电子邮件简朴传速协议（Simple Mail Transfer Protocol，简称 SMTP）来完毕的，电子邮件的接受是通过 POP3 协议来实现。它是 Internet 下的一种电子邮件通信协议。

电子邮件的基本原理，是在通信网上设置“电子信箱系统”，它实际上是一种计算机系统。系统的硬件是一种高性能、大容量的计算机。硬盘作为信箱的存储介质，在硬盘上为顾客分一定的存储空间作为顾客的“信箱”，每位顾客均有属于自己的一种电子信箱。并确定一种顾客和顾客可以随意修改的口令。存储空间包括寄存所收信件、编辑信件以及信件存盘三部分空间，顾客使用口令启动自己的信箱，并进行发信、读信、编辑、转发、存档等多种操作。系统功能重要由软件实现。

1.4 开发环境及运行环境

开发环境

AMD Athlom(TM), 512 内存，80G 硬盘

Microsoft（Ｒ）Windows XP Professional

Micosoft　Visual Studio 2023 （C Sharp）

Micosoft　Developer Network for Visual Studio.NET 2023

1.4.2 运行环境

Internet pentium 2 及以上处理器，32M 以上内存，4G 以上硬盘

Micosoft　windows 9X/NT 操作系统

800*600 或以上的屏幕辨别率

保证机器上安装有.Net FrameWork 1.0 或者以上版本

第二章 系统需求分析

2.1 系统功能需求分析

电子邮件系统需求实现的功能包括新建顾客的帐号，接受简朴邮件或带附件的邮件，发送简朴邮件或发送带附件的邮件，电子邮件编号，电子邮件分类管理，通信薄管理。为了使用通信薄，于是添加了对顾客资料的增长，修改，取消操作。

