版权声明

摘要

为解决地震预报这一难题，北京大学地震监测预测技术研究中心研发了多分量地震监测预测系统 AETA，简称 AETA 系统。AETA 系统自 2015 年小批量试制以来，经过多次软硬件迭代，目前布设规模已接近 230 套，采集了约 18TB 左右的电磁和地声监测数据。随着地震预报工作的进一步深入开展以及断裂带监测项目的启动，基于抽样采集的 AETA 系统软件已不足以满足 AETA 项目当前的研究需求。因此需要重新设计并开发一套连续采集版本的 AETA 系统软件，并结合 AETA 系统软件运行多年来出现的问题，对 AETA 系统软件进行系统优化设计与实现。

本文对 AETA 系统的数据处理终端软件和探头数据采集软件进行了优化、设计与实现，并对软件进行了上线测试，主要完成的工作点和创新点如下：

(1) 在 Linux 平台下，采用 C++语言完成了数据处理终端软件的开发；基于 STM32 硬件平台，采用 C 语言完成了探头数据采集软件的开发。

(2) 设计了基于位重组思想的数据压缩算法对地震监测数据进行压缩处理，有效地降低了数据的传输带宽与存储成本。

(3) 采用生产者消费者模型对接收模块、解包模块、滤波模块进行优化设计，提高了 CPU 的利用率。

(4) 采用线程池技术对发送模块进行优化，可以增大 HTTP 数据传输的超时时间，提高数据传输的稳定性，同时降低了服务器的压力。

(5) 为应对终端的高并发请求，对 server 应用程序进行了优化，通过调度模块将不同的业务请求调度到不同的线程池中进行处理。

(6) 开发了模拟探头和模拟终端等测试工具并使用该工具对 AETA 系统软件进行了上线前的测试。

实际运行证明，本文设计的连续采集版本 AETA 系统软件能够可靠、稳定地运行，并且有效地支撑地震预报与断裂带活动监测的研究工作。

关键字：地震预报，抽样采集，连续采集，系统优化，测试

The Software Optimization Design and Implementation of Seismic Monitoring and Prediction System AETA

Kangsheng Zhou(Microelectronics and Solid-State Electronics)
Directed by Xing Zhang and Xin'an Wang

ABSTRACT

In order to solve the problem of earthquake prediction, the Peking University Earthquake Monitoring and Prediction Technology Research Center has developed a multi-component seismic monitoring and forecasting system AETA, referred to as the AETA system. Since the AETA system was trial-produced in small batches in 2015, after several hardware and software iterations, the current layout has reached nearly 230 sets, and electromagnetic and ground sound monitoring data of about 18TB has been collected. With the further deepening of the earthquake prediction work and the start of the monitoring of the fault zone activity, the AETA system software based on sampling acquisition is insufficient to meet the current research needs of the AETA project. Therefore, it is necessary to redesign and develop a continuous acquisition version of the AETA system software, and combine the AETA system software to run the problems that have occurred over the years, and optimize the design and implementation of the AETA system software.

In this paper, the data processing terminal software and probe data acquisition software of AETA system are designed, optimized and realized, and the software is tested online. The main work points and innovations are as follows:

(1) Under the Linux platform, the development of data processing terminal software is completed by C++ language; based on STM32 hardware platform, the development of probe data acquisition software is completed by C language.

(2) The data compression algorithm based on bit transformation is designed to compress the seismic monitoring data, which effectively reduces the data transmission bandwidth and storage cost.

(3) Optimize the design of the receiving module, the unpacking module and the filtering module by using the producer consumer model, which can improve the utilization rate of the CPU.

(4) Using the thread pool technology to optimize the sending module, can increase the timeout of HTTP data transmission, improve the stability of data transmission, and reduce the pressure of the server application server.

(5) In response to the high concurrent request of the terminal, the server application is optimized, and different service requests are scheduled to be processed into different thread pools by the scheduling module.

(6) Developed test tools such as analog probes and analog terminals and used the tool to test the AETA system software before going online.

Practice shows that shows that the continuous acquisition version of the AETA system software designed in this paper can operate reliably and stably, and effectively support the research work of earthquake prediction and fault zone activity monitoring.

