中国民航大学

本科毕业设计（论文）

ACARS 航班监控系统设计

学生姓名：罗聪

专　　业：交通运输

学　　号：130440320

指导教师：张召悦

所属学院：空中交通管理学院

二〇一七年六月

中国民航大学

本科毕业设计（论文）

ACARS 航班监控系统设计

ACARS flight monitor and control system design

学生姓名：罗聪

专　　业：交通运输

学　　号：130440320

指导教师：张召悦

学　　院：空中交通管理学院

2017 年 6 月

创见性声明

本人声明：所呈交的毕业论文是本人在指导教师的指导下进行的工作和取得的成果，论文中所引用的他人已经发表或撰写过的研究成果，均加以特别标注并在此表示致谢。与我一同工作的同志对本论文所做的任何贡献也已在论文中作了明确的说明并表示谢意。

毕业论文作者签名：　　签字日期：　　年　月　日

本科毕业设计（论文）版权使用授权书

本毕业设计（论文）作者完全了解中国民航大学有关保留、使用毕业设计（论文）的规定。特授权中国民航大学可以将毕业设计（论文)的全部或部分内容编入有关数据库进行检索，并采用影印、缩印或扫描等复杂手段保存、汇编以供查阅和借阅。同意学校向国家有关部门或机构送交毕业设计（论文）的复印件和磁盘。

（保密的毕业论文在解密后适用本授权说明）

毕业论文作者签名：　　指导教师签名：

签字日期：　年　　月　　日　　签字日期：　年　　月　日

摘　要

【摘要】航空安全是民航运输界的永恒课题，随着民航运输业的迅猛发展，民航通信业务量快速增加，相应的航空安全受到愈加严重的挑战。因此对航班运行状态进行实时监控显得格外重要，而及时准确的信息来源就是保障监控飞机运行的前提。针对我国空中航班运行的现状，以数据链信息传输通信协议为依据，使用 C++编程语言搭建一个使用 ACARS 接收机接收 ACARS 报文并对报文解析提取出所需数据的平台。研究成果可以为专业人员提供人机的交互界面，从交互界面中系统使用人员能够观测到飞机实时数据，并将 ACARS 报文历史数据保存为文档，对提高飞行安全具有重要意义。

关键词：航空运输；报文解析；航班监控；数据提取

Abstract

Abstact: Aviation safety is the eternal subject of civil aviation transportation industry. With the rapid development of civil aviation transportation industry, civil aviation traffic increases rapidly, corresponding aviation security is more and more serious challenges.So the flight running status real-time monitoring is particularly important. Timely and accurate source of information is the premise of security surveillance aircraft operation. Aiming at the current situation of the air flight, based on data link information transmission communication protocol, set up through ACARS message ACARS message parsing method to extract the required data platform. Research results can be preserved for professional ACARS message history data, for the system using personnel in a timely manner to provide real-time data. It is of great significance to improve the flight safety

KeyWord: Air Transportation; Message Decode; Flight Control; Data Extraction

目　录

第 1 章　绪论

1.1　背景及意义

近年来民航运输业发展快速,民航通信业务量快速增加。目前航空管制通信使用的高频和甚高频话音通信系统,因为通信频道拥挤、易受到外界干扰等,航班的飞行安全和航班的正点率受到了严重的影响。地空数据通信系统拥有快速的数据传输率、不易干扰、准确性高的特点,已越来越受保障飞机安全和效益部门的重视。其通信方面的优越性日益凸显。随着对数据通信优越性认识的增加,世界上许多地区和国家已采用了一种面向字符的地空数据通信系统——ACARS(飞机通信寻址与报告系统)。[1]

本文为加强对地空数据链传输数据的处理，开发出一个专门用于处理 ACARS 报文数据的系统。使用 ACARS 数据接收机接收来自飞机下发的 ACARS 报文，并对报文进行筛查，如果为完整的符合 ARINC 618 标准格式的报文则将提取其中有用数据且把报文保存在使用对应所发报文航班的航班注册号为文件名的文档中，方便以后查阅。同时使用该系统的人员可以直观的操作界面查看报文数据，大大的方便了依赖 ACARS 数据的使用人员和公司。

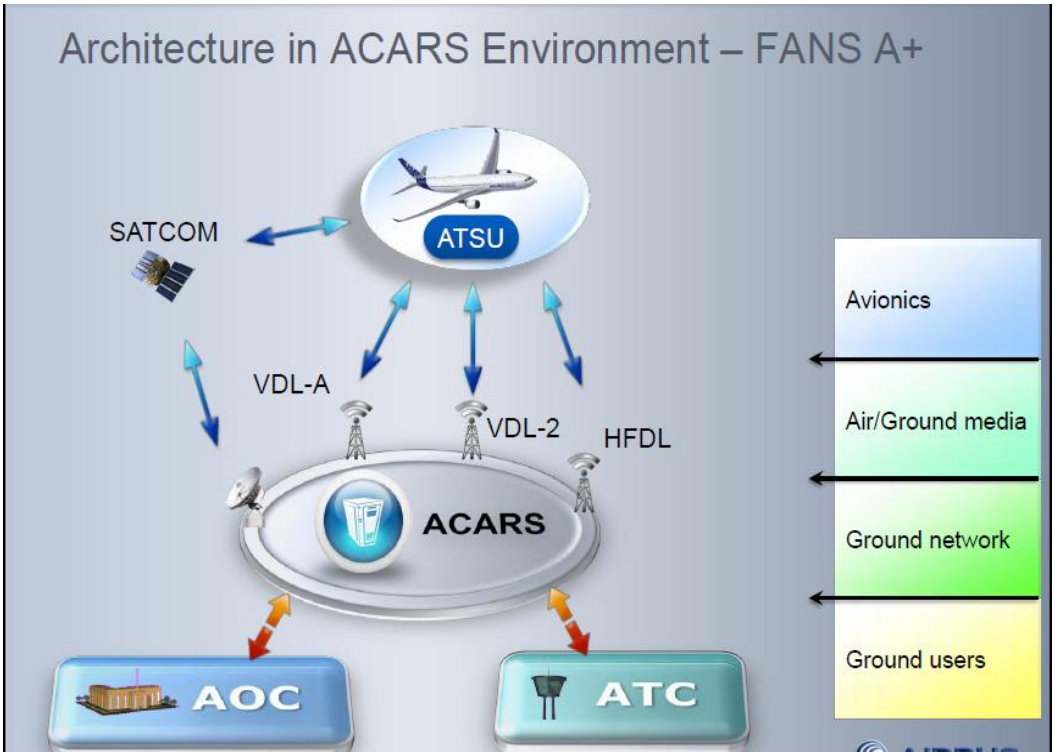


图 1.1ACARS 数据的收发过程

1.2 国内外研究现状

在国际民航新的通信、导航、监视和空中交通管理（CNS/ATM）的技术方案中，新一代方案施行的基础是创建一个新的航空电信网（ATN）。其中发展重点是研发地空数据链技术及其在飞机通信系统中的应用[2]。自从 20 世纪 80 年代后期，以美国为代表的国家和地区开始使 ARINC 公司开发的飞机通信寻址和报告系统（ACARS Aircraft Communication Addressing and Reporting System），世界各国民航业界也开始使用地空数据链传输通信数据，但是由于民航业的发展迅猛，特别是航班量激增与航班飞行量快速升高，地空数据通信的数据总量量也迅速增加，目前国内外一些地区的 ACARS 系统即将或已经处于过度饱和甚至溢满的状态，为了从根本上解决地空数据通信量快速升高带来的问题，民航界在地空数据通信基础上开发出了 VDL—M2、VDL—M3、VDL—M4 等新技术，以对目前的 ACARS 进行优化改进，从而增加地空数据通信的通信流量和提高通信效率。[3]

我国在 20 世纪 90 年代就已经开始了数据链地面网络的建设，现如今中国民用航空约有 97%的运行飞机已安装了 ACARS 相关设备；所以现阶段使用 ACARS 报告位置对跨洋航班和跨区域飞行航班及在偏远地区运行的航班实施航班监控和追踪已经具备硬件条件基础。同时为加强我国航空安全保障体系建设,提升民航飞行安全的监控能力，民航局基于现有成熟技术，决定开发出中国民航运行飞机的全球追踪监控系统。

1.2.1 ARINC 公司 ACARS 的应用

罗克韦尔柯林斯公司已成功为韩国机场公司装备了空对地航空通信网络，为该航空公司和韩国民航通信提供服务。

新的网络利用飞机通信训着报告系统（ACARS）支持航空公司运行控制（AOC）和空中交通管制（ATC）应用程序，比如起飞前放行许可（PDC）和数字自动终端信息服务（D-ATIS）。甚高频（VHF）数字链模式 2 能够比现有的甚高频系统网络快 10 倍。

该网络也已经成功在中国、泰国和日本等国家也实施运行。通过罗克韦尔柯林斯公司几十年的经验在开发和实施全球航空网络正帮助飞机飞行更安全、更高效的航路。

目前世界上很多国家正在从声音通信过度到数据通信，世界各地包括美国联邦航空管理局在内的新一代航空运输系统都提倡空域管理中进行这一变革。新一代的技术将使实时控制设备和机组人员之间的通信以数字数据传输取代模拟语音技术。

1.2.2 SITA 公司 ACARS 的应用

ICAO 和 IATA 正寻找更好的主动地飞机跟踪方法，SITA 公司也参与其中，包括提供一个已经应用在飞机上技术可行的解决方案。

ICAO 工作组可能提议政府要求航空公司追踪他们的飞机，美国以外的政府并

不是所有的都要求追踪其上空的飞机。IATA 正研究能被其成员尽快接受的解决方案。

航空公司没有追踪他们的飞机有两个主要原因。首先，因为他们依赖空中导航服务提供者的空域，此空域中他们拥有雷达监测系统以维持空域的安全。第二，因为在世界上大多数航空公司没有获得由 ANSPs 掌握的监测数据。

缺憾部分：

航空调度中心系统通常是被动的，他们依赖 ATC 的报告，这其中缺少的部分就是如何使这些系统化被动为主动。

这些报告并没有搜寻预期飞机通信和一旦发生突发事件没有提供警报，这正是 SITA 公司将要处理的内容。

SITA 公司正研究一种先进的端到端飞机跟踪解决方案以满足行业的迫切需要。

AIRCOM 服务是 SITA 航空运行中心系统最初开发用于管理与在飞机上的 ACARS 系统通信的服务。许多航空公司已经将此运用到实际工作中。

它根据来自飞机本身系统内的信息显示飞机的位置，也从由请求和响应产生的报告显示飞机的位置和空中交通管制发送的雷达监视数据提要。

SITA 正在为 AIRCOM 服务建立一种方法来识别 ATC 产生的消息，也是一种主动请求报告，所以航空公司系统可以通过编程来使用 ATC 使用的接口。

这不是一个简单的过程，AIRCOM 服务地面系统需要实现的接口用于接入 FANS 航电必须专门定制使 ACARS 信息支持 ATC 不仅仅局限于交换文本的应用程序。

一旦完成，只要飞机系统和航空公司选择好它的配置，就可以为航空公司提供飞机从起飞到着陆的位置。

SITA 将使航空公司跟踪飞机的方案符合任何 ICAO 工作组确立的 ACARS 解决方案。

SITA 提供超高 150 家航空公司、飞行超过 10000 架飞机通过其全球 AIRCOM 网络使用通信服务给他们的驾驶舱数据链系统应用 ACARS。

1.2.3 国外相关公司的应用

在过去的二十年里，CyberJet 已经建立了一个飞机信息系统与地面信息系统的大型专业接口。可能使用该应用的范围很大，包括飞行操作，运行控制，维护控制，工程和一些乘客服务。

从这个专项，CyberJet 为实现足够灵活的大多数需求已经设计了一个软件解决方案，实行 ACARS 成为主要任务并且不需要特定的开发成本。

该软件是独立的电信提供商和航电供应商，因此用户可以自由选择知道数据链接集成的供应商继续工作。例如 ACARS 网关有能力直接通过 Email 和 web 服务链

接到铱星提供商。

CyberJet 公司自 2003 年以来已经建立了专业技术发送数据到 FMC，该系统能够自动响应从飞机飞行计划和天气请求。FMC 上行链路导致减少燃油消耗和减少机组值班时间以优化机组的状态。

ACARS 网关软件被设计为一个高可用性解决方案，甚至由一个引擎制造商用于监控引擎

ACARS 网关有自己业务规则、飞机数据存储和提供报告的数据库，其中包括飞行时间、燃料信息、维护信息。航空公司也使用 SkyOps 和 SkyCrew，犹豫所有这些应用都是使用同一个数据库所以航班信息立即就可以得到，

1.4 本文的内容安排

1.3 本文所完成的工作

通过对飞机通信、寻址和报告系统（ACARS）数据编码与解码技术的研究设计并完成一个基于接收和解析实时 ACARS 数据的 ACARS 航班监控系统，使之能够配合相关设备完成对地空数据链报文的接收和解析工作。通过接收实时的航空数据，剔除在数据收发过程中产生的乱码，整理出完整的报文数据，并实时的显示在系统中。

本文主要是对 ACARS 报文的解码和编码的研究，实现解析 ACARS 报文，将报文中的数据单独提取出来，显示在系统中。

第 1 章，概述。主要介绍当前国内外的 ACARS 研究现状，阐述本文研究的背景和目的，以及简要说明论文需要达到的目的。

第 2 章，研究 ACARS 数据链及相关标准论述 ACARS 航班监控特性、参考 618、620、623 文件深入理解其间的内容与关系。

第 3 章，ACARS 航班监控系统设计，画出系统运行流程图，记录使用的关键函数，误码滤波的方法。

第 4 章，系统运行实验分析接收天津上空的 ACARS 信息，分析连续性、数据的准确率、特性分析系统运行效果。

第 5 章，结尾总结系统的工作成果，展望未来发展方向。

第 2 章 ACARS 数据链及相关标准

2.1 ACARS 基本结构及报文种类

ACARS(Aircraft Communication Addressing and Reporting System) 飞机通信寻址和报告系统是一种空地数字通信系统，飞机通过 ACARS 设备向地面数据中心发送 ACARS 报文报告飞机的位置、状态等信息。报文主要有以

下种类：OUT 报；OFF 报；ON 报；IN 报；位置报；DFD 发动机状态报；预计到达报；气象报；配载报。

ACARS 最初的设计目的主要用于飞行器运行状态的监控，为机务维修飞机提供数据依赖。通过 ACARS 地面维护人员可实时获取飞行器运行状态数据和性能数据，无需等待航班降落后才通过接口下载数据。