软件的总体架构

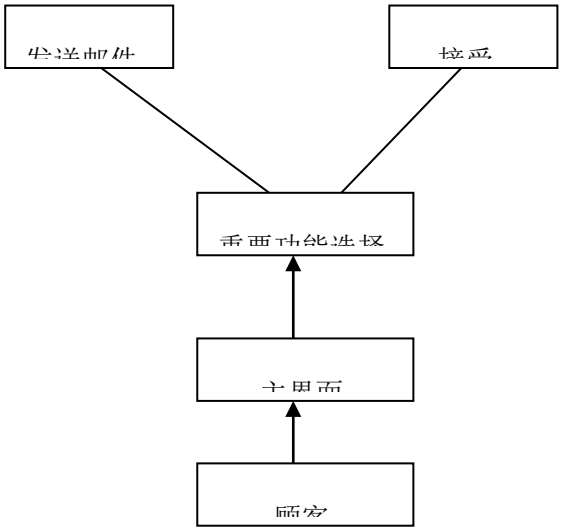

图 1　软件构架图

系统功能

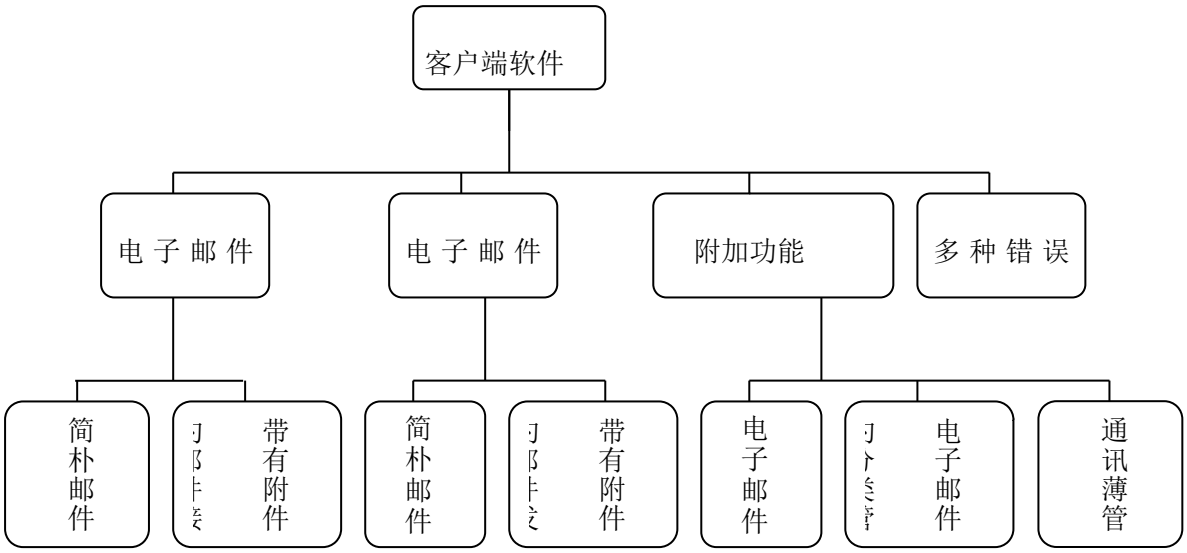

图 2　系统功能图

系统总体用例图

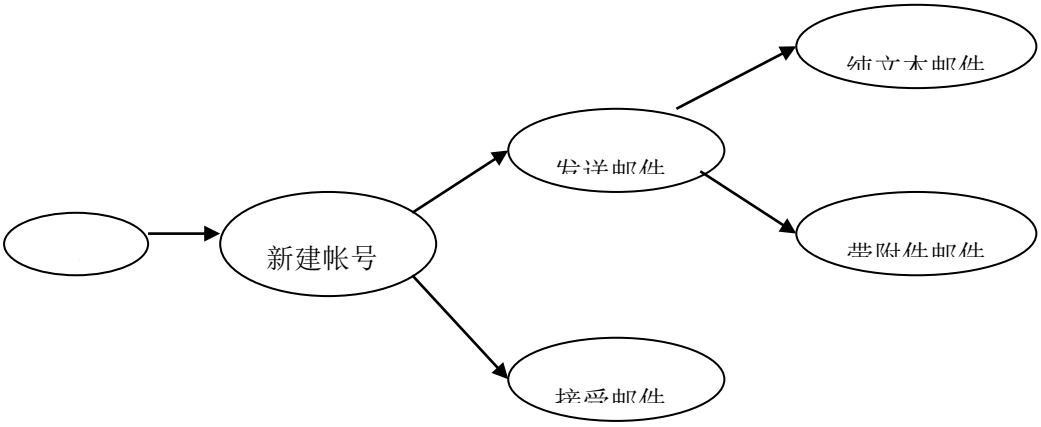

图 3 系统总体用例图

2.2 数据库需求分析

在对系统进行系统需求分析的基础上，可以得到系统在处理数据时会用到下面所示的数据项和数据构造：

1）顾客信息：帐号名称，顾客名，密码，电子邮箱地址，SMTP 服务器，SMTP 端口号，POP3 服务器，POP3 端口号。

2）通信薄信息：姓名，邮箱地址，　　号码，　　号，　　号码，通信地址。

第三章 系统设计

3.1 系统的流程设计

邮件客户端最重要的两个功能就是接受邮件和发送邮件，其中接受邮件的流程图如图 4 所示。从流程图中可以看出，接受邮件时首先要创立一种 TCP 连接到 POP3 服务器。假如连接不成功就退出执行，连接成功后再发送 USER 和 PASS 命令进行身份验证，身份验证通过后再通过 STAT 命令获得要接受的邮件数，当邮件数不小于 0 时，通过 RETR 命令逐一接受邮件。接受邮件完毕后，检查帐号中与否保留服务器上的邮件设置，假如是就不作任何操作，否则从服务器上删除已经接受的邮件。最终关闭连接。完毕邮件接受。