KEY WORDS: Earthquake prediction, Sampling acquisition, Continuous acquisition, System Optimization, software test

目录

第一章 绪论

1.1 课题背景和意义

1.1.1 背景概述

地震对人类社会造成了巨大的社会经济损失。据统计，全球因自然灾害而死亡的人数中，有 54%是由地震导致的[1]。我国是地震频发且受灾最为严重的国家之一[2,3]。自 1949 年以来，我国发生了 7 级以上地震十余次,包括 1954 年甘肃山丹地震、1976 年河北唐山地震、2008 年四川汶川地震、2010 年青海玉树地震、2013 年四川雅安地震和 2017 年九寨沟地震[4,5]。其中河北唐山 7.8 级地震一共造成 24.2 万人死亡，汶川 8.0 级地震造成了 69000 人死亡[6,7]，青海玉树 7.1 级地震造成 2698 人死亡。

地震预报是指在地震发生之前比较准确地预报地震三要素（即发震时间、发震地点和地震等级）[8]。准确的地震预报能够让人类社会提前做好防灾准备，极大程度地降低地震带来的人员伤亡。因此，地震预报很早就成为了学术界的关注重点。国内外地震预报研究的方法种类多样，其中部分国内外学者尝试从地球的地质构造和地壳运动这一角度出发，通过研制相应的观测仪器，建立观测体系，探索地震发生的机理，从而预测地震[9-15]。此外，还有地震学者通过地震监测的方法来预测地震。地震监测是指利用传感设备来探测与地震相关的某些信号分量的变化，通过对这些信号分量的分析，试图找出这些信号的变化与地震发生事件的关系，从而预测地震的发生[16-20]。

我国对地震预报的大规模探索始于 1966 年邢台 7.2 级大地震发生之后。在 1966 年-1976 年期间，我国出现了十余次 7.0 级以上地震，同时也积累了一定的地震观测数据资料。1971 年，中国地震局成立，1975 年，我国预报成功了海城 7.3 级地震，给地震预报探索者带来了信心和鼓舞。然而，1976 年河北唐山 7.8 级大地震的预报失败，以及 2008 年汶川 8.0 级特大地震在没有任何预测信息的情况下发生使得地震预报陷入了困境[21]。地震的准确预报问题仍然是一个亟待解决的世界性难题。20 世纪末至 21 世纪初，计算机和信息技术开始蓬勃发展，中国地震局开始布局建设由国家地震台网、区域地震台网和流动地震台网组成的大规模数字地震台网观测系统[22]。在此期间，国内地震学者对地震预报的研究一直未曾间断，我国对地震预报的研究主要通过地震监测的方法，通过对地震前兆异常信号进行分析来实现地震预报的目标[23]。早在 1966 年，我国就正式采用钻孔手段来观测地应力的变化[24-26]。池顺良等[27]发明的钻孔应变仪成功探测到了汶川大地震清晰的长时间应变前兆信号。随着空间信息技术的发展，GIS[28,29]、GPS[30-33]、InSAR[34,35]等新型观测方法也逐渐投入到地震研究中来。地下流体

（比如地下水，气，油等）也是地震预报的重要观测量。我国第一个水位台网于 1979 年开始建设，在地下流体监测这一领域积累了丰富的经验，对地下流体的监测体系已较为完善[36-38]。此外，电磁异常信号也是很重要的地震前兆信号观测量。国内地震学者通过岩石破裂实验证实了地震或会产生电磁现象[39-41]。

目前发现的地震前兆信号种类较多，相应地，地震监测设备类型也多种多样，可以监测的异常信号包括声、光、电、磁、应力、气体和地下流体等。虽然这些信号分量已经在多次地震中出现异常变化，但是由于这些设备的监测效果受设备的稳定性、一致性、安装成本和安装环境制约，难以大规模高密度地布设并对一定区域进行较为全面系统实验，导致地震前兆与地震在时间和地点上难以一一对应，难以发现地震事件与前兆信号量的确定关系。因此，相关学者指出，地震监测手段需要新技术[42,43]。

为尝试解决目前地震监测预测领域面临的问题，北京大学深圳研究生院地震监测预测技术研究中心于 2010 年开始启动 AETA 项目。AETA 项目组自研了一款低成本，可在大区域内高密度布设的地震监测设备，在四川，云南，河北等地区建立了一套完善的地震监测网络。截止 2018 年底，AETA 系统监测设备累计布设 230 余套，覆盖四川、云南、首都圈的大部分地区。在地震预报方面，AETA 项目组研究人员使用大数据分析和机器学习建模等方法对采集到的地震监测数据进行研究，期望发现地震事件与前兆异常信号的关系，并最终准确预报地震。