ACARS 系统主要分为机载设备和地面设备,其中机载设备由 ACARS 管理组件、综合显示控制器(CDU,通常也称 MCDU(多功能 CDUMDIU)、第三部甚高频通信系统和相关线路、相应软件等部件组成；地面设备由地面收发站和数据处理站、航空公司处理和分析终端等组成。

需要着重注意方面,ACARS 报文的主体部分通常只能包含 100 到 200 个字符的内容。少于或等于此长度的报文则可以一次性完成内容传输。如果长于 220 个字符的 ACARS 报文则只能分成多块多次传送，但是任何报文内容切分也不能多于 16 块。地面站也只是在收到所有报文块后才开始处理这条报文。

ACARS 报文有两种传输方式：如果有 VHF 地面基站的区域，使用 VHF 链路；如果没法建设地面 VHF 基站的区域，使用卫星链路。

ACARS 协议同时支持报文传输失败重传机制,也可以因为服务提供商改变的原因而再次发送数据。地面接收站一旦接收报文完成就可以通过地面线路(landilnes)将报文发送到数据链服务提供商(DSP)所有主机系统中。提供商再根据主机上的路由表将该报文发送到需求报文的航空公司或其他终端。[4]

ACARS 发送得比较多的是关于飞机状态和预计飞机运行时间的报文，航班的机载设备则可以接收地面的天气信息或者相关部门的信息通报，机组也可以自行编辑报文数据用于发送所要传输的信息。但是 ACARS 受限于频率只能进行视距传输，ACARS 依赖于地面基站和通信网络。

目前 ACARS 报文数据服务仍然不够便宜，单架飞行器一年所缴费用达到数十万级别，所以几乎所有的公司并不需要下发全部的飞行数据，而仅仅下发部分关键的飞行器的状态数据，因而目前 ACARS 目前数据传输率还不够高。

2.2 ARINC 解析标准

2.2.1 ARINC 618

–为 ACARS/CMU 和甚高频地面系统之间通信定义空/地通信协议

–定义由 ACARS/CMU 收发的 ACARS 信息的格式[5]

2.2.2 ARINC 620

–定义地对地通信协议

–定义一个服务提供商与航空公司或其他地面系统信息路由的信息格式。[6]

2.2.3 ARINC 623

–面向字符的空中交通服务应用程序[7]

2.2.4 ARINC 618 文件

这个规范包含之前定义的条款如 ARINC 546，566，716,597，724，和 724B 文件，这些特征包含了哪些多余的资料、在不同时期的更新、以及文本修正产生的矛盾。

本规范的目的是将这些多余的材料整合成一个文档用作描述 ACARS 空地环境的操作。本规范也理清在上述文档中出现的矛盾和现在新的将整个 ACARS 文档变成已经形成了多年状态的材料。在此中列出的条款包括适当的和原始的参考文档，因此继续使用规定系统确定的原始文档。

概述了全部的空/地块的格式，每个字段信息接了定义 ISO-5 字符集字符的授权使用范围。

定义了 ACARS MU 消息传递协议的优先级多模块处理

描述了甚高频 ACARS

描述了甚高频空地网络链接管理

定义了一个可选的功能 ACARS 允许用于数据通信甚高频收发器业能用于传输音频。

描述了空地卫星链路面向字符 ACARS 信息传输协议。

描述了空地高频数据面向字符 ACARS 信息传输协议。

描述了需要同时操作多个通信媒体的过程以及在需要时停止使用一个或多个媒体。

2.2.5 ARINC 620 文件

阐述特征数据链路服务提供商到数据链路用户所需的接口，本文档的目的是提供数据两路用户开发应用程序所需要的信息和尽可能提倡给中数据之间连接服务提供商的一致性和标准化，这个文档包含通常和具体指导数据链服务提供商与记载、地面用户之间的接口。

本文档从用户角度描述了输入数据、功能和输出数据链服务提供商系统。描述的功能必须提供一个需要做什么的功能而不是限制数据链路服务提供者网络内部实施的完整标准。有不同的消息格式或在不同的影响数据链用户接口服务提供商提供的功能，需要特别的说明。

2.2.6 ARINC 623 文件

此文件定义了可以在飞机上传输通讯寻址和报告系统数据链路传输消息的面向对象字符的空中交通服务的应用程序文本格式。ACARS 可以使用多个数据链路，包括但不仅限于甚高频、高频和卫星传输。此文件中定义的消息并不特定于任何数据链。

本文在面向字符的范围是有限的应用程序。面向位的应用程序定义的文件。依照 ARINC 622 规范这些应用程序。面向字符的消息的格式和内容不符合面向位的信息。

ACARS 数据链最初是提供给一个用户（通常是一个航空公司）与用户的飞机之间的联系。虽然空地协议对所有用户都是通用的，消息内容对于一个给定的应用程序并不是标准化的自用户控制两端的应用程序。

面向字符串的信息可能会直接发送给 ACARS 网络，然而，需要 ARINC 622 文件描述的内容辅助，然后，应用程序可以使用 ARINC 622 文件的技术处理产生的信息。为了将信息发送给 ACARS 网络，本文档所给信息格式必须根据 ARINC620 文件处理。

ARINC 620 文件不重复本文档任何文本格式仅仅描述它的包膜。

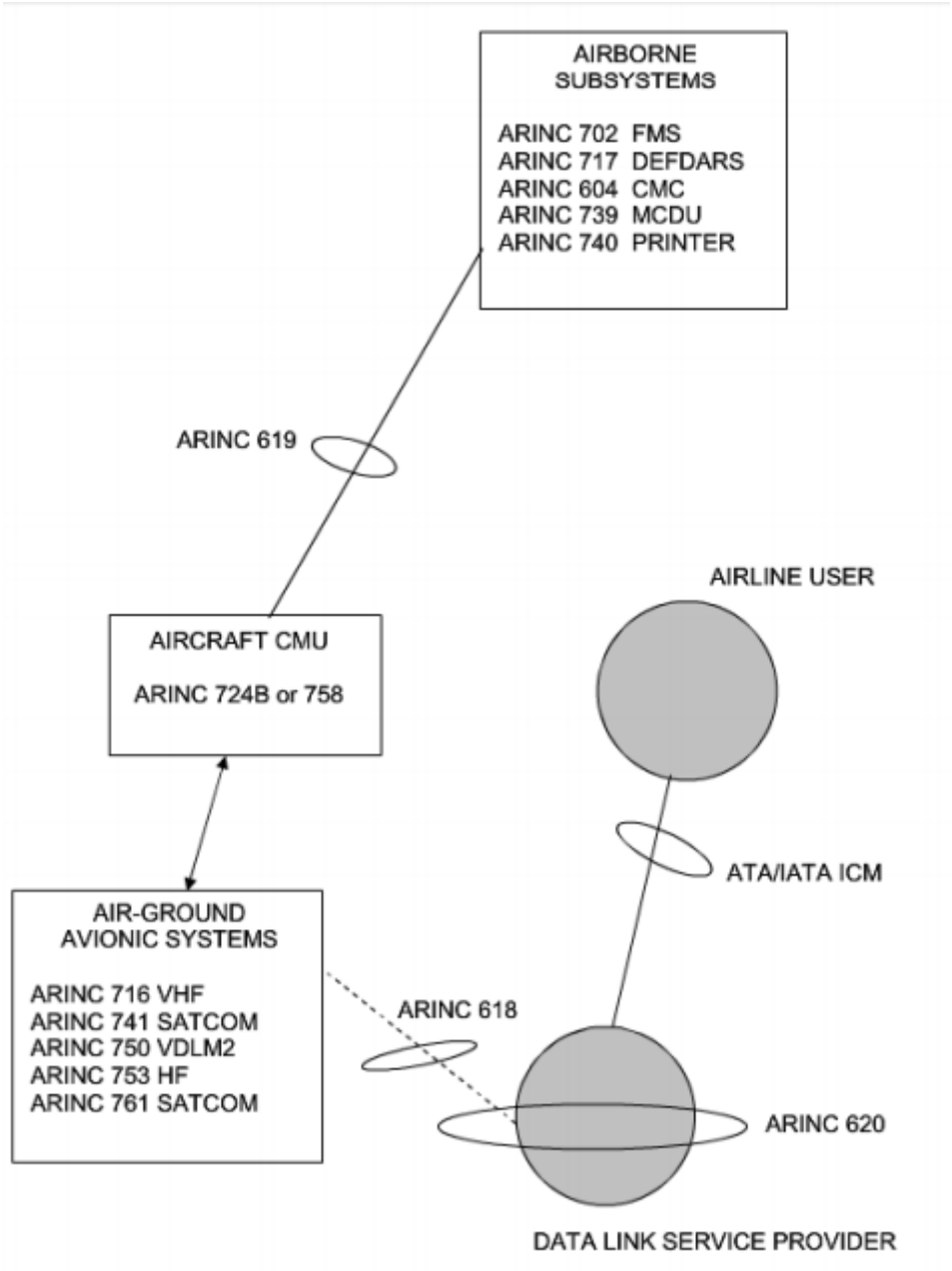


图 2-1　ARINC 各文件之间的应用关系

2.3　ACARS 数据链

通信协议是指信息在传输过程中所有传输数据都必须遵守的公共规范。地面终端设备与机载设备之间的数据传送都必须严格依照通信协议的标准，如 ARINC 618 文件、620 文件和 623 文件等。通信协议要求先对数据进行封装才能传送，否则接收设备就会因无法识别所接收到的信息从而致使数据传输失败。

飞机下传的 ACARS 报文通过机载设备的收发机自动选择合适链路（甚高频，高频，卫星通讯）下发，终端地面站（RGS）接收站会将接收到的信息通过地面网络传输至 ADCC 地面网络控制中心处理（将信息由 ARINC 618 格式的报文转换成

ARINC 620 格式的报文），转换成适用于地面终端用户使用的 ACARS 报文，网络控制中心根据民航系统的 X.25 分组交换网或 ATA/IATA ICM 选择合适的路由线路将报文发送至终端用户数据链信息处理中心进行进一步的数据解析，实时处理的报文信息则通过局域网或广域网发至飞机运行状态监控部门，以实现对飞机运行状态实时监控。上传请求报文则与下发报文相反。

ACARS 的通信网络主要有：

• Very High Frequency 甚高频 (VHF)

–使用一个甚高频地面无线电台网络

–使用音频频移键控，（或最小频移键控）

–本国的航班

–数据提供商：ARINC 公司和 SITA 公司

• SATCOM 卫星通讯：

–使用国际海事卫星组织（或者铱星）的通信卫星

–提供全球覆盖，除高纬度地区外

–当跨两极飞行时使用有障碍

• High Frequency 高频 (HF)：

–使用一个高频地面无线电台网络

–为极地航路提供服务

–只有 ARINC 公司提供该服务

2.4 ACARS 报文基本结构

2.4.1 空-地报文结构

空地数据通信传输过程中，空-地段传输报文一般由报头、正文和 BCS 校验码三部分组成。其基本结构如表 2-1 和表 2-2 所示。

表 2-1 空/地下行报文格式

名称	SOH	Mode	Aircraft Registration Number	TAK	Label	STX	MSN	Flight ID	Application Text	Suffix	BCS	BCS Suffix
大小	1	1	7	1	2	1	4	6	0-210	1	2	1
例	<SOH>	2	.N123XX	<NAK>	5Z	2	<STX>	XX0000		<ETX> OR <ETB>		<DEL>

表 2-2　地/空上行报文格式

名称	SOH	Mode	Aircraft Registration Number	TAK	Label	STX	Application Text	Suffix	BCS	BCS Suffix
大小	1	1	7	1	2	1	0-210	1	2	1
例	<SOH>	2	.N123XX	<NAK>	5Z	2		<ETX> OR <ETB>		<DEL>

在 ACARS 报文正文中传输的字符内容不是人为任意制定的，必须限制在 ISO-5 字符集的非控制字符中，并且报文正文的长度规定不能长于 220 个字符，否则该报文内容将被分成多块信息（Multi-block Message）进行下传。对于单块报文，报尾（Suffix）使用控制字符<ETX>(End of the Text)标示结束；对于多块报文，除了最后一块使用控制字符<ETX>标示结束外，其余各块报文均采用<ETB>(End of Block)标示结束。

报文的报头以起始标志符 <SOH>（Start of Head-ing）标示开始；标签（Label）只能使用 ARINC 620 文件中定义的标签，标签用于表明电报的类别；正文则以控制字符<STX>（Start of Text）标示正文开始；块校验码（BCS）则是利用循环冗余纠错原理形成的校验码，保证报文的准确性和完整性，减少数据出错的机率，BCS 校验的对象是从报头的起始标志符<SOH>开始到报尾的结束控制符<ETX>/<ETB>结束，但不包括<SOH>字符。[8]

按照表 2-1 形成的标准 ARINC 618 格式的报文，通过甚高频发射机发送至 RGS。

图 2-2 所示为航班 G-VMEG 于下午 15:49 从 香港国际机场起飞，飞往英国伦敦希斯罗机场的空 - 地段标准起飞报文。

图 2-2　报文格式

地面网络处理中心根据 ARINC 620 规范对图 2.2 中的信息重组形成地-地段报文格式如下。

QU XXXXXXX
.DSPXXXX DDHHMM
DEP
FI VS0201/AN G-VMEG
DT DSP RGS DDHHMM M37A
- VHHHHEGLL15493210

需要着重注意的：在报文信息开始传送之前，ACARS 报文还需在“SOH”字符前依次加上特殊控制字符 “+*SYNSYN”，用于表示数据开始传送，否则接收机将无法正常接收。同理，必须在“DEL”字符后面加上特殊的控制字符 “DLE”,以表示信息传输结束。前面四个特殊控制字符用于接收机检测和同步更新 ACRAS 报文信息。

2.4.2 地-地报文结构

经过地面网络控制中心转换后适合于地面终端用户使用的地-地段报文格式如表 2-3 和表 2-4 所示。ACARS 报文所使用的字符必须严格按照 ISO-5 字符集的规定使用。

表 2-3 下传地-地段报文格式

行	内容	实例
1	优先级/目的地址	QUKAKUOOHUXXXXXX
2	转发报文的地面站地址/转发时间	.BJSXCXA 261133
3	标准信息标识符 SMI	M14
4	文本元素	FIHU7920/ANB-2159/DA XXX/OF XXX/FB 0123/DS XXX
5	通信服务行	DT BJS HKG 261133 M67A
6-n	自由文本	-POSITION DMY 20JUN05，UTC 113327，FLT HU7920，LAT N24.013，LON E114.057，CAS 306,WD 5847，WS11，ALT 31500，FOB 190