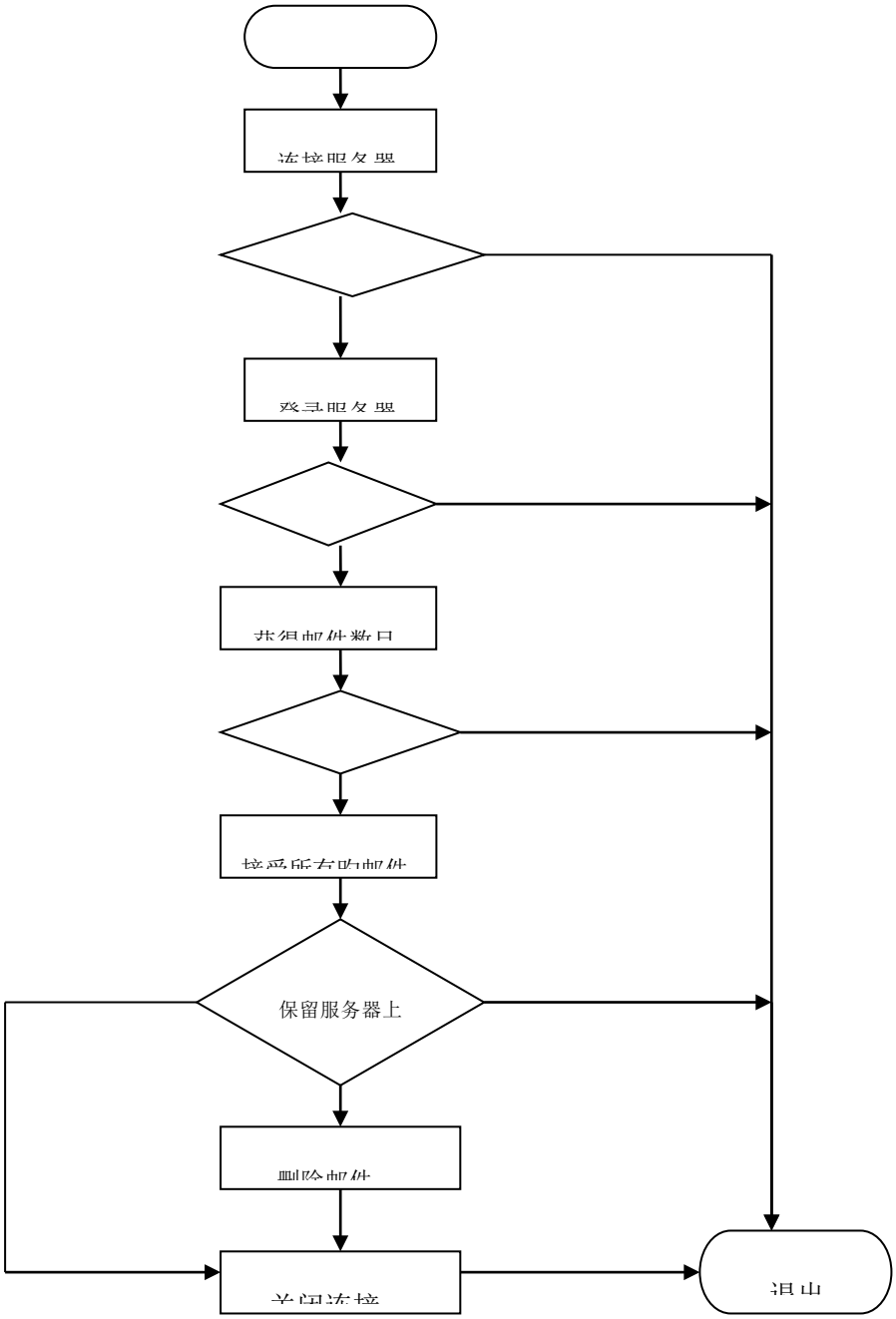

图 4 接受邮件流程图

发送邮件的流程图，先检查“发信箱”目录中与否有待发邮件，假如有就逐一发送这些邮件，流程图如图 5 所示。其发送过程，首先需要创立一种 TCP 连接，连接到 SMTP 服务器，假如连接不成功就退出程序。连接成功后发送 USER 和 PASS 命令进行身份验证。身份验证通过后发送邮件，假如发送成功就关闭连接，更新数据库，完毕邮件发送任务。