1.1.2 AETA 系统概述

在地震的孕育和发生过程中，会有包括声、光、电和气体等信号分量异常的发生。这些分量均可以作为地震前兆监测的信号指标[8]。AETA 项目旨在研制一款低成本，对布设环境要求低，可在大区域内进行高密度布设的地震监测设备。因此，选择何种观测分量是进行系统设计的关键一环。

表 1.1 AETA 系统监测指标

探头类型	监测频段（HZ）	灵敏度	低频采用率(HZ)	全频采样率(HZ)
地声	0.1~50k	3 LSB/pa	500	150k
电磁	0.1~10k	20mV/nT	500	30k

地声是由岩石挤压产生的声音震动信号，从我国自古以来的地震记录资料来看，都存在与地声有关的记录；现代诸多震例也表明，地声信号是很重要的地震前兆信号[44-48]。在地震的孕育过程中，岩石破裂可能会产生压磁效应和压电效应，导致电磁辐射源产生变化，从而产生电磁辐射。国内外相关试验表明，岩石破裂产生的电磁辐射具有

较宽的频谱，囊括了电磁波信号的各个频段[49-51]。因此，AETA 系统监测的各项指标定义如表 1.1 所示。

AETA 系统由地声传感探头、电磁传感探头、数据处理终端，及云服务器平台和数据分析系统组成[4,8,52-56]，如图 1.1 所示。

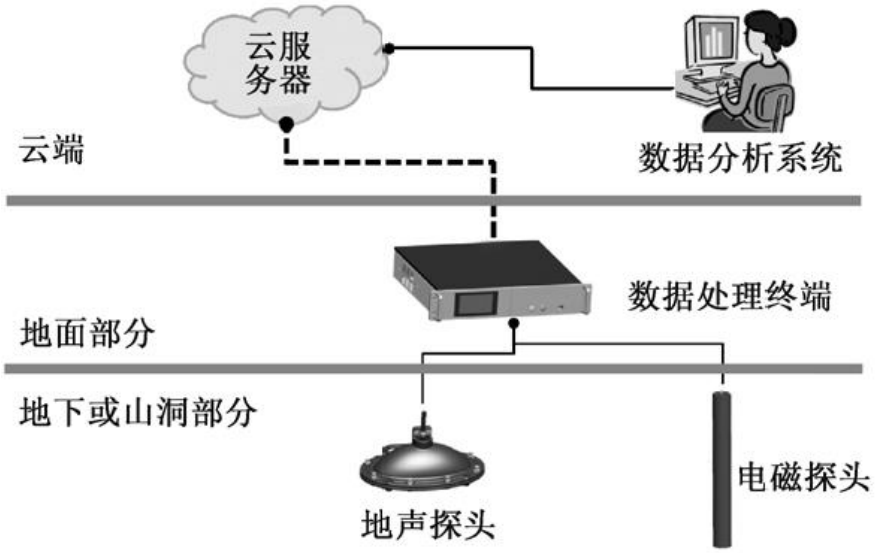

图 1.1　AETA 系统示意图

传感探头感知来自地下的电磁扰动和地声信号，数据处理终端实时采集数据并通过互联网（有线或无线）将数据传输到云平台进行特征提取、持久化存储和异常分析等。AETA 系统于 2015 年 8 月完成了第一版设备的小批量试制，快速迭代后于 2016 年 6 月完成了第二版设备 20 套的小批量生产。在中国地震局监测预报司的支持以及河北、四川、云南、北京地震局的协助配合下，两版设备均进行了现场布设实验。现场实验显示，AETA 系统对地震前兆信号具有较好的捕捉效果，系统灵敏性、稳定性和一致性得到初步验证。2016 年底 AETA 项目与专业硬件服务商深圳卓翼科技达成深度合作，AETA 多分量地震监测系统正式定型批量生产。

截止目前，在中国地震局的支持下，AETA 系统布设范围已覆盖了河北、四川、云南、西藏、广东和台湾等地区，其中在四川布设密度最大，布设数量已达 100 余套，基本覆盖四川全境重点区域。图 1.2 为 AETA 系统的四川站点分布图。