下传地 - 地段报文标准格式说明

第 1 行：优先级 / 目的地址行

目前只有一个优先级在使用（QU），优先级后面是长度为 7 个字符的目的地地址，也可以选择添加多个地址，而多个地址之间需要使用空格隔开，但是最多只允许连续输入 16 个地址。

第 2 行：转发报文的地面站和报文转发时间

该行以 “.”开始，之后是转发报文的地面站地址以及转发时间，二者之间使用空格进行隔开。

第 3 行：标准信息标示符（SMI）

标准信息标示符（SMI）是一个长度为三的字符串，它和 label/sub-label 一起决定消息类型，如 SMI 为 ARR，label 为 QC，表示 ON（着陆）报。

第 4 行：文本元素行

该行是由文本元素（TE）组成的。每一个文本元素由三部分组成：文本元素标标示符（TEI）、文本元素内容（长度取决于 TEI）以及结束符（/）。其中，TEI 和文本元素内容之间用空格隔开。

第 5 行：通信服务行

该行以文本标示符 “DT”标示开始，随后是四个字段：DSP 标示符，接收下传信息的地面接收站的地址，接收信息的时间为 UTC 时间采用 DDHHMM 格式，报文序列号。各字段之间用空格隔开。

第 6-n 行：自由文本

该部分为自由确定内容，首行内容必须是短划线开头，紧接一个空格，之后则是自由文本主体部分。

表 2-4 上传地 - 地段报文格式

行	内容	实例
1	优先级/目的地址	QUKAKUOOHUXXXXXX
2	转发报文的地面站地址/转发时间	.BJSXCXA 261133
3	标准信息标识符 SMI	M14
4	文本元素	FIHU7920/ANB-2159/DA XXX/OF XXX/FB 0123/DS XXX
5-n	自由文本	-POSITION DMY 20JUN05，UTC 113327，FLT HU7920，LAT N24.013，LON E114.057，CAS 306,WD 5847，WS11，ALT 31500，FOB 190

上传地 - 地段报文标准格式说明

第 1~3 行，和自由文本行和下传地 - 地段报文相同；

第 4 行，文本元素行，该行一般为文本元素：

解释：AN：飞机注册号，来定位飞机；FI：航班号；MA：报文发送确认。

上传报文地址的定位，可以通过航班号或飞机注册号来施行。

不同的处理方法：

SITA：拒绝发送，并反馈错误代码 240；

ARINC：选择 AN（注册号）定位；

AIR CANADA、AVICOM：拒绝发送。

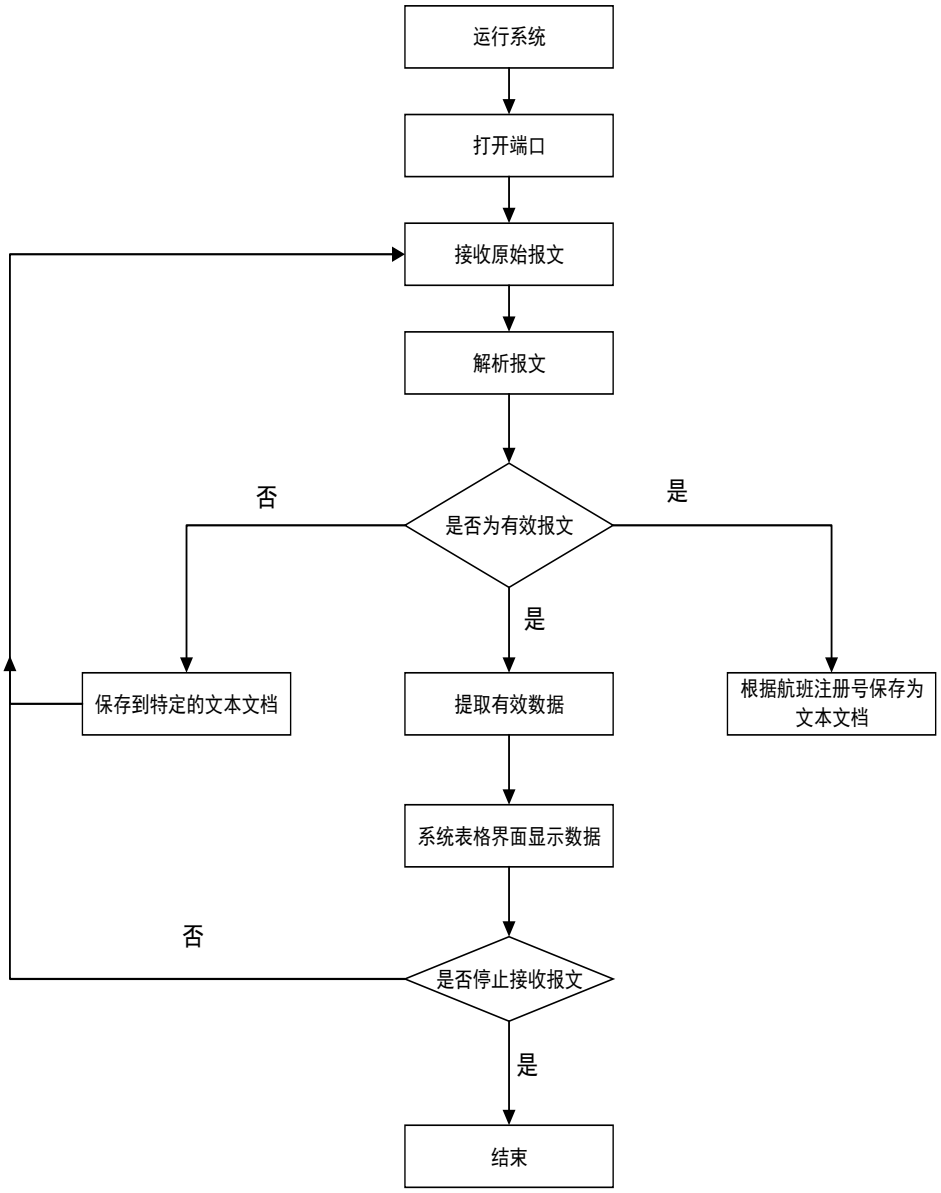
2.5 ACARS 报文解析方法

符合 ARINC 620 格式的报文解析通常采用报头解析和正文解析分别进行。由于 ACARS 报文报头参数位置相对固定，所以一般采用提取固定位置参数的方法对报头进行解析。固定位置参数提取法是根据各参数内容所在的位置和字符长度设定相应字段属性按位提取数据来实现报头解析的。

对于报文的正文部分则使用模板解析法和数据特征提取法相结合的方法来解析。模板解析法是根据报文正文中某些固定不变的格式所创建的模板库，解析时则调用相应模板，对报文正文按照模板库中模板已经设定好的字段属性按位提取解析。特征提取法则是针对正文中有些参数位置呈规律性的动态变化而产生的提取方法，它是利用专家知识建立的知识库，该库里保存了反映报文种类的最高准确率特征和相关参数信息，该方法是对模板解析法不足的有效补充。用它能够动态的对报文参数进行搜索截取的特性，能有效保证报文解析的准确性。

第 3 章　ACARS 航班监控系统设计

3.1　程序设计思路



？？？？？？？？？？？？？？？？？？？？

ACARS 航班监控系统主要分为 4 个模块：第一个模块，串口数据接收模块。由于需要接收实时的 ACARS 报文数据，所以本系统将实时导入从雷达接收的 ACARS 报文数据，通过调用 USB 串口，从数据接收机中导入已经接收的实时数据；第二个模块，报文解析模块。从接收机导入的报文时源码报文，需要对数据进行位

处理，从而将源码报文转换成明码报文；第三个模块，报文检测保存模块。检测报文是否为完整的报文，如果不是标准格式的报文或者乱码数据则将其保存在一个用于存储错误报文的文档中，如果是完整的报文，则提取出报文中的航班注册号，将报文保存在以其命名的文档中；第四个模块，报文数据提取并显示模块。使用特征提取法将所需数据从报文中提取出来，并将其保存在缓存中，然后将缓存中的数据显示在系统运行界面的表格中，而对应的报文则显示在表格下方的列表框中以便使用人员查询。

系统使用流程描述：

-启动系统。

-打开 USB 数据串口，开始从 USB 串口中导入实时 ACARS 报文数据。

-判断是否是符合 ACARS 格式的标准报文，如果不是，则将收到的乱码或错误报文存储于一个收集错误数据的文本文档中，并从新接收数据，如果是则进入下一步。

-解析报文，将报文从源码解析为明码。

-检查解析后的报文数据是否有效，如果是则筛选并提取其中有效数据缓存起来，将有效的报文保存为文本文档。

-将缓存中的报文数据提取出显示在系统界面的表格中。

-判断是否继续接收报文，如果是则转到第三部开始接收报文，否则进入下一步。

-停止接收报文，关闭系统。

3.2 主要函数

（1）.。。

BOOL CusDlg::OnInitDialog(){}

该函数用于初始化 MFC 程序，设置 Static Text、Edit 控件的默认显示内容，设置 List 控件将显示对象的集合，根据逻辑关系只允许导入数据按钮可以使用，暂时禁用其他按钮。

（2）

void CusDlg::OnInputData(){}

该函数用于监控“导入数据”按钮的点击情况，通过点击“导入数据”按钮，打开串口，激活 MScommn 控件，让 MScommon 控件开始监控接收数据。将“导入数据”按钮和“显示数据”按钮禁用，让“停止导入”按钮和“开始解码”按钮启用。

void CusDlg::OnOncommMscomm1(){}

该函数在程序打开串口后一旦接收到来自串口的数据就被激活，按照两个字符的流量接收数据，将接收到的数据以十六进制的格式存入 outDate 全局变量中，将数据以 ASCII 码格式转换成的字符串存入全局变量 w_code 中。

void CusDlg::OnDecode(){}

该函数通过监控按钮控件“开始解码”的点击事件进行激活。运行后首先清空s_reports模板内的数据，防止在后面运行过程中出现历史数据干扰。将outData，w_code数据分别赋值给m_outdata，m_code全局变量，用于后面函数的数据处理。随后清空outDate和w_code数据，调用 splitcode1()，splitcode2()函数用于解析接收到的数据并保存为文本文件。完成之后清空m_outdata和m_code数据。调用定时器函数，每隔2秒激活一次onTimer（）函数。

void splitcode1(CString & data){}

该函数以报头为区别，将收到的数据拆分为单条的报文，调用decode()函数进行解码用于，之后将解完码的报文加入s_reports模板中。

CString decode(const CString &hex){}

该函数将数据按三个字符的量输入charge（）函数用于将数据从源码转换成明码。

void splitcode2(CString &data){}

该函数将收到的数据拆分为单条报文，并调用savefile函数将报文保存为文本文档。

void savefile(CString & cur){}

该函数从报文数据中提取出航班注册号，并以此作为文本文档的文件名，为每条报文添加时间戳后保存到相应的文本文档中。

void CusDlg::OnTimer(UINT_PTR nIDEvent){}

该函数用于监控定时器，通过setTimer()函数激活，通过设置固定的时间间隔运行一次“开始解码”按钮绑定的函数。

void CusDlg::OnShowData(){}

该函数从s_reports模板中逐条提取出报文数据，保存在临时变量t_report中，通过reportToshow（）函数将报文中的数据提取出来存入o_report结构体中，将o_report中的数据通过m_programLangList.SetItemText()函数依次显示在List控件中。

void CusDlg::OnstopInput(){}

该函数关闭串口，禁用除“导入数据”按钮外所有的按钮控件，并停止定时器。

3.3 数据提取

```
void reportToshow(Mdata &mreports,CString &kk){
}
```

代码解释

```
kk.Replace(_T("SP"), _T(" "));
```

```
kk.Replace(_T("del"), _T(" "));
Cstring kkdata=kk;
kkdata.TrimLeft(_T("SYN"));
```

以上四行代码用于去除报头报尾，将SP字符串换为空格，方便后面数据提取。

```
sx = kkdata.Find(_T("SOH"));
    CString test;
    if (sx >= 0){
        mreports.mode = (kkdata.Mid(sx + 3)).Left(1);
        test = (kkdata.Mid(sx + 5)).Left(6);
        if (test.Find(_T("B-"))!=0)
        {
            mreports.bebreak = _T("1");
            return;
        }
        else{
            mreports.airReg = test;
        }
    }
    else{
        return;
    }
```

以上代码用于提取出报文中的航班注册号，如果提取的航班注册号不符合标准或没有提取出注册号则该条报文将不会被保存在文本文档中，同时也不会显示在表格中。

```
sx = kkdata.Find(_T("STX"));
    if (sx>=0)
    {
        CString messgeflightnum = (kkdata.Mid(sx + 3)).Left(10);
        mreports.mesNum = messgeflightnum.Left(4);
        mreports.flightNum = messgeflightnum.Mid(4);
    }
```

以上代码将提取出报文中的航班号和信息号

```
sx = kkdata.Find(_T("NAK"));
    if (sx >= 0)
    {
        CString labelblock = (kkdata.Mid(sx + 3)).Left(3);
```

```
        mreports.mesLabel = labelblock.Left(2);
        mreports.blockId = labelblock.Mid(2);
    }
```

以上代码将提取出报文中的信息标签和块号，此两者可以用于判断报文类型。

```
sx = kkdata.Find(_T("DEP"));
    if (sx >= 0)
    {
        mreports.dep = (kkdata.Mid(sx + 4)).Left(4);
    }
```

以上代码用于提取报文中目的机场四字码，如果该数据存在的话。

```
sx = kkdata.Find(_T("DES"));
    if (sx >= 0)
    {
        mreports.des =  (kkdata.Mid(sx + 4)).Left(4);
    }
```

以上代码用于提取报文中起飞机场四字码，如果该数据存在的话。

```
    sx = kkdata.Find(_T("CAS"));
    if (sx >= 0)
    {
        CString ccas = kkdata.Mid(sx + 4);
        int lin =0 ;
        while (ccas.GetAt(lin)>='0' && ccas.GetAt(lin)<='9'){
            lin++;

        }
        mreports.cas =  ccas.Left(lin+1);
    }
```

以上代码用于提取报文中的空速，如果该数据存在的话。

```
    sx = kkdata.Find(_T("LAT"));
    if (sx >= 0)
    {
```

```
            mreports.lat =  (kkdata.Mid(sx + 4)).Left(8);
            sx = kkdata.Find(_T("LON"));
            if (sx >= 0)
            {
                mreports.lon =  (kkdata.Mid(sx + 4)).Left(8);
            }
        }
        else{
            sx = kkdata.Find(_T("N "));
            if (sx>-1)
            {
                CString temp = (kkdata.Mid(sx)).Left(14);
                if (temp.GetAt(3) >= '0' && temp.GetAt(3) <= '9')
                {
                    mreports.lat = temp.Left(4) + _T(".") + (temp.Mid(4)).Left(3);
                    mreports.lon = (temp.Mid(7)).Left(4) + _T(".") + temp.Right
(3);

                }
            }
        }
```