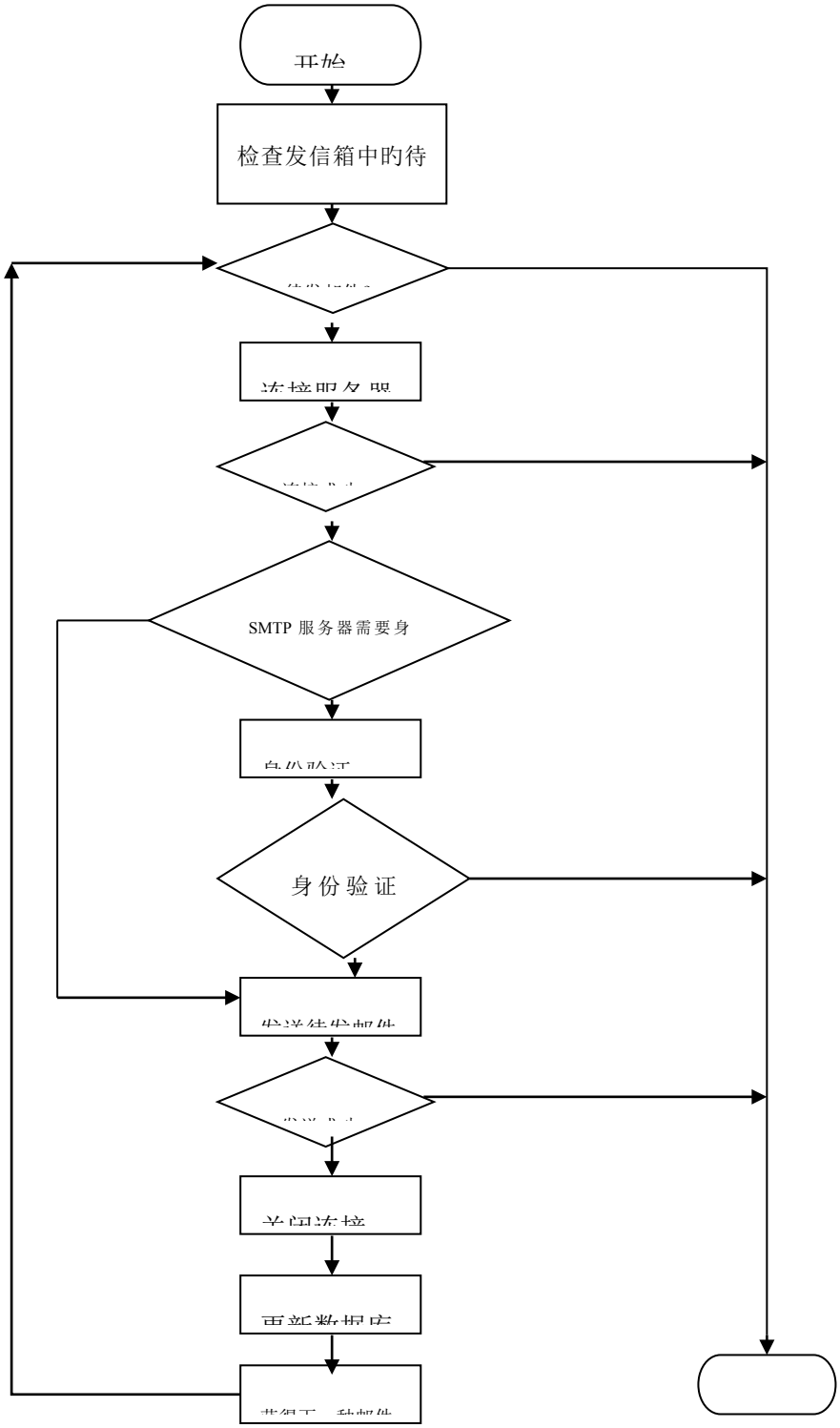

图 5 发送邮件流程图

3.2 SMTP 协议的研究

由于要开发的是邮件客服端程序，就不得不用到 SMTP 协议和 POP 协议。而我个人负责的是邮件发送功能的实现，因此就必然会波及到 SMTP（Simple Mail Transfer Protocol