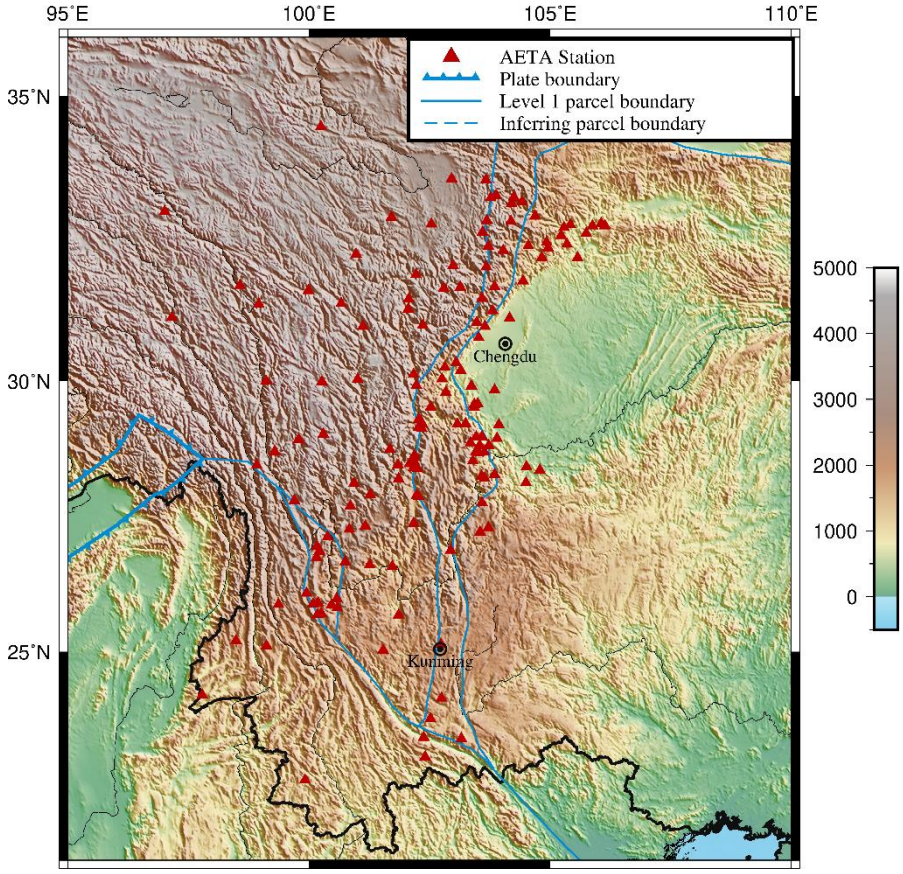

图 1.2　AETA 系统四川站点分布图

1.1.3　AETA 数据采集软件的迭代过程和面临的问题

AETA 项目自立项以来，按照低成本、抗环境干扰能力强，可在大区域内高密度布设的设计要求，自研了软硬件设备。硬件设备包括电磁传感探头、地声传感探头和数据处理终端。其中电磁传感探头和地声传感探头安装于地下两米处或者放置于山洞中，数据处理终端需放置在具备电力和网络供应的环境中。数据采集软件包括运行于传感探头内的探头数据采集软件，运行于数据处理终端内的数据处终端软件以及运行于阿里云服务器的 server 应用程序。其中探头数据采集软件主要完成传感数据的采集、打包和发送。数据处理终端主要完成对两个传感探头数据的实时接收、处理和发送。数据处理终端还要接收并处理来自 server 应用程序的指令，并且实现对探头的控制管理。云服务器负责汇总 AETA 系统各个站点中数据处理终端发上来的数据，在对数据进行特征值的处理后，将原始数据和特征数据做持久化存储（Mysql 数据+本地磁盘阵列）。

AETA 系统自 2015 年 8 月小批量试制到现在，软硬件进行了多次迭代改进。项目开始之初，地声传感器和电磁传感器集成在同一个探头内，硬件上采用 FPGA 作为计算单元，并用 Verilog 语言开发探头探头数据采集软件，用于传感数据的接收、处理与

发送。数据处理终端采用广州致远电子的 AM3352 工控板，其核心处理单元是 32 位的 Cortex-A8 处理器，内部移植了精简的 Linux 操作系统。为了尽快验证监测设备的映震效果，快速开发出了第一版数据处理终端软件，该软件采用单线程架构，完成了对探头数据的接收、处理与发送。2016 年 3 月对硬件进行了改进，将电磁传感探头和地声传感探头分开设计，内部采用 STM32F407 单片机作为核心计算单元，软件代码采用 C 语言实现。此外，设计了第二版数据处理终端软件，采用多线程设计，增加了数据处理终端与探头和服务器之间的交互，极大地降低了运维成本。