以上代码用于提取报文中飞机的经纬度坐标，如果该数据存在的话。

```
sx = kkdata.Find(_T("ALT"));
        int s1;
        CString tempx;
        if (sx >= 0)
        {
            tempx = kkdata.Mid(sx+4);

            s1 = tempx.Find(_T(","));
            mreports.alt =  tempx.Left(s1);
        }
```

以上代码用于提取报文中的飞机高度，如果该数据存在的话。

```
    if (kkdata.Find(_T("FOBt")))
```

```
{
    sx = kkdata.Find(_T("FOBt"));
    if (kkdata.GetAt(sx + 4) >= 0 && kkdata.GetAt(sx + 4) <= 9)
    {
        tempx = kkdata.Mid(sx + 4);
        s1 = tempx.Find(_T(","));
        mreports.fobt =  tempx.Left(s1);
    }
    sx = kkdata.Find(_T("FOBp"));
    if (kkdata.GetAt(sx + 4) >= 0 && kkdata.GetAt(sx + 4) <= 9)
    {
        tempx = kkdata.Mid(sx + 4);
        s1 = tempx.Find(_T(","));
        mreports.fobp =  tempx.Left(s1);
    }

}
else{
    sx = kkdata.Find(_T("FOB"));
    if (sx>-1)
    {
        CString ffob = kkdata.Mid(sx + 4);
        int nin = 0;
        while (ffob.GetAt(nin) >= '0' && ffob.GetAt(nin) <= '9')
        {
            nin++;

        }
        mreports.fob = ffob.Left(nin + 1);
    }

}
```

以上代码用于提取报文中的飞机载油量，如果该数据存在的话。

```
sx = kkdata.Find(_T("UTC"));
 if (sx>=0)
 {
```

```
            mreports.utc = (kkdata.Mid(sx + 4)).Left(6);
        }
```

以上代码用提取报文中的UTC时间，如果该数据存在的话。

```
sx = kkdata.Find(_T("ETA"));
        if (sx >= 0)
        {
            mreports.eta =  (kkdata.Mid(sx + 4)).Left(4);
        }
```

以上代用于提取飞机预计到达时间，如果该数据存在的话。

```
sx = kkdata.Find(_T("DMY"));
        if (sx >= 0)
        {
            tempx = kkdata.Mid(sx + 4);

            s1 = tempx.Find(_T(","));
            mreports.dmy = tempx.Left(s1);
        }
```

以上代码用于提取报文的日期，如果该数据存在的话。

```
sx = kkdata.Find(_T("WD"));
        if (sx >= 0)
        {
            if (kkdata.GetAt(sx+3)>= 0 && kkdata.GetAt(sx+3)<=9 )
            {
                tempx = kkdata.Mid(sx + 3);
                s1 = tempx.Find(_T(","));
                mreports.wd =tempx.Left(s1);
            }
        }
```

以上代码用于提取风向，如果该数据存在的话。

```
sx = kkdata.Find(_T("WS"));
        if (sx >= 0)
        {
```

```
            if (kkdata.GetAt(sx + 3) >= 0 && kkdata.GetAt(sx + 3) <= 9)
            {
                tempx = kkdata.Mid(sx + 3);
                s1 = tempx.Find(_T(","));
                mreports.ws =  tempx.Left(s1);
            }
        }
```

以上代码用于提取风速，如果该数据存在的话

```
 sx = kkdata.Find(_T("ON"));
        if (sx >= 0)
        {
            if (kkdata.GetAt(sx + 3) >= 0 && kkdata.GetAt(sx + 3) <= 9)
            {
                tempx = kkdata.Mid(sx + 3);
                s1 = tempx.Find(_T(","));
                mreports.on =  tempx.Left(s1);
            }
        }
```

以上代码用于提取飞机到达时间，如果该数据存在的话

```
 sx = kkdata.Find(_T("OUT"));
        if (sx >= 0)
        {
            if (kkdata.GetAt(sx + 4) >= 0 && kkdata.GetAt(sx + 4) <= 9)
            {
                tempx = kkdata.Mid(sx + 4);
                s1 = tempx.Find(_T(","));
                mreports.out =  tempx.Left(s1);
            }
}
```

以上代码用于提取飞机推出时间，如果该数据存在的话。

```
 sx = kkdata.Find(_T("OFF"));
        if (sx >= 0)
        {
```

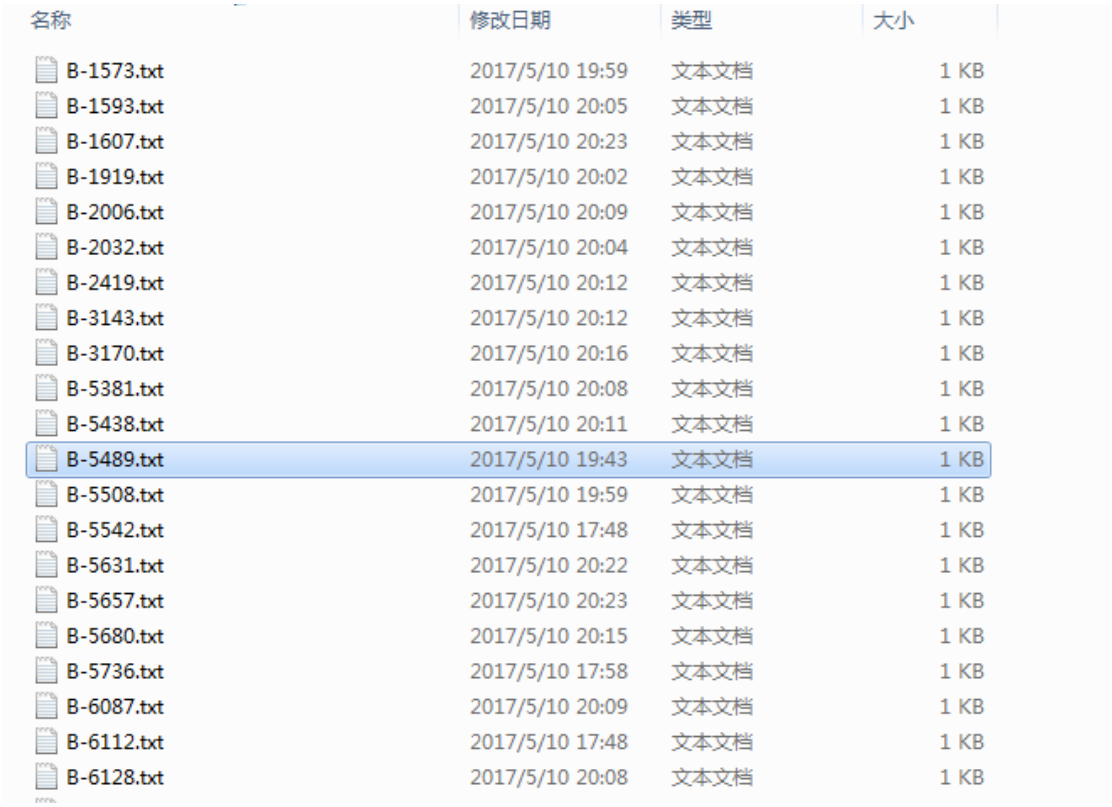
```
            if (kkdata.GetAt(sx + 4) >= 0 && kkdata.GetAt(sx + 4) <= 9)
            {
                tempx = kkdata.Mid(sx + 4);
                s1 = tempx.Find(_T(","));
                mreports.off =tempx.Left(s1);
            }
        }

}
```

以上代码用于提取飞机离场时间，如果该数据存在的话。

第 4 章　系统运行实验分析

4.1　ACARS 实时数据接收

通过接收设备，实时接收天津上空的 ACARS 信息，并将其按照飞机注册号存于相应的文本文档中



名称	修改日期	类型	大小
B-1573.txt	2017/5/10 19:59	文本文档	1 KB
B-1593.txt	2017/5/10 20:05	文本文档	1 KB
B-1607.txt	2017/5/10 20:23	文本文档	1 KB
B-1919.txt	2017/5/10 20:02	文本文档	1 KB
B-2006.txt	2017/5/10 20:09	文本文档	1 KB
B-2032.txt	2017/5/10 20:04	文本文档	1 KB
B-2419.txt	2017/5/10 20:12	文本文档	1 KB
B-3143.txt	2017/5/10 20:12	文本文档	1 KB
B-3170.txt	2017/5/10 20:16	文本文档	1 KB
B-5381.txt	2017/5/10 20:08	文本文档	1 KB
B-5438.txt	2017/5/10 20:11	文本文档	1 KB
B-5489.txt	2017/5/10 19:43	文本文档	1 KB
B-5508.txt	2017/5/10 19:59	文本文档	1 KB
B-5542.txt	2017/5/10 17:48	文本文档	1 KB
B-5631.txt	2017/5/10 20:22	文本文档	1 KB
B-5657.txt	2017/5/10 20:23	文本文档	1 KB
B-5680.txt	2017/5/10 20:15	文本文档	1 KB
B-5736.txt	2017/5/10 17:58	文本文档	1 KB
B-6087.txt	2017/5/10 20:09	文本文档	1 KB
B-6112.txt	2017/5/10 17:48	文本文档	1 KB
B-6128.txt	2017/5/10 20:08	文本文档	1 KB

图 4-1　接收到的正常报文保存文档截图

接收到的报文范例：

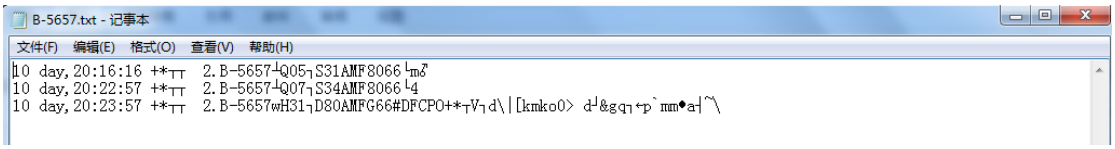


图 4-2　接收到的正常报文内容截图

每条报文在其报头之前添加一个时间戳，用于记录保存报文的时间。分析连续性、数据的准确率、特性分析系统运行效果。

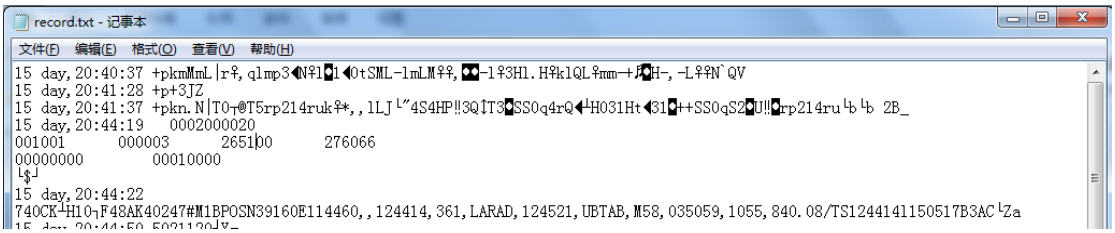


图 4-3　接收到的非正常报文内容截图

将每条飞正常的报文也添加上时间戳，方便后期查询。

系统运行实时截图

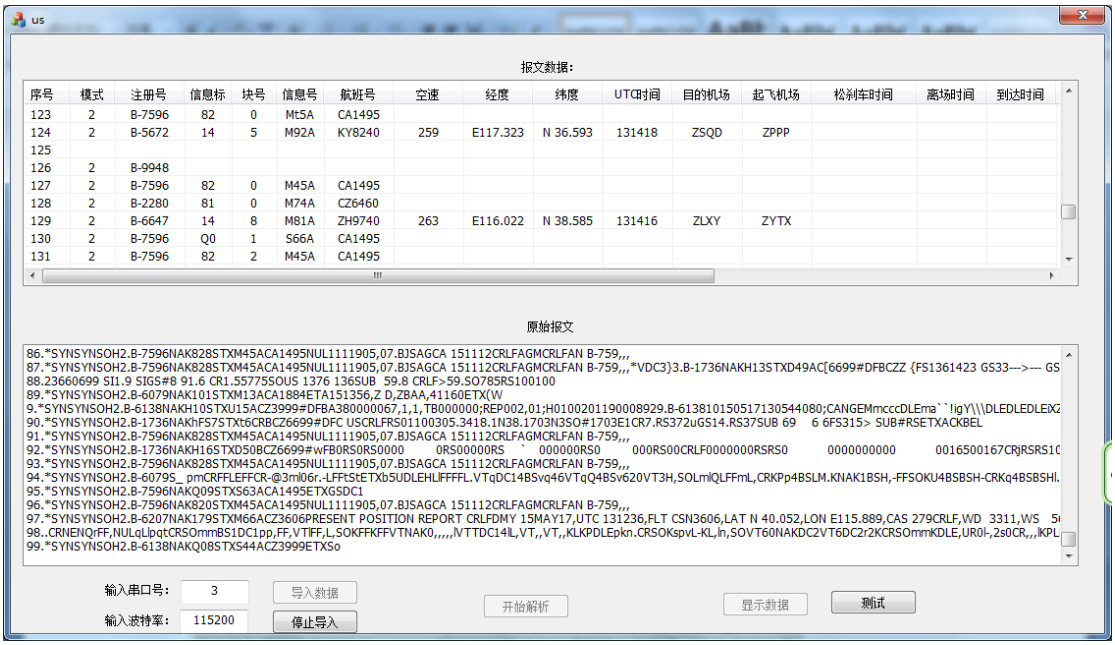


图 4-4　系统运行实时截图

按照接收报文的时间顺序，依次把提取的数据显示在系统运行框上方的报文数据表格中，下方的原始报文列表框用来显示每条数据解码后的报文，便于使用人员验证从报文中提取出来的数据。