）协议。SMTP 被用来在因特网上发送邮件，该协议规定了某些基本的命令和措施使客服端与服务器进行交互，以到达发送邮件的目的。

SMTP 简介

简朴邮件传播协议（SMTP）的目的是可靠高效地传送邮件，它独立于传送子系统并且仅规定一条可以保证传送数据单元次序的通道。

SMTP 的一种重要特点是它可以在传送中接力传送邮件，传送服务器提供了进程间通信环境（IPCE），此环境可以包括一种网络，几种网络或一种网络的子网。理解到传播送系统（或 IPCE）不是一对一的是很重要的。进程也许直接和其他进程通过已知的 IPCE 通信。邮件是一种应用程序或进程间通信。邮件可以通过连接在不一样 IPCE 上的进程跨网络进行邮件传送。更尤其是，邮件可以通过不一样网络上的主机接力式传送。

SMTP 模型

SMTP 设计基于以上通信模型：针对顾客的邮件祈求，发送 SMTP 建立于接受 SMTP 之间建立一种双向传送通道。接受 SMTP 可以是最终接受者也可以是中间传送者。SMTP 命令由发送 SMTP 发出，由接受 SMTP 接受，而应答则反方面传送。

一旦传送通道建立，SMTP 发送者发送 MAIL 命令指明邮件发送者。假如 SMTP 接受者可以接受邮件则返回 OK 应答。SMTP 发送者再发出 RCRT 命令确认邮件与否接受到。假如 SMTP 接受者接受，则返回 OK 应答；假如不能接受到，则发出拒绝接受应答（但不中断整个邮件操作），双方将如此反复多次。当接受者到所有邮件后会接受到尤其的序列，假如接受者成功处理了邮件，则返回 OK 应答。

SMTP 提供传送邮件的机制，假如接受方与发送方连接在同一种传送服务下时，邮件可以直接由发送方主机传送到接受方主机；或者，当两者在不一样一种传送服务下时，通过中继 SMTP 服务器传送。为了可以对 SMTP 服务器提供中继能力，它必须拥有最终目的主机地址和邮箱名称。

MAIL 命令参数是答复途径，它指定邮件从何处来；而 RCPT 命令的参数是转发途径的，它指定邮件向何处去。向前途径是源途径，而答复途径是返回途径（它用于发生错误时返回邮件）。

当同一种消息要发往不一样的接受者时，SMTP 碰到了向不一样接受者发送同一份数据的复制品的问题，邮件命令和应答有一种比较奇怪的语法，应答也有一种数字代码。在下面，例子中可以看到哪些使用实际的命令和应答。完整的命令和应答在第四节。

命令与应答对大小写不敏感，也就是说，命令和应答可以是大写，小写或两者的混合，但这一点对顾客邮件名称却不一定是对的，由于有的主机对顾客名大小写是敏感的。这样 SMTP 实现中就将顾客邮箱名称保留成初始时的样子，主机名称对大小写不敏感。

命令与应答由 ASCII 字母表构成，当传送服务提供 8 位子节传送通道，每 7 位字符对的传送，而最高位被填充为 0。当指定一般的命令或应答格式后，参数会由某些类似于语言的字符串表达出来，如“<string>”或“<reverse-path>”，这里尖括号表达这是一种类似于语言的变量。