然而，随着 AETA 系统布设规模的扩大，第二版的数据处理终端软件也慢慢体现出了它的不足之处。AETA 系统监测站点多数布设于四川、云南等地，部分区域电力和网络设施不太完善，经常面临长时间的停电和断网等突发情况，如何避免断网情况下数据的丢失问题是数据处理终端软件需要处理的问题；数据处理终端采用有线网络传输数据，然而，由于有线网络安装较为麻烦且部分区域不支持有线网络的安装，因此数据处理终端还需要兼容有线和无线两种数据传输方式；降低设备的维护难度也是终端软件的设计需要考虑的因素，AETA 系统监测站点的维护人员多数为地震台站工作人员或者乡镇工作人员，数据处理终端运行出现问题，只能由维护人员断电重启，无法由专业的 AETA 工作人员进行远程操作和故障诊断；数据处理终端软件偶尔出现运行不稳定的问题，如探头掉线，软件卡死等现象。数据是 AETA 系统最宝贵的资源，一旦出现软件运行不稳定的现象，将严重影响采集数据的质量，因此数据处理终端的稳定性还有待提高。

此外，相关研究表明，地震的前兆信号持续时间较长，因此在实验初期阶段，数据处理终端采用抽样采集的方式，即每 3 分钟连续采集 1 分钟的低频数据和 1s 的全频数据。第一版和第二版的数据采集软件均为抽样采集方式，基于这种采集方式，AETA 系统在云南、四川、西藏、河北、北京、广东、台湾等地区布设了 200 余个站点。结合这些站点数据和相应的震例进行分析，研究表明，AETA 系统能够在一定程度上反映映震效果。在 AETA 系统地震预报取得一定效果之后，为满足数据分析的需求，采集更多的连续信号用于地震预报和断裂带活动监测的研究，需要将数据的采集方式从抽样采集优化为连续采集。

AETA 系统已经运行四年左右的时间，项目组在地震数据分析上也有了一定的进展。在这个时间节点上，针对数据处理终端运行多年来所面临的一些问题，并结合项目组在 Linux 嵌入式软件开发上所积累的一些经验，对数据处理终端软件进行优化与重构，并将数据采集方式由抽样采集优化为连续采集，进一步提高软件运行的稳定性，降低运维成本，为 AETA 系统提供更加优质的数据，对 AETA 系统的稳定运行、断裂带活动监测的开展以及地震预报研究工作的稳步推进具有重要的意义。

1.2 国内外研究现状

地震监测数据采集软件系统是一种典型的物联网数据采集软件系统，在这一领域，国内外相关学者做出了丰富的研究。

杜静怡，李文涛等采用 IN-6358 数据采集卡，基于 LabVIEW 工具设计了一种多通道高速数据采集软件系统[57]。该系统在软件的设计与实现上采用块编程技术，使得各个模块之间功能，易于修改和扩展。此外，状态机和生产/消费系统的结合极大地提高了系统的采用率。

NazireMerveÖnder,MuhammetCanÇakmak 等采用 FPGA 设计了一种集数据转换、收集和传输于一体的数据采集系统，并通过 VHDL 语言实现了设计[58]。该系统的特点是处理速度块，安全可靠。

林海林，张智胜等设计了一个可配置的自动灌装线数据采集系统[59]。该系统基于 ARM 平台，由数据采集，服务器和云平台三个部分组成；数据采集部分负责采集数据并将数据发送到主机。服务器部分用于从主机接收数据并将其保存在数据库中。云平台为用户服务并提供数据分析。为了确保数据的稳定传输，为不同部分之间设计了相应的通信协议。

魏绍，郭浩然等提出了一种基于 STM32 的 ZigBee 与以太网智能网关的物联网信息获取系统[60]。该网关系统采用双 MCU 设计，在软件实现上，通过调用 CGI（通用网关接口）和 SSI（Sever Side Include）技术可以实现上下位机之间的联动, 此外还采用了 FIFO 输入流控制策略用于缓冲数据流，既确保了数据存储效率又避免了数据的丢失。

Sailee Pawar,VV Shete 采用 Cyflex 软件，基于 Linux 平台为发动机测试设计和构建了一个易于配置的数据采集系统（DAS）[61]。该系统具有低成本，易维护，功耗低等优点。

Mohamed Sarrab 指出了 IOT 应用软件潜在的一些不安全问题，包括不安全的接口，源代码漏洞，约束应用程序协议和中间件安全性等，提出在 IOT 软件开发过程中需要从软件安全角度进行考虑[62]。