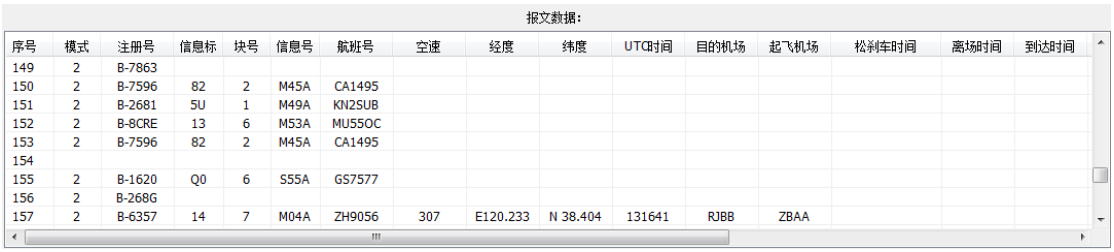


图 4-5　显示报文数据表格特写图片

该表格主要显示报文显示序号、报文模式、航班注册号、报文信息标、报文块号、报文信息号、航班号、航班空速、航班位置经纬度、UTC 时间、目的机场、起飞机场、松刹车时间、离场时间、到达时间、航班高度、航班载油量、风向、风速、日期数据。

图 4-6　原始报文显示特写

将每条原始报文添加与表格中数据对应的序号显示在列表框中，可以直观的查看每条接收到的报文。

4.2　数据特性分析

通过对接收到的 508 份报文进行分析得到以下结论：

报文接收完整的占比为 378:508 即 74.41%，所以有效数据率为 74.41%；

接收到不完整的报文占比为 130:508 即 25.59%，这其中接收到不完整正常报文的占总报文数的 13.59%，接收到的乱码报文占总报文数的 12%。

在系统运行过程中，发现接收到完整的报文也会掺杂部分乱码或者无效数据。比如图 4-7：

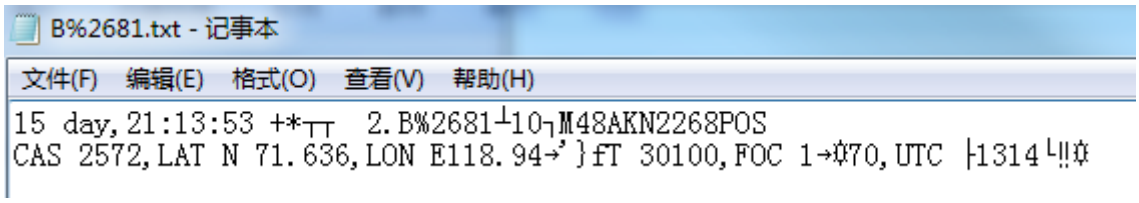


图 4-7 完整的报文夹杂乱码的报文截图

该条报文，有“+*SYNSYN”控制字符和“DLE”对报文进行包装，但是该报文包含的航班注册号夹杂了一个“%”字符乱码，在经度坐标之后夹杂了“}”乱码，该报文传输的 UTC 数据也不是完整的信息，由于此类错误在所接收到的报文中出现机率不算罕见，在检查过程序设计后，判断此类数据的错误是由于接收机的缘故导致的，由于接收机的功率不够，在接收数据过程中对以接收到的数据处理出错，当然也有可能是在报文发送是受到干扰出现错误。因此本 ACARS 航班监控系统对已经提取处来的数据需要进行过滤，只有符合该数据对应格式的数据才会显示在系统界面的表格中，否则，将拒绝显示该数据。

4.3　系统运行效果分析

通过实验运行得知，系统能够将接收到的完整的正常报文中的数据顺利提取出所需要的数据，并显示在系统的表格中，与之对应的报文原文将显示在下方的。而不完整的报文或者乱码报文则被刷选出去，不予显示，只将其保存在文本文档中，因此初步达到了论文开题时的要求。但是 74.41%的有效报文说明系统还有很大的提升空间，使用更好的接收设备、选择更好的接收环境、接收设备的功率等都是影响

接收报文质量的重要影响因素。由于本系统只涉及到软件开发层面，因此更多的需要提升对数据提取程序的优化以及对提取数据的验证，减少使用者在使用本系统时的疑惑以及增加提取数据的正确率。

第 5 章　总结与展望

5.1　全文总结

ACARS航班监控系统从论文开题立项、系统总体设计、框架搭建、程序编写到最后完成系统开发并进行实时数据接收测试。从无到有，笔者全程参与整个系统的开发过程。在完成整个系统过程，不仅是完成一个接收报文提取数据的过程，更是一次对自身掌握知识运用到实践中的尝试。总的来说，本系统，运行稳定，满足了开题时的任务要求。

创新点有：

（1）利用雷达收到来自机场实时的ACARS报文数据，通过筛选将完整ACARS报文保存为文本文档，方便以后提取报文并与ADS-B数据等航班监控数据进行对比。

（2）独立自主开发出报文数据提取模块，将报文中的有效信息从夹杂有误码滤波等的报文提取出来，并显示在表格中，可供使用者实时观测

此系统的开发与运用，可以辅助该系统的使用人员进行ACARS报文数据的观测，同时通过保存下来的实时报文数据，在日后工作过程中需要查询相关报文时也有保存好的历史数据可用。

5.2　ACARS应用展望

在系统开发过程中，不仅察觉到了ACARS航班监控的便利之处，同时也意识到了该系统的缺陷。主要表现有：

（1）报文发送频率很低，导致真实航班运行信息与报文中的数据有所差异。

（2）误码过多，受到环境和设备的影响，在发送和接收过程中导致报文内数据丢失或出错，给数据提取造成一定困扰。

以上问题容易造成的信息收发丢失或链路中断。因此，若要规范的使用ACARS发挥其真正的效用，提高航班监控手段和预先处理问题的能力就需要加强对设备的研发、严格按照运行规章和制度来使用ACARS。

正是因为ACARS传输数据快速且抗干扰性能优秀，因此，ACARS必将承担愈加重要的数据传输作用，基于ACARS在国内航班飞机的部署相当广泛，依靠ACARS所开发的相关产品的推广成本也可以轻松的降低。反过来较低的推广成本，又能够促进ACARS的相关研究和试验，让ACARS技术的发展进入一个良性促进发展的状态。

同时由于 ACARS 的运用前景相当广泛，比如由于 ACARS 具有速度快，信息传输量大的特点，通过 ACARS 设备下传信息不仅可以涵盖飞机的性能数据，而且也能够携带部分商用信息，如乘客订票数据或订专车等私人服务，而 ACARS 在空中能够发挥的功能更多也更规范。比如使用地面性能分析计算机将跑道的风向、风

速和温度以及飞机重量实况等信息快速性能分析然后上传，飞机的机载设备使用接收飞行计划报文，机组可立即获得自动上传的性能报告，机组根据报告能够得知飞机起飞所需推力的大小、跑道的长度以及应采用的爬升率等。通过这种信息处理模式可以大大提高航班飞行的经济性和安全性。

在空管系统中，航空管制员则可以使用 ACARS 的特性采用数字化起飞放行以及数字化终端区信息服务，据此可以安全快速的同时放行多个航班，提高管制员的放行能力和机场的利用率。

由于 ACARS 在航空多个领域的大量运用，使原先简单的地空通信系统的作用变得越加重要，也将承担更加繁重的任务，也使航空公司看到了 ACARS 的使用上带来的巨大利益，这也是 ACARS 今后的发展方向。[9]

参考文献

[1] 钟青，张其善. ACARS 报文通信服务器的设计与实现. 北京航空航天大学学报, 2001, 27(5)：1001-5965.

[2] 熊力, 催杉. 民航监视系统. 科技创新导报, 2014.

[3] 郭旭周. 甚高频空地数据链 ACARS 系统的调制解调技术研究:（硕士学位论文）.南京：南京航空航天大学，2008.

[4] 姜景明. 民航飞机新型通信寻址与报告系统介绍. 互联网+健康, 2009.

[5] ARINC specification 618-6, Air/ground character-oriented pro-tocol specification[S]. 2006，7-17.

[6] ARINC specification 620-6, Air/ground character-oriented pro-tocol specification[S]. 2007，11-39.

[7] ARINC specification 623-3, Air/ground character-oriented pro-tocol specification[S]. 2005，12-26.

[8] 李书明, 吕文礼等. ACARS 空地数据通信系统及报文解析. Equipment Manufacturing Technology, 2014 ：1672-545X

[9] 杨龙. 飞机通信寻址与报告系统的研究与应用:（硕士学位论文）.山东：山东大学，2011.

致　谢

经过数月的论证、程序编写、测试、打磨，随着致谢，这篇本科论文也将落下最后一笔，同时大学四年生活划伤一个圆满的句号时刻也即将来临。回忆大学四年的学习生活，留下许多宝贵的人生财富，四年来，收到来自师长、同学和朋友各方面的帮助与支持，是我终身受益，一切中的一切都是历历在目，让人倍感留恋，倍感珍惜。

在本文的撰写过程中，张召悦老师作为我的毕业设计指导老师，他治学严谨，平易近人，视野广阔，往往在我做毕业设计遇到无法解决的难题时，他总能及时给予我正确的引导和帮助。他严肃的科学态度，一丝不苟的学术精神，求同存异的工作作风也激励和引导着我，让我向着正确方向前行。因此，他不仅仅是在学业上给我悉心的指导，更是在思想、生活上给予我帮助和引导。可以说这篇论文的完成离不开他耐心的教诲和细心的更正。

同时，也要感谢本论文所有引用过的各位学者的专著，如果没有这些学者研究成果的启发和帮助，本篇论文的完成也就不会如此顺利和完善，甚至是有可能影响完成本篇论文的最终成果。所以对于本论文引用过的专著表达诚挚的谢意。