3.3 SMTP 协议的命令和应答

SMTP 协议的命令

SMTP 命令定义了邮件传播或顾客定义的系统功能。它的命令是由<CRLF>结束的字符串。而在带有参数的状况下，命令自身由<SP>和参数分开，假如未带参数可以直接和<CRLF>

连接。邮箱旳语法格式必须和接受站点旳格式一致。

SMTP 旳应答码

对 SMTP 命令的响应是多样的，它确定了在邮件传播过程中祈求和处理的同步，也保证了发送 SMTP 懂得接受 SMTP 的状态。每个命令必须有且只有一种响应。

SMTP 响应由三位数字构成，其后跟某些文本。数字协助决定下一种应当进入的状态，而文本对人是故意义的。三位的响应已经包括了足够的信息，不用再阅读文本，文本可以直接抛弃或者传递给顾客。尤其的是，文本是与接受和环境有关的，因此每次接受到的文本也许不一样。正规的状况下，响应由下面序列构成：三位的数字，<SP>,一行文本和一种<CRLF>,或者也可以是一种多行响应。只有 EXPN 和 HELP，命令可以导致多行应答，然而，对所有命令，多行响应都是容许的。

REPLY CODES BY FUNCTION GROUPS 500 格式错误，命令不可识别（此错误也包括命令行过长）

第四章 RFC822

说到发送和接受邮件，就不得不提 RFC822 了。RFC822 的全称是“ARPA 因特网文信件格式的原则”（Standard for the Format of ARPA Internet Text Messages）。该原则提供了邮件内容的格式和有关语义。

4.1 RFC822 简朴简介

RFC822 规定的电子邮件内容所有由 ASCII 字符构成，就是一般所说的文本文献，因而原则将它称为 Internet 文本信件（Internet Text Messages）。

从直观上看，信件非常简朴，就是一系列由 ASCII 字符构成的文本行，每一行以回车换形符结束。

从组织上看，信件内容构造分为两大部分，中间用一种空白行（只有 CRLF 符的行）来分隔。第一部分称为信件的头部，包括有关发送方、接受方、发送日期等信息。第二部分称为信件的体部，包括信件内容的正文文体。信头是必需的，信体是可选的，即信体可有可无。假如不存在信体，用作分隔的空白行也就不需要。在信体中，也可以有用作分隔的空白行。这样设计的信件便于进行语法分析，提取信件的基本信息。

在 RFC822 中规定，信件体就是一系列的向收信人体现信息的文本行，比较简朴，可以包括任意文本。并没有附加的构造。信件头则具有比较复杂的构造，在下一小节中详述。

4. 2 信件的头部

信头一般格式

信头的构造比较复杂，信头由若干信头字段（header field）构成，这些字段为顾客和程序提供了有关信件的信息。要理解信头的构造就要弄清晰多种信头字段。

所有的信头字段都具有相似的语法构造，从逻辑上说，包括四部分，字段名（field name）紧跟冒号“:”（colon），后跟字段体（field body），最终以回车换形符（CRLF）终止。即

信头字段=字段名：字段体 CRLF

字段名必须由除了冒号和空格以外的可打印 US—ASCII 字符（其值在 33 和 126 之间）构成，大多数字段的字段名称由一系列字母，数字构成，中间常常插入横线符。字段名告诉电子邮件软件怎样翻译该行中剩余的内容。

字段体可以包括除了 CR 和 LF 之外的任何 ASCII 字符。不过其中的格式的空格，加括号的注释，引号和多行都比较复杂，此外，字段体的语法和语义依赖于字段名，每个类型的字段有特定的格式。

构造化字段和非构造化字段

每个字段包括的信息不一样，字段大体可以分为构造化字段和非构造化字段。构造化字段有特定的格式，由语法分析程序检测。Sender 字段就是一种很好的例子，它的字段内容是信箱，有一种离散的构造。

非构造化的字段具有任意的数据，没有固定格式。例如，Subject 字段可以具有任意的文字，并且没有固定格式。非构造化的字段数量较少，只有 Subject、Comments、扩展字段、非原则字段、IN—Reply 和 References 等。所有其他字段都是构造化的。

4.2.3 信头字段的元素

尽管 Email 信件的总体构造非常简朴，但某些信头字段的构造是很复杂的。下面简介某些大多数字段共有的元素。

（1）空白符

像其他文本文献同样，空白符包括空格符（ASCII 码 32）和制表符 TAB（ASCII 码 19）此外，行末的回车换形符 CRLF 也应算是空白符。使用空白符可以对字段进行格式化，增长它的可续性。例如，每个字段间用 CRLF 来分离，在字段内用空格来分隔字段名和字段内容。在 Subject 背面的冒号和内容之间插入空格字符，会使字段构造愈加清晰。在 Email 中。空白符的使用并没有固定的规则，但应当对的地使用，仅在需要时才使用空白符，以便接受软件进行语法分析。

（2）注解

注解是由括号括起来的一系列字符，例如，（这份礼品）。注解一般用在非构造化的信头字段中，没有语法语义，仅为人提供了某些附加的信息。假如在加引号的字符串中包括在括号中的字符，那是字符串的一部分，不是注解。在解释信件的时候，会将注解忽视，可以用一种空格字符替代它们，这样就什么也不会破坏。

（3）字段折叠

每个信头字段从逻辑上说应当是一种由字段名、冒号、字段体和 CRLF 构成的单一的行，但为了书写与显示的以便，增长可续行，也为了符号 1000/80 的行字符数的限制，可以将超过 80 个字符的信头字段分为多行，即对于比较长的字段，可以分割成几行，形成折叠。在成果化和非构造化字段中都容许折叠，第一行背面的行称为信头字段的续行。续行都以一种空白符开始，这种措施称为折叠（folding），例如标题字段 Subject：This is a test 可以表达为：

Subject：This is a test

反之，将一种被折叠成多行的信头字段恢复到它的单行表达的过程叫做去折叠，只要简朴地移除背面跟着空格的 CRLF，将折叠空白符 CRLF 转换成空格字符，就可以完毕折叠。在分析被折叠的字段的语法时，要把一种多行的折叠字段展开为一行，根据它的非折叠的形式来分析它的语法与语义。

（4）字段大小写

字段名称是不辨别大小写的，因此 Subject、subject 或 SUBJECT 都同样。不过字段名称大小写有习惯的常用形式，如主题字段的大小写形式一般为 Subject。字段体的大小写稍微复杂点，要视状况而定。例如 Subject 背面的字段体，其中的大写也许就是缩写的专用名词，不能改动。

（5）扩展字段

假如想在信头中加入 RFC822 中没有规定的字段，就需要创立非原则字段。措施非常简朴，只要在自定义的信头字段名的前面使用 X-前缀。RFC822 将这种措施称为扩展字段。实际上已经有许多扩展字段被广泛应用，但没有原则定义。例如：

X—LOOP 字段

X—LOOP 字段用来防止邮件的循环传送。过滤或邮件列表处理程序，可以给它处理的每个信件增长一种 X—LOOP 字段，后来就可以根据这个字段中具有的尤其值，判断一种信件与否被循环传送。假如确认邮件发生了循环，过滤或邮件列表处理程序就可以用不一样的方式处理该信件。

◆X—Mailer 字段

X—Mailer 字段用于指示什么样的程序产生了这个信件，它是使用最广泛的扩展字段。产生邮件的软件可认为所有发送的信件增长合适的 X—Mailer 字段，该字段不仅具有软件的名称，还包括软件的版本号。例如软件名为 Littlefox Mailer，版本为 V1.0，可以将“X—Mailer：Littlefox Mailer V1.0”加到邮件信头中去。