求学生涯暂告段落，但是求知的道路永无停滞，大学教会我终身学习的品质我将永记于心。这些品质将伴随我在以后的人生道路上，勇敢前行。

附录 A：程序清单

usDlg.cpp 文件

```
// usDlg.cpp :  实现文件
//

#include "stdafx.h"
#include "us.h"
#include "usDlg.h"
#include "afxdialogex.h"
#include "mscomm1.h"
#include <string>
#include <vector>
#include <string.h>
#ifdef _DEBUG
#define new DEBUG_NEW
#endif
using namespace std;
CRect rect;
CString m_outdata;
CString m_code;
int w_count=0;
CString w_code;
//  用于应用程序“关于”菜单项的  CAboutDlg  对话框
CString outData;
vector <CString> s_reports;
class CAboutDlg : public CDialogEx
{
public:
	CAboutDlg();

//  对话框数据
	enum { IDD = IDD_ABOUTBOX };

	protected:
	virtual void DoDataExchange(CDataExchange* pDX);	// DDX/DDV  支持

//  实现
protected:
	DECLARE_MESSAGE_MAP()
public:
	afx_msg void OnTimer(UINT_PTR nIDEvent);
};

CAboutDlg::CAboutDlg() : CDialogEx(CAboutDlg::IDD)
{
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialogEx::DoDataExchange(pDX);
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialogEx)
	ON_WM_TIMER()
```

```
END_MESSAGE_MAP()


// CusDlg 对话框



CusDlg::CusDlg(CWnd* pParent /*=NULL*/)
	: CDialogEx(CusDlg::IDD, pParent)
	, m_portnum(0)
	, m_baudR(0)
{
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

void CusDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialogEx::DoDataExchange(pDX);
	DDX_Control(pDX, IDC_LIST1, m_programLangList);

	DDX_Text(pDX, IDC_EDIT5, m_portnum);
	DDX_Text(pDX, IDC_EDIT6, m_baudR);
	DDX_Control(pDX, IDC_MSCOMM1, m_common1);
	DDX_Control(pDX, IDC_LIST2, m_ListWords);
}

BEGIN_MESSAGE_MAP(CusDlg, CDialogEx)
	ON_WM_SYSCOMMAND()
	ON_WM_PAINT()
	ON_WM_QUERYDRAGICON()
	ON_BN_CLICKED(IDC_BUTTON1, &CusDlg::OnInputData)
	ON_BN_CLICKED(IDC_BUTTON2, &CusDlg::OnDecode)
	ON_BN_CLICKED(IDC_BUTTON3, &CusDlg::OnShowData)
	ON_WM_TIMER()
	ON_BN_CLICKED(IDC_BUTTON4, &CusDlg::OnstopInput)
	ON_BN_CLICKED(IDC_BUTTON5, &CusDlg::OnBnsavefile)
END_MESSAGE_MAP()


// CusDlg 消息处理程序

BOOL CusDlg::OnInitDialog()
{
	CDialogEx::OnInitDialog();

	// 将“关于...”菜单项添加到系统菜单中。

	// IDM_ABOUTBOX 必须在系统命令范围内。
	ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
	ASSERT(IDM_ABOUTBOX < 0xF000);

	CMenu* pSysMenu = GetSystemMenu(FALSE);
	if (pSysMenu != NULL)
	{
		BOOL bNameValid;
		CString strAboutMenu;
		bNameValid = strAboutMenu.LoadString(IDS_ABOUTBOX);
		ASSERT(bNameValid);
```

```
        if (!strAboutMenu.IsEmpty())
        {
            pSysMenu->AppendMenu(MF_SEPARATOR);
            pSysMenu->AppendMenu(MF_STRING,                IDM_ABOUTBOX,
strAboutMenu);
        }
    }

    // 设置此对话框的图标。    当应用程序主窗口不是对话框时，框架将自动
    //  执行此操作
    SetIcon(m_hIcon, TRUE);              // 设置大图标
    SetIcon(m_hIcon, FALSE);         // 设置小图标

    // TODO:  在此添加额外的初始化代码
    CEdit*   pEdit1 = (CEdit*)GetDlgItem(IDC_EDIT5);//获取相应的编辑框 ID
    pEdit1->SetWindowText(_T("1")); //设置默认显示的内容
    CEdit*   pEdit2 = (CEdit*)GetDlgItem(IDC_EDIT6);//获取相应的编辑框 ID
    pEdit2->SetWindowText(_T("115200")); //设置默认显示的内容
    // 获取编程语言列表视图控件的位置和大小
    m_programLangList.GetClientRect(&rect);

    // 为列表视图控件添加全行选中和栅格风格
    m_programLangList.SetExtendedStyle(m_programLangList.GetExtendedStyle()    |
LVS_EX_FULLROWSELECT | LVS_EX_GRIDLINES);

    // 为列表视图控件添加三列
    m_programLangList.InsertColumn(0, _T("序号"), LVCFMT_CENTER, rect.Width()
/ 23, 0);
    m_programLangList.InsertColumn(1, _T("模式"), LVCFMT_CENTER, rect.Width()
/ 23, 1);       //mode
    m_programLangList.InsertColumn(2,  _T("注册号"),  LVCFMT_CENTER,
rect.Width() / 15, 2);  //airReg
    m_programLangList.InsertColumn(3,  _T("信息标"),  LVCFMT_CENTER,
rect.Width() /22, 3);   //mesLabel
    m_programLangList.InsertColumn(4, _T("块号"), LVCFMT_CENTER, rect.Width()
/ 23, 4);       //blockId
    m_programLangList.InsertColumn(5,  _T("信息号"),  LVCFMT_CENTER,
rect.Width() / 22, 5);  //mesNum
    m_programLangList.InsertColumn(6,  _T("航班号"),  LVCFMT_CENTER,
rect.Width() / 15, 6);  //flightNum
    m_programLangList.InsertColumn(12,  _T("目的机场"),  LVCFMT_CENTER,
rect.Width() / 15, 12);     //dep
    m_programLangList.InsertColumn(13,  _T("起飞机场"),  LVCFMT_CENTER,
rect.Width() / 15, 13);     //des
    m_programLangList.InsertColumn(14,  _T("松刹车时间"),  LVCFMT_CENTER,
rect.Width() / 10, 14);     //out
    m_programLangList.InsertColumn(19,  _T("离场时间"),  LVCFMT_CENTER,
rect.Width() / 15, 19);          //off
    m_programLangList.InsertColumn(11,  _T("到达时间"),  LVCFMT_CENTER,
rect.Width() / 15, 11);//on
    m_programLangList.InsertColumn(7, _T("空速"), LVCFMT_CENTER, rect.Width()
/ 15, 7);       //cas
    m_programLangList.InsertColumn(8, _T("经度"), LVCFMT_CENTER, rect.Width()
/ 15, 8);       //lat
    m_programLangList.InsertColumn(9, _T("纬度"), LVCFMT_CENTER, rect.Width()
/ 15, 9);       //lon
    m_programLangList.InsertColumn(15,  _T("高度"),  LVCFMT_CENTER,
rect.Width() / 15, 15);          //alt
```

```
    m_programLangList.InsertColumn(16, _T(" 载 油 量 "), LVCFMT_CENTER,
rect.Width() / 22, 16);          //fob
    m_programLangList.InsertColumn(17, _T("FOBt"), LVCFMT_CENTER,
rect.Width() / 23, 17);          //fobt
    m_programLangList.InsertColumn(18, _T("FOBp"), LVCFMT_CENTER,
rect.Width() / 23, 18);          //fobp
    m_programLangList.InsertColumn(10, _T("UTC 时 间 "), LVCFMT_CENTER,
rect.Width() / 15, 10);          //utc
    m_programLangList.InsertColumn(20, _T("预计到达时间"), LVCFMT_CENTER,
rect.Width() / 10, 20);     //eta
    m_programLangList.InsertColumn(21, _T(" 风 向 "), LVCFMT_CENTER,
rect.Width() / 23, 21);          //wd
    m_programLangList.InsertColumn(22, _T(" 风 速 "), LVCFMT_CENTER,
rect.Width() / 23, 22);          //ws
    m_programLangList.InsertColumn(23, _T(" 日 期 "), LVCFMT_CENTER,
rect.Width() / 10, 23);          //dmy


    //((CButton*)GetDlgItem(IDC_BUTTON2))->EnableWindow(false);
    ((CButton*)GetDlgItem(IDC_BUTTON3))->EnableWindow(false);
    ((CButton*)GetDlgItem(IDC_BUTTON4))->EnableWindow(false);
    return TRUE;  // 除非将焦点设置到控件，否则返回 TRUE
}

void CusDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
    if ((nID & 0xFFF0) == IDM_ABOUTBOX)
    {
        CAboutDlg dlgAbout;
        dlgAbout.DoModal();
    }
    else
    {
        CDialogEx::OnSysCommand(nID, lParam);
    }
}

// 如果向对话框添加最小化按钮，则需要下面的代码
//  来绘制该图标。  对于使用文档/视图模型的 MFC 应用程序，
//  这将由框架自动完成。

void CusDlg::OnPaint()
{
    if (IsIconic())
    {
        CPaintDC dc(this); // 用于绘制的设备上下文

        SendMessage(WM_ICONERASEBKGND,
reinterpret_cast<WPARAM>(dc.GetSafeHdc()), 0);

        // 使图标在工作区矩形中居中
        int cxIcon = GetSystemMetrics(SM_CXICON);
        int cyIcon = GetSystemMetrics(SM_CYICON);
        CRect rect;
        GetClientRect(&rect);
        int x = (rect.Width() - cxIcon + 1) / 2;
        int y = (rect.Height() - cyIcon + 1) / 2;
```

```
        // 绘制图标
        dc.DrawIcon(x, y, m_hIcon);
    }
    else
    {
        CDialogEx::OnPaint();
    }
}

//当用户拖动最小化窗口时系统调用此函数取得光标
//显示。
HCURSOR CusDlg::OnQueryDragIcon()
{
    return static_cast<HCURSOR>(m_hIcon);
}
CString xd[128] = { _T("NUL"), _T("SOH"), _T("STX"), _T("ETX"), _T("EOT"),
_T("ENQ"), _T("ACK"), _T("BEL"), _T("BS"), _T("HT"), _T("LF"), _T("VT"),
_T("FF"), _T("CR"), _T("SO"), _T("SI"),
_T("DLE"), _T("DC1"), _T("DC2"), _T("DC3"), _T("DC4"), _T("NAK"), _T("SYN"),
_T("ETB"), _T("CAN"), _T("EM"), _T("SUB"), _T("ESC"), _T("FS"), _T("GS"),
_T("RS"), _T("US"),
_T("SP"), _T("!"), _T("\""), _T("#"), _T("$"), _T("%"), _T("&"), _T("'"), _T("("),
_T(")"), _T("*"), _T("+"), _T(","), _T("-"), _T("."), _T("/"),
_T("0"), _T("1"), _T("2"), _T("3"), _T("4"), _T("5"), _T("6"), _T("7"), _T("8"), _T("9"),
_T(":"), _T(";"), _T("<"), _T("="), _T(">"), _T("?"),
_T("@"), _T("A"), _T("B"), _T("C"), _T("D"), _T("E"), _T("F"), _T("G"), _T("H"),
_T("I"), _T("J"), _T("K"), _T("L"), _T("M"), _T("N"), _T("O"),
_T("P"), _T("Q"), _T("R"), _T("S"), _T("T"), _T("U"), _T("V"), _T("W"), _T("X"),
_T("Y"), _T("Z"), _T("["), _T("\\"), _T("]"), _T("^"), _T("_"),
_T("`"), _T("a"), _T("b"), _T("c"), _T("d"), _T("e"), _T("f"), _T("g"), _T("h"), _T("i"),
_T("j"), _T("k"), _T("l"), _T("m"), _T("n"), _T("o"),
_T("p"), _T("q"), _T("r"), _T("s"), _T("t"), _T("u"), _T("v"), _T("w"), _T("x"), _T("y"),
_T("z"), _T("{"), _T("|"), _T("}"), _T("~"), _T("del")
};
CString ISO5toascii(const CString &val)
{
    CString x = val;
    char x1 = x.GetAt(0);
    char x2 = x.GetAt(1);
    int y, y1, y2;
    if (x1 >= '0' && x1 <= '9')
    {
        y1 = x1 - '0';
    }
    else if (x1 >= 'A' && x1 <= 'Z')
    {
        y1 = x1 - 'A' + 10;
    }
    if (x2 >= '0' && x2 <= '9')
    {
        y2 = x2 - '0';
    }
    else if (x2 >= 'A' && x2 <= 'Z')
    {
        y2 = x2 - 'A' + 10;
    }
    y = y1 * 16 + y2;
```

```
    return xd[y];
}
CString trans(const int &x){
    switch (x)
    {
    case 0:
        return _T("0");
    case 1:
        return _T("1");
    case 2:
        return _T("2");
    case 3:
        return _T("3");
    case 4:
        return _T("4");
    case 5:
        return _T("5");
    case 6:
        return _T("6");
    case 7:
        return _T("7");
    case 8:
        return _T("0");
    case 9:
        return _T("1");
    case 10:
        return _T("2");
    case 11:
        return _T("3");
    case 12:
        return _T("4");
    case 13:
        return _T("5");
    case 14:
        return _T("6");

    default:
        return _T("7");

    }
}
CString charge(CString &value)
{
    CString hex = value;
    if (hex.GetLength()>3    || hex.GetLength()<3)
    {
        return _T("");
    }
    CString strbin = _T("");
    CString binCur;
    char cCur = hex.GetAt(0);
    if (cCur >= '0' && cCur <= '9')
    {
        int x = cCur - '0' + 0;
        binCur = trans(x);
    }
    else if (cCur >= 'A' && cCur <= 'Z')
    {
```

```
            int x = cCur - 'A' + 10;
            binCur = trans(x);
        }
        strbin = binCur + hex.GetAt(1)+_T(" ");
        strbin = ISO5toascii(strbin);
        return strbin;
}

CString decode(const CString &hex){
        CString temp = hex;
        CString result;
        temp = temp.TrimLeft();
        int getindex=0;
        while (temp.GetLength()>0)
        {
            CString cCur = temp.Left(3);
            result += charge(cCur);
            temp = temp.Mid(3);

        }

        return result+_T("\n");
}

void CusDlg::OnstopInput()
{
        // TODO:  在此添加控件通知处理程序代码
        m_common1.put_PortOpen(FALSE);
        KillTimer(1);
        ((CButton*)GetDlgItem(IDC_BUTTON4))->EnableWindow(FALSE);
        ((CButton*)GetDlgItem(IDC_BUTTON1))->EnableWindow(TRUE);
        ((CButton*)GetDlgItem(IDC_BUTTON2))->EnableWindow(FALSE);

}
void CusDlg::OnInputData()
{
        // TODO:  在此添加控件通知处理程序代码
        //更新控件数据到变量
        UpdateData(TRUE);
        if (m_common1.get_PortOpen())
        {
            m_common1.put_PortOpen(FALSE);

        }/*
        else
        {
            m_common1.put_PortOpen(TRUE);
            ((CButton*)GetDlgItem(IDC_BUTTON1))->SetWindowTextW(_T("导入数据
"));
        }*/

        m_common1.put_CommPort(m_portnum); //选择 com3.可以根据情况更改
        m_common1.put_InBufferSize(1024); //设置输入缓冲区的大小，Bytes
        m_common1.put_OutBufferSize(1024); //设置输出缓冲区大小，bytes
        CString settings;
        settings.Format(_T("%d,n,8,1"),m_baudR);
        m_common1.put_Settings(settings); //波特率 115200，无校验，8 个数据位，停止
位 1、
```

```
    m_common1.put_InputMode(1);  //1：表示以二进制方式检索数据
    m_common1.put_RThreshold(1);  //参数 2 表示每当串口接收缓冲区中有多余或等于 1 个字符时引发一个接收数据的 oncomm 事件
    m_common1.put_InputLen(0);     //设置当前接收区长度是 0
    if (!m_common1.get_PortOpen())
    {
        m_common1.put_PortOpen(TRUE);
        ((CButton*)GetDlgItem(IDC_BUTTON1))->SetWindowTextW(_T("导入数据"));

    }
    else
    {
        AfxMessageBox(_T("Can not open serial port!"));
    }
    m_common1.get_Input(); //先预读缓冲区以清除残留数据
    /*CString settings;

    settings.Format(_T("%d"), m_portnum);
    MessageBox(settings);
    */
    /*
    //outData = _T("s;lkdaslsjeionskkassldl");
    CStdioFile file;
    CString pathfile = _T("E:\\c++\\mfc\\us\\data.txt");
    if (file.Open(pathfile,CFile::modeCreate | CFile::modeNoTruncate | CFile::modeReadWrite))
    {
        file.SeekToEnd();
        file.WriteString(outData);
        memset(&outData, NULL, sizeof(outData));
    }
    else
    {
        MessageBox(_T("写入文件失败"));
        return;
    }
    file.Close();
    MessageBox(_T("完成"));*/

    ((CButton*)GetDlgItem(IDC_BUTTON3))->EnableWindow(false);
    ((CButton*)GetDlgItem(IDC_BUTTON2))->EnableWindow(true);
    ((CButton*)GetDlgItem(IDC_BUTTON1))->EnableWindow(false);
    ((CButton*)GetDlgItem(IDC_BUTTON4))->EnableWindow(true);
}
char HexChar(char c)
{
    if ((c >= '0') && (c <= '9'))
        return c - '0';//将?0-9 的 Ì?数 ºy 字 Á?字 Á?符¤?转 Áa 为 a 十 º?六¢¨´进?制?格?式 º?
    else if ((c >= 'A') && (c <= 'F'))
        return c - 'A' + 10;//将?A-F 的 Ì?字 Á?符¤?转 Áa 为 a 十 º?六¢¨´进?制?格?式 º?例¤y 如¨?字 Á?符¤?'C'-'A'+10=12=0x0C
    else if ((c >= 'a') && (c <= 'f'))
        return c - 'a' + 10;//将?a-f 的 Ì?字 Á?符¤?转 Áa 为 a 十 º?六¢¨´进?制?格?式 º?
    else
        return 0x10;
}
```

```
void CusDlg::OnOncommMscomm1()
{
    // TODO:  在此处添加消息处理程序代码
    VARIANT variant_inp;
    COleSafeArray safearray_inp;
    LONG len, k,x;
    BYTE rxdata[2048]; //设置 BYTE 数组  AN 8-bit integerthat is not signed.
    CString strhextemp;
    CString strtemp;

    if (m_common1.get_CommEvent() == 2)  //事件值为 2 表示接收缓冲区内有字符
    {
        variant_inp = m_common1.get_Input();  //读缓冲区
        safearray_inp = variant_inp;  //VARIANT 型变量转换为 ColeSafeArray 型变
量
        len = safearray_inp.GetOneDimSize(); //得到有效数据长度
        for (k = 0; k < len; k++)
        {
            safearray_inp.GetElement(&k, rxdata + k);//转换为 BYTE 型数组
        }

        for (k = 0; k < len; k++)  //将数组转换为 CString 型变量
        {
            BYTE bt = *(char*)(rxdata + k);          //字符型
            //char bT = HexChar(bt);
            strhextemp.Format(_T("%02X "), bt);       // 将字符送入临时变量
strhextemp 存放十六进制
            strtemp.Format(_T("%c"), bt);            //按照 ASCII 码保存
            outData += strhextemp;                   //加入接收编辑框对应字符串
            w_code += strtemp;


        }
    }
    /*((CusDlg*)(AfxGetApp()->m_pMainWnd))->m_ListWords.AddString(outData);
    ((CusDlg*)(AfxGetApp()->m_pMainWnd))->m_ListWords.SetTopIndex(
        ((CusDlg*)(AfxGetApp()->m_pMainWnd))->m_ListWords.GetCount() - 1);

    CStdioFile file;
    CString pathfile = _T("E:\\c++\\mfc\\us\\data.txt");
    if (file.Open(pathfile, CFile::modeCreate | CFile::modeNoTruncate |
CFile::modeReadWrite))
    {
        file.SeekToEnd();
        file.WriteString(outData);
        memset(&outData, NULL, sizeof(outData));
    }
    else
    {
        MessageBox(_T("写入文件失败"));
        return;
    }
    file.Close();*/
}
void CAboutDlg::OnTimer(UINT_PTR nIDEvent){

}
```

```
bool te = true;
void splitcode1(CString & data){
	CString th1 = _T("2B 2A 16 16 01 ");//通过 2B 2A 16 16 01  实现报文分离
	CString th2 = _T("AB 2A 16 16 01 ");//通过 AB 2A 16 16 01  实现报文分离
	CString th;
	if (data.Find(th1)>-1)
	{
		th = th1;
	}
	else
	{
		th = th2;
	}
	CString ctemp;
	int getre = 1;
	getre = data.Find(th);
	int x = data.GetLength();
	//int cou = 0;
	CString findvalue = _T(" ");
	while (getre >= 0 && data.GetLength()>0)
	{
		CString cCur;
		CString tring;
		CString temp;

		cCur = data.Mid(0);
		getre = cCur.Find(th, 1);

		if (getre<0)
		{
			s_reports.push_back(decode(cCur));
			return;
		}
		cCur = data.Left(getre);
		//cou++;
		/*temp=m_pBuf.Left(getre);
		//tring = decode(cCur);
		//ctemp=ctemp+tring;
		//decode(cCur);
		temp = temp.TrimLeft();
		CString result=_T("0");

		int getindex = 0;
		while (temp.GetLength()>0)
		{
		CString cCur = temp.Left(3);
		result += charge(cCur);
		temp = temp.Mid(3);

		}*/

		//MessageBox(result);
		/**/
		ctemp += decode(cCur);//使用解码函数，将源码解析为明码
		s_reports.push_back(decode(cCur));

		data = data.Mid(getre);
```

```
    }
    //w_code = ctemp;//  明码保存到全局变量 w_code 中，用于显示数据按钮调用
    //AfxMessageBox(ctemp);
}
void savefile(CString & cur){
    int sx,sy;
    CString filename,temp;
    sx=cur.Find(_T("+*"));
    if (sx>-1)
    {
        sy = cur.Find(_T("B-"));
        if (sy>-1)
        {
            temp = cur.Mid(sy).Left(6);
        }
        else
        {
            temp = cur.Mid(sx + 7).Left(6);
        }
    }
    else{
        filename = _T("E:\\c++\\mfc\\us\\wrongdata\\record.txt");
        CString str;//获取系统时间
        SYSTEMTIME st;
        GetLocalTime(&st);
        str.Format(_T("%2d  day,%2d:%2d:%2d  "),  st.wDay,  st.wHour,  st.wMinute,
st.wSecond);
        str = str + cur + _T("\n");
        CStdioFile file;

        if  (!file.Open(filename,  CFile::modeCreate  |  CFile::modeWrite  |
CFile::modeNoTruncate))
        {

            return;
        }
        else
        {
            file.SeekToEnd();
            file.WriteString(str);
        }
        file.Close();
        return;
    }
    filename =_T("E:\\c++\\mfc\\us\\data\\")+ temp + _T(".txt");
    CString str; //获取系统时间
    SYSTEMTIME st;
    GetLocalTime(&st);
    str.Format(_T("%2d  day,%2d:%2d:%2d  "),  st.wDay,  st.wHour,  st.wMinute,
st.wSecond);
    str = str + cur +_T("\n");
    CStdioFile file;
    /*if                  (file.Open(filename,                  CFile::modeCreate|
CFile::modeWrite|CFile::modeNoTruncate))
    {
        file.SeekToEnd();
```

```
            memset(&str, NULL, sizeof(str));
            file.WriteString(str);

        }
        else
        {
            AfxMessageBox(_T("xiru shibai"));
        }
        file.Close();*/

        if      (!file.Open(filename,      CFile::modeCreate      |      CFile::modeWrite      |
CFile::modeNoTruncate))
        {

            return;
        }
        else
        {
            file.SeekToEnd();
            file.WriteString(str);
        }
        file.Close();
}
void splitcode2(CString &data){
        /*
        CFile file(_T("E:\\c++\\mfc\\us\\aa01.txt"), CFile::modeRead);
        char *pBuf;
        int iLen = file.GetLength();

        pBuf = new char[iLen + 1];
        file.Read(pBuf, iLen);
        pBuf[iLen] = 0;
        file.Close();

        CString m_pBuf(pBuf);
        CString th1 = _T("2B 2A 16 16 01 ");//通过 2B 2A 16 16 01  实现报文分离
        CString th2 = _T("AB 2A 16 16 01 ");//通过 AB 2A 16 16 01  实现报文分离
        CString th;
        if (m_pBuf.Find(th1)>-1)
        {
        th = th1;
        }
        else
        {
        th = th2;
        }
        CString ctemp;
        int getre=1;
        getre = m_pBuf.Find(th);
        int x = m_pBuf.GetLength();

        CString findvalue = _T(" ");
        while(getre>=0 && m_pBuf.GetLength()>0)
        {
        CString cCur;
        CString tring;
        CString temp;
```

```
	cCur = m_pBuf.Mid(0);
	getre = cCur.Find(th,1);
	cCur = m_pBuf.Left(getre);*/
	/*temp=m_pBuf.Left(getre);
	//tring = decode(cCur);
	//ctemp=ctemp+tring;
	//decode(cCur);
	temp = temp.TrimLeft();
	CString result=_T("0");

	int getindex = 0;
	while (temp.GetLength()>0)
	{
	CString cCur = temp.Left(3);
	result += charge(cCur);
	temp = temp.Mid(3);

	}*/

	//MessageBox(result);
	/*
	ctemp += decode(cCur);//使用解码函数，将源码解析为明码
	s_reports.push_back (decode(cCur));
	m_pBuf=m_pBuf.Mid(getre);
	}*/


	CString th = _T("+*");   //实现报文分离
	CString ctemp;
	int getre = 0;
	getre = data.Find(th);
	int x = data.GetLength();
	//int cou = 0;
	while (getre >= 0 && data.GetLength() > 0)
	{
		CString cCur;
		CString tring;
		CString temp;

		cCur = data.Mid(0);
		getre = cCur.Find(th, 1);
		if (getre < 0 )
		{
			savefile(cCur);
			//AfxMessageBox(cCur);
			return;
		}
		//cou++;
		cCur = data.Left(getre);
		//AfxMessageBox(cCur);
		//s_reports.push_back (cCur); //将分离出来的报文加入数组
		savefile(cCur);
		data = data.Mid(getre);
	}

}

void CusDlg::OnTimer(UINT_PTR nIDEvent)
```

```
{
    // TODO:  在此添加消息处理程序代码和/或调用默认值

    m_outdata += outData;
    m_code += w_code;
    outData.Empty();
    w_code.Empty();
    splitcode1(m_outdata);
    splitcode2(m_code);
    CusDlg::OnShowData();
    m_outdata.Empty();
    m_code.Empty();
    CDialogEx::OnTimer(nIDEvent);
}
void CusDlg::OnDecode()
{
    // TODO:  在此添加控件通知处理程序代码

    s_reports.clear();
    m_outdata += outData;
    m_code += w_code;
    outData.Empty();
    w_code.Empty();
    splitcode1(m_outdata);
    splitcode2(m_code);
    m_outdata.Empty();
    m_code.Empty();
    SetTimer(1, 3000, NULL);

    //((CButton*)GetDlgItem(IDC_BUTTON2))->EnableWindow(false);
    //((CButton*)GetDlgItem(IDC_BUTTON1))->EnableWindow(false);
    ((CButton*)GetDlgItem(IDC_BUTTON3))->EnableWindow(true);
    //splitcode1();
    //splitcode2();

    /*
    CFile file(_T("E:\\c++\\mfc\\us\\aa01.txt"), CFile::modeRead);
    char *pBuf;
    int iLen = file.GetLength();

    pBuf = new char[iLen + 1];
    file.Read(pBuf, iLen);
    pBuf[iLen] = 0;
    file.Close();

    CString m_pBuf(pBuf);
    CString th1 = _T("2B 2A 16 16 01 ");//通过 2B 2A 16 16 01  实现报文分离
    CString th2 = _T("AB 2A 16 16 01 ");//通过 AB 2A 16 16 01  实现报文分离
    CString th;
    if (outData.Find(th1)>-1)
    {
        th = th1;
    }
    else
    {
        th = th2;
    }
    CString ctemp;
```

```
    int getre=1;
    getre = outData.Find(th);
    int x = outData.GetLength();

    CString findvalue = _T(" ");
    while(getre>=0 && outData.GetLength()>0)
    {
        CString cCur;
        CString tring;
        CString temp;

        cCur = outData.Mid(0);
        getre = cCur.Find(th,1);
        cCur = outData.Left(getre);*/
                                    /*temp=m_pBuf.Left(getre);
                                    //tring = decode(cCur);
                                    //ctemp=ctemp+tring;
                                    //decode(cCur);
                                    temp = temp.TrimLeft();
                                    CString result=_T("0");

                                    int getindex = 0;
                                    while (temp.GetLength()>0)
                                    {
                                        CString cCur = temp.Left(3);
                                        result += charge(cCur);
                                        temp = temp.Mid(3);

                                    }*/

        //MessageBox(result);
    /*
        ctemp += decode(cCur);//使用解码函数，将源码解析为明码
        s_reports.push_back (decode(cCur));

        outData=outData.Mid(getre);



    }
    w_code = ctemp;//  明码保存到全局变量 w_code 中，用于显示数据按钮调用

    MessageBox(ctemp);*/


}
struct Mdata
{
    CString mode;
    CString airReg;
    CString mesLabel;
    CString blockId;
    CString mesNum;
    CString flightNum;
    CString dep;
    CString des;
    CString off;
    CString on;
```

```
    CString in;
    CString out;
    CString cas;
    CString lat;
    CString lon;
    CString alt;
    CString fob;
    CString fobt;
    CString  fobp;
    CString utc;
    CString  eta;
    CString wd;
    CString ws;
    CString dmy;
    CString bebreak;

};
//Mdata mreports;
//void reportToshow(CString &mode,CString &airReg,CString & mesLabel,CString &blockId,CString &mesNum,CString &flightNum,CString &cas, CString &lat,CString &lon,CString &utc,CString &on,CString &dep,CString &des,CString &out,CString &alt,CString &fob,CString &fobt,CString &fobp,CString &off,CString &eta,CString &wd,CString &ws,CString &day, CString & kk){
void reportToshow(Mdata &mreports,CString &kk){
    kk.Replace(_T("SP"), _T(" "));
    kk.Replace(_T("del"), _T(" "));
    kk.Replace(_T("?"), _T(" "));
    kk.Replace(_T("+"), _T(""));
    CString kkdata = kk;
    kkdata.TrimLeft(_T("SYN"));
    int sx;
    sx = kkdata.Find(_T("SOH"));
    CString test;
    if (sx >= 0){
        mreports.mode = (kkdata.Mid(sx + 3)).Left(1);
        //mreports.mode = (kkdata.Mid(sx+3)).Left(1);
        //mreports.airReg = (kkdata.Mid(sx + 5)).Left(6);
        test = (kkdata.Mid(sx + 5)).Left(6);
        if (test.Find(_T("B-"))!=0)
        {
            mreports.bebreak = _T("1");
            return;
        }
        else{
            mreports.airReg = test;
        }
        /*
        CString labelblock = (kkdata.Mid(sx + 14)).Left(3);
        mreports.mesLabel = labelblock.Left(2);
        mreports.blockId = labelblock.Mid(2);
        CString messgeflightnum = (kkdata.Mid(sx + 18)).Left(10);
        mreports.mesNum = messgeflightnum.Left(4);
        mreports.flightNum = messgeflightnum.Mid(4);*/
    }
    else{
        return;
    }
```

```
sx = kkdata.Find(_T("STX"));
if (sx>=0)
{
	CString messgeflightnum = (kkdata.Mid(sx + 3)).Left(10);
	mreports.mesNum = messgeflightnum.Left(4);
	mreports.flightNum = messgeflightnum.Mid(4);
}

sx = kkdata.Find(_T("NAK"));
if (sx >= 0)
{
	CString labelblock = (kkdata.Mid(sx + 3)).Left(3);
	mreports.mesLabel = labelblock.Left(2);
	mreports.blockId = labelblock.Mid(2);
}

sx = kkdata.Find(_T("DEP"));
if (sx >= 0)
{
	mreports.dep = (kkdata.Mid(sx + 4)).Left(4);
}

sx = kkdata.Find(_T("DES"));
if (sx >= 0)
{
	mreports.des =    (kkdata.Mid(sx + 4)).Left(4);
}

sx = kkdata.Find(_T("CAS"));
if (sx >= 0)
{
	CString ccas = kkdata.Mid(sx + 4);
	int lin =0 ;
	while (ccas.GetAt(lin)>='0' && ccas.GetAt(lin)<='9'){
		lin++;

	}
	mreports.cas =    ccas.Left(lin);
}

sx = kkdata.Find(_T("LAT"));
if (sx >= 0)
{
	mreports.lat =    (kkdata.Mid(sx + 4)).Left(8);
	sx = kkdata.Find(_T("LON"));
	if (sx >= 0)
	{
		mreports.lon =    (kkdata.Mid(sx + 4)).Left(8);
	}
}
else{
	sx = kkdata.Find(_T("N "));
	if (sx>-1)
	{
		CString temp = (kkdata.Mid(sx)).Left(14);
		if (temp.GetAt(3) >= '0' && temp.GetAt(3) <= '9')
		{
```

```
                mreports.lat = temp.Left(4) + _T(".") + (temp.Mid(4)).Left(3);
                mreports.lon = (temp.Mid(7)).Left(4) + _T(".") + temp.Right(3);

            }
        }
}



sx = kkdata.Find(_T("ALT"));
int s1;
CString tempx;
if (sx >= 0)
{
    tempx = kkdata.Mid(sx+4);

    s1 = tempx.Find(_T(","));
    mreports.alt =    tempx.Left(s1);
}
size_t i;

if (kkdata.Find(_T("FOBt")))
{
    sx = kkdata.Find(_T("FOBt"));
    if (kkdata.GetAt(sx + 4) >= 0 && kkdata.GetAt(sx + 4) <= 9)
    {
        tempx = kkdata.Mid(sx + 4);
        s1 = tempx.Find(_T(","));
        mreports.fobt =    tempx.Left(s1);
    }
    sx = kkdata.Find(_T("FOBp"));
    if (kkdata.GetAt(sx + 4) >= 0 && kkdata.GetAt(sx + 4) <= 9)
    {
        tempx = kkdata.Mid(sx + 4);
        s1 = tempx.Find(_T(","));
        mreports.fobp =    tempx.Left(s1);
    }

}
else{
    sx = kkdata.Find(_T("FOB"));
    if (sx>-1)
    {
        CString ffob = kkdata.Mid(sx + 4);
        int nin = 0;
        while (ffob.GetAt(nin) >= '0' && ffob.GetAt(nin) <= '9')
        {
            nin++;

        }
        mreports.fob = ffob.Left(nin + 1);
    }

}

sx = kkdata.Find(_T("UTC"));
if (sx>=0)
{
```

```
        mreports.utc = (kkdata.Mid(sx + 4)).Left(6);
}

sx = kkdata.Find(_T("ETA"));
if (sx >= 0)
{
        mreports.eta =    (kkdata.Mid(sx + 4)).Left(4);
}

sx = kkdata.Find(_T("DMY"));
if (sx >= 0)
{
        tempx = kkdata.Mid(sx + 4);

        s1 = tempx.Find(_T(","));
        mreports.dmy = tempx.Left(s1);
}

sx = kkdata.Find(_T("WD"));
if (sx >= 0)
{
        if (kkdata.GetAt(sx+3)>= 0 && kkdata.GetAt(sx+3)<=9 )
        {
                tempx = kkdata.Mid(sx + 3);
                s1 = tempx.Find(_T(","));
                mreports.wd =tempx.Left(s1);
        }
}

sx = kkdata.Find(_T("WS"));
if (sx >= 0)
{
        if (kkdata.GetAt(sx + 3) >= 0 && kkdata.GetAt(sx + 3) <= 9)
        {
                tempx = kkdata.Mid(sx + 3);
                s1 = tempx.Find(_T(","));
                mreports.ws =    tempx.Left(s1);
        }
}

sx = kkdata.Find(_T("ON"));
if (sx >= 0)
{
        if (kkdata.GetAt(sx + 3) >= 0 && kkdata.GetAt(sx + 3) <= 9)
        {
                tempx = kkdata.Mid(sx + 3);
                s1 = tempx.Find(_T(","));
                mreports.on =    tempx.Left(s1);
        }
}

sx = kkdata.Find(_T("OUT"));
if (sx >= 0)
{
        if (kkdata.GetAt(sx + 4) >= 0 && kkdata.GetAt(sx + 4) <= 9)
        {
                tempx = kkdata.Mid(sx + 4);
                s1 = tempx.Find(_T(","));
```

```
                mreports.out =    tempx.Left(s1);
            }
        }

        sx = kkdata.Find(_T("OFF"));
        if (sx >= 0)
        {
            if (kkdata.GetAt(sx + 4) >= 0 && kkdata.GetAt(sx + 4) <= 9)
            {
                tempx = kkdata.Mid(sx + 4);
                s1 = tempx.Find(_T(","));
                mreports.off =tempx.Left(s1);
            }
        }




}
void resetstruct(Mdata &st){
        st.airReg.Empty();
        st.alt.Empty();
        st.blockId.Empty();
        st.cas.Empty();
        st.dep.Empty();
        st.des.Empty();
        st.dmy.Empty();
        st.eta.Empty();
        st.flightNum.Empty();
        st.fob.Empty();
        st.fobp.Empty();
        st.fobt.Empty();
        st.in.Empty();    //
        st.lat.Empty();
        st.lon.Empty();
        st.mesLabel.Empty();
        st.mesNum.Empty();
        st.mode.Empty();
        st.off.Empty();
        st.on.Empty();
        st.out.Empty();
        st.utc.Empty();
        st.wd.Empty();
        st.ws.Empty();
        st.bebreak.Empty();
}
void CusDlg::OnShowData()
{
        // TODO:  在此添加控件通知处理程序代码
        int count = w_count;
        Mdata o_report;
        CString xuhao;
        CString
mode,airReg,mesLabel,blockId,mesNum,flightNum,cas,lat,lon,utc,on,dep,des,out,alt,fob,
fobt,fobp,off,eta,wd,ws,day;
        CString listdata;
        int x = s_reports.size();
```

```
    /*    int y;
    for (size_t i = 0; i < 57; i++)
    {
        CString t_report = s_reports[i];
        reportToshow(o_report, t_report);
        y = o_report.airReg.Find(_T("B-"));
        listdata =
o_report.airReg+o_report.alt+o_report.blockId+o_report.cas+o_report.dep+o_report.des
+o_report.dmy+o_report.eta+o_report.flightNum+o_report.fob+o_report.lat+o_report.lo
n+o_report.mode;
        ((CusDlg*)(AfxGetApp()->m_pMainWnd))->m_ListWords.AddString(listdata);
        ((CusDlg*)(AfxGetApp()->m_pMainWnd))->m_ListWords.SetTopIndex(
            ((CusDlg*)(AfxGetApp()->m_pMainWnd))->m_ListWords.GetCount()    -
1);
    }

if (t_report==_T(""))
    {
        MessageBox(_T("kongle"));
    }
    MessageBox(t_report);


    s_reports.clear();
    for (size_t i = 0; i < 30; i++)
    {
        CString                                ddd                                =
_T("+*SYNSYNSOH2.B-2335NAK017STXM12AMU5479POSCRLFCASSP287,LAT
SPNSP36.834,LONSPE117.235,ALTSP23605,FNBSP37880,UTCSP001048ETXDC1Vd
el");
        s_reports.push_back(ddd);
    }

    // 在列表视图控件中插入列表项，并设置列表子项文本
    */
    for (int i = 0; i < s_reports.size(); i++){
        CString t_report = s_reports[i];
        //mode = airReg = mesLabel = blockId = mesNum = flightNum = cas = lat = lon
= utc = on = dep = des = out = alt = fob = fobt = fobp = off = eta = wd = ws = day =
NULL;
        //memset(&mreports, 0, sizeof(mreports));
        //reportToshow(mode, airReg, mesLabel, blockId, mesNum, flightNum, cas, lat,
lon, utc, on, dep, des, out, alt, fob, fobt, fobp, off, eta, wd, ws, day, t_report);
        resetstruct(o_report);
        reportToshow(o_report, t_report);
        if (o_report.bebreak==_T("1"))
        {
            continue;
        }
        xuhao.Format(_T("%d"), count+1);
        //listdata.Format(_T("%d .") + t_report, count + 1);
        listdata = xuhao + _T(".") + t_report;
        //listdata = o_report.mode + o_report.airReg + o_report.mesLabel +
o_report.blockId + o_report.mesNum + o_report.flightNum + o_report.dep +
o_report.des + o_report.out + o_report.off + o_report.on +
        //o_report.cas + o_report.lat + o_report.lat + o_report.lon + o_report.alt +
o_report.fob + o_report.utc + o_report.eta + o_report.wd + o_report.ws + o_report.dmy;
        ((CusDlg*)(AfxGetApp()->m_pMainWnd))->m_ListWords.AddString(listdata);
```

```
        ((CusDlg*)(AfxGetApp()->m_pMainWnd))->m_ListWords.SetTopIndex(
        ((CusDlg*)(AfxGetApp()->m_pMainWnd))->m_ListWords.GetCount() - 1);

        m_programLangList.InsertItem(count, xuhao);
        m_programLangList.SetItemText(count, 1, o_report.mode);
        m_programLangList.SetItemText(count, 2, o_report.airReg);
        m_programLangList.SetItemText(count, 3, o_report.mesLabel);
        m_programLangList.SetItemText(count, 4, o_report.blockId);
          m_programLangList.SetItemText(count, 5, o_report.mesNum);
        m_programLangList.SetItemText(count, 6, o_report.flightNum);
        m_programLangList.SetItemText(count, 7, o_report.cas);
        m_programLangList.SetItemText(count, 8, o_report.lon);
        m_programLangList.SetItemText(count, 9, o_report.lat);
        m_programLangList.SetItemText(count, 10, o_report.utc);
        m_programLangList.SetItemText(count, 11, o_report.dep);      //on
        m_programLangList.SetItemText(count, 12, o_report.des);
        m_programLangList.SetItemText(count, 13, o_report.out);
        m_programLangList.SetItemText(count, 14, o_report.off);
        m_programLangList.SetItemText(count, 15, o_report.on);
        m_programLangList.SetItemText(count, 16, o_report.alt);
        m_programLangList.SetItemText(count, 17, o_report.fob);
        m_programLangList.SetItemText(count, 18, o_report.fobt);
        m_programLangList.SetItemText(count, 19, o_report.fobp);
        m_programLangList.SetItemText(count, 20, o_report.eta);
        m_programLangList.SetItemText(count, 21, o_report.wd);
        m_programLangList.SetItemText(count, 22, o_report.ws);
        m_programLangList.SetItemText(count, 23, o_report.dmy);

        count++;
        UpdateData();
    }
    w_count = count;
    s_reports.clear();
    ((CButton*)GetDlgItem(IDC_BUTTON2))->EnableWindow(false);
    ((CButton*)GetDlgItem(IDC_BUTTON1))->EnableWindow(false);
    ((CButton*)GetDlgItem(IDC_BUTTON3))->EnableWindow(false);
    /*
    xuhao.Format(_T("%d"), count+1);
    m_programLangList.InsertItem(0, xuhao);
    m_programLangList.SetItemText(0, 1, _T("1"));
    m_programLangList.SetItemText(0, 2, _T("1"));
    m_programLangList.InsertItem(1, _T("C"));
    m_programLangList.SetItemText(1, 1, _T("2"));
    m_programLangList.SetItemText(1, 2, _T("2"));
    m_programLangList.InsertItem(2, _T("C#"));
    m_programLangList.SetItemText(2, 1, _T("3"));
    m_programLangList.SetItemText(2, 2, _T("6"));
    m_programLangList.InsertItem(3, _T("C++"));
    m_programLangList.SetItemText(3, 1, _T("4"));
    m_programLangList.SetItemText(3, 2, _T("3"));
    for (size_t i = 4; i < 10; i++)
    {
        m_programLangList.InsertItem(i, _T("Java"));
        m_programLangList.SetItemText(i, 1, _T("1"));
        m_programLangList.SetItemText(i, 2, _T("1"));
    }

    CString kkdata;
```

```
    kkdata =
_T("SYNSYNSOH2.B-2335NAK017STXM12AMU5479POSCRLFCASSP287,LATSP
NSP36.834,LONE117.235,ALT23605,FNB37880,UTCSP001048ETXDC1VDEL");
    kkdata.Replace(_T("SP"),_T(" "));
    kkdata.TrimLeft(_T("SYNOH"));
    CString mode = kkdata.Left(1);
    int sx = kkdata.Find(_T("-"));
    CString reg = kkdata.Mid(sx - 1);
    sx = kkdata.Find(_T("STX"));
    CString messgeflightnum = kkdata.Mid(sx + 3);
    CString messagenum = messgeflightnum.Left(4);
    messgeflightnum = messgeflightnum.Mid(4);
    CString flightnum = messgeflightnum.Left(6);
    reg = reg.Left(6);
    CString showdata = _T("mode:") + mode + _T("reg number:") + reg +
_T("messagenum:") + messagenum + _T("flight num:") + flightnum;
    MessageBox(showdata);
    ((CusDlg*)(AfxGetApp()->m_pMainWnd))->m_ListWords.AddString(showdata);
    ((CusDlg*)(AfxGetApp()->m_pMainWnd))->m_ListWords.SetTopIndex(
        ((CusDlg*)(AfxGetApp()->m_pMainWnd))->m_ListWords.GetCount()-1);
*/
}
BEGIN_EVENTSINK_MAP(CusDlg, CDialogEx)
    ON_EVENT(CusDlg, IDC_MSCOMM1, 1, CusDlg::OnOncommMscomm1,
VTS_NONE)
END_EVENTSINK_MAP()




void CusDlg::OnBnsavefile()
{
    // TODO:  在此添加控件通知处理程序代码
    /*CString cCur = _T("lskdf;las");
    CString filename = _T("E:\\c++\\mfc\\us\\temp.txt");
    CString str; //获取系统时间
    SYSTEMTIME st;
    GetLocalTime(&st);
    str.Format(_T("%2d 日 ,%2d:%2d:%2d "), st.wDay, st.wHour, st.wMinute,
st.wSecond);
    //str = str + cCur + _T("\n");
    str = _T("slkdfa;sd");
    */
    outData =
_T("+*2.B-16732_•+*2.B-1673103M67AGJ8674OUT070912,ZBSN,ZSHC,106
————————————$P•+*2.B-16733_•");

    /*SYSTEMTIME st;
    CString str;
    GetLocalTime(&st);
    str.Format(_T("%2d day,%2d:%2d:%2d "), st.wDay, st.wHour, st.wMinute,
```

```
st.wSecond);
	str = str + outData + _T("\n");
	CStdioFile file;
	CString filename = _T("E:\\c++\\mfc\\us\\temp.txt");
	if
(!file.Open(filename,CFile::modeCreate|CFile::modeWrite|CFile::modeNoTruncate))
	{
		MessageBox(_T("shibai"));
	}
	file.SeekToEnd();
	file.WriteString(str);
	file.Close();
	MessageBox(_T("完成"));*/
}
```