图 6 列出了某些在因特网电子邮件可以找到的一般关键字，以及使用它们的目的。

关键字	含义
From	发送方地址
TO	接受方地址
Cc	复制副当地址
Date	信息创立日期
Subject	信息主题
Reply—To	答复地址
X—Charset	使用的字符集（一般为 ASCII）
X—Mailer	发送信息所使用的软件
X—Sender	发送方地址的副本
X—Face	经编码的发送方面孔的图像

整个系统的关键是收发信件的操作，因此为了以便维护，后来的升级，故将这两个最重要的操作写成类库（.dll）的形式，以组件的形式加载到主程序中，并且其他的功能假如需要的话，也可以通过这样的组件的形式增长到主程序中。这也体现了 C Sharp 这一新的微软主推语言的以便和高效。并且这样做也以便了我们小组的程序的顺利结合。

第五章 系统实现

5.1 发送邮件类

SmtpMail 是发送邮件的关键，类名为 SmtpMail，从属于命名空间 MailSend。封装了发送邮件的详细实现措施，也是详细的 RFC 用代码实现的过程。而顾客通过详细的操作接口，接受与 SmtpMail 类通过交互操作来实现顾客发送信件的操作。

5.1.1 重要组员变量阐明

1）网络连接类及实例 TcpClient tc

为 TCP 网络服务提供客户端连接类 TcpClient 实例对象 tc。TcpClient 类提供了某些简朴的措施，用于在同步阻塞模式下通过网络来连接、发送和接受流数据。而实例化的过程也是连接 SMTP 服务器的过程。它的重载措施之一的两个参数一种为服务器名称字符串，另一种为服务器的埠。

2）提供用于网络访问的基础数据流及其实例 NetworkStream ns

此类提供访问网络的基础数据流的措施。其中最基本也是最重要的两个措施就是 Write（）和 Read()措施，至于参数不再次描述。

3）一维字符串数组变量 FilePath

此字符串数组重要用来寄存顾客选择的附件的绝对途径名，并在发送带附件的邮件时用到。

4)发送邮件所需的基本参数

例如用于 ESMTP 登录检查用的顾客名、密码，发送邮件需要的收信人，发信人地址以及主题等等在此不再陈说。

5.1.2 重要组员函数阐明

1）重载的构造函数 SmtpMail （）

此函数重要用于在初始化过程中，把顾客选择的附件的途径以参数的形式传给 FilePath。

2）添加附件的函数 AddAttachment

传给 FilePath 的途径，通过这样一种函数就可以循环的动态的添加到 LIST 接口的一种对象中了，以便后来在详细的实现过程中使用。

3）得到上传的附件的文献流 GetStream

由于在网络中的操作都是以网络流的形式来实现的，因此先将上传的附件转换成文献流，然后再用 Write 的措施把这些附件的文献流写入到网络中，来完毕发送附件的操作。详细实现代码如下所示：

```
///<summary>
    /// 得到上传附件的文献流
    ///</summary>
    ///<param name="FilePath">附件的绝对途径</param>
    Private string GetStream(string FilePath)
    {
        Try
```

```
{
    //新建文献流对象
    System.IO.FileStream FileStr =new System.IO.FileStream(FilePath,
System.IO.FileMode.Open);
    Byte[]by=new byte[System.Convert.ToInt32(FileStr.Length)];
    FileStr.Read(by, 0, by.Length);
    FileStr.Close();
    Return(System.Convert.ToBase64String(by));
}
Catch
{
    MessageBox.Show(“也许你要打开的文献的属性是只读的！”，“请检查权
限”);
    Return null;
 }
 }
```

4)将字符串编码为 Base64 字符串的函数 Base64Encode

由于 ESMTP 的 LOGIN 认证机制是采用 Base64 编码，当顾客发出 AUTHLOGIN 的命令后，服务器返回 334 的应答码等待顾客输入。假如身份确认后服务器返回 235 的应答码，否则返回失败信息。因此要将顾客名和密码转换 Base64 编码然后再发给服务器。此函数的作用就是把给定的字符串转换成对应的 Base64 编码的字符串。