usDlg.h 文件

```
// usDlg.h :  头文件
//

#pragma once
#include "afxcmn.h"
#include "mscomm1.h"
#include "afxwin.h"


// CusDlg  对话框
class CusDlg : public CDialogEx
{
//  构造
public:
	CusDlg(CWnd* pParent = NULL);	//  标准构造函数

//  对话框数据
	enum { IDD = IDD_US_DIALOG };

	protected:
	virtual void DoDataExchange(CDataExchange* pDX);  // DDX/DDV  支持


//  实现
protected:
	HICON m_hIcon;

	//  生成的消息映射函数
	virtual BOOL OnInitDialog();
	afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
	afx_msg void OnPaint();
	afx_msg HCURSOR OnQueryDragIcon();
	DECLARE_MESSAGE_MAP()
public:
	CListCtrl m_programLangLis;
	CListCtrl m_programLangList;
	afx_msg void OnInputData();
	afx_msg void OnDecode();
	afx_msg void OnShowData();
	int m_portnum;
	int m_baudR;
	CMscomm1 m_common1;
```

```
    DECLARE_EVENTSINK_MAP()
    void OnOncommMscomm1();
    CListBox m_ListWords;
    afx_msg void OnTimer(UINT_PTR nIDEvent);
    afx_msg void OnstopInput();

    afx_msg void OnBnsavefile();
};
```

附录 B:外文资料翻译