在地震数据采集系统设计方面，裴高，李志华等设计了一种分布式三分量地震数据采集系统[63]，该数据采集系统采用 LoRa 远程无线通信技术，有效地避免了以太网有线传输方式电缆繁琐，布线复杂，相邻传输线之间的串扰和噪声等缺点。

Ozkan Kafadar 和 Ibrahim Sertcelik 等开发了一种多通道地震监测和数据采集嵌入式系统，并使用 Microsoft.NET Framework 开发了基于 Windows 操作系统的图形用户界面，用于传输，处理和分析数据[64]。该系统可方便地应用于不同的工程应用中。

中国地震台网中心的黄经国，李正媛等基于 SFTP 技术设计了 GNSS 数据采集软件，该软件的传输方式采用了 SFTP 文件传输协议，保证了数据产品的稳定和可靠传输

[65]。

吉林大学的朱倩钰采用动态电源管理和动态频率调节等技术，对自主研发的GEIWSR 地震数据采集系统进行了软件层面的优化，降低了系统的功耗[66]。

中海油的谢荣清设计了一个适用于海上地震数据采集的软件，该软件采用多线程设计，运行于 VxWorks 操作系统，采用 DMA 数据传输方式，具有高效传输性、高实时性和高可靠性等特点[67]。

电子科技大学的汪超设计了一种基于 GPRS 技术的数据采集终端系统，根据设计需求对嵌入式实时操作系统内核进行修改[68]，满足了系统低功耗的要求，同时采用GPRS 网络数据传输方式，可以很大程度上避免因意外（如地震）导致网络中断数据无法及时传输的问题

在地震监测软件系统方面，德国 GEO-FON 台网开发了 SesComP[69]；该系统支持MySQL、PostgreSQL、SQLite 等数据接口，可以统一管理地震台网处理的数据[70]。

当前，国内外学者对物联网数据采集软件的研究主要集中在数据的采集方式和传输方式，在物联网软件的安全性、稳定性和容错性等方面研究不多。AETA 系统采用实验室自研的传感探头和终端设备，在汲取国内外现有研究成果的基础上，需要结合AETA 系统自身的业务特点和需求，研发一款适用于 AETA 系统的地震监测应用软件。该 AETA 系统软件不仅要完成数据的采集与传输，同时要更加注重软件的稳定性、可靠性、鲁棒性与容错性等。

1.3 论文组织架构

本文以 AETA 系统的软件优化设计与实现为核心，从软件的详细设计，软件的优化和软件的测试三个角度展开描述。

全文第一章为绪论，分析了 AETA 系统的研究背景、技术指标、系统结构和发展历程，总结了 AETA 系统软件历史迭代过程与当前面临需求问题。对国内外地震监测系统的研究现状进行了调研后，提出要结合 AETA 系统本身的特点和需求，研发一款适用于 AETA 系统的地震监测软件。

第二章结合 AETA 系统软件运行的硬件平台，分析了传感探头和数据处理终端软件的功能需求，并根据功能需求，设计了相应的软件框架。

第三章对连续采集数据处理终端软件和探头数据采集软件的设计与实现进行了详细的描述。

第四章对 AETA 系统软件进行了优化，包括引入了压缩算法对数据进行压缩传输与存储，基于生产者消费者模型对接收、解包、滤波模块进行优化，基于线程池技术对

发送模块进行优化和日志管理系统的优化。此外，为保证数据处理终端的可靠性与容错性，从监控机制、弱网络环境的处理和数据可靠性保证三个角度来进行了设计。

第五章为软件的上线与测试。开发了模拟终端和模拟探头两个测试工具，设计了测试方案和测试评价标准，并利用测试工具搭建了测试平台，对 AETA 系统软件进行了上线前的测试。

第六章对全文进行了总结，并提出了对 AETA 系统软件设计的未来展望。

第二章 AETA 系统软件的需求分析与总体设计

充分而完备的软件需求对软件设计与实现具有重要的指导作用。软件的运行离不开硬件的支撑，选择符合要求的硬件设备是软件设计中需要考虑的重点。为了保证 AETA 系统软件能够完成 AETA 系统的业务需求，需要对 AETA 系统的业务特点进行分析，选择合适的硬件资源。在已选定硬件的基础上，根据 AETA 系统业务需求和软件设计的稳定性、可靠性、鲁棒性等要求提出 AETA 系统软件的功能需求，最后根据软件的功能需求设计 AETA 系统软件的主体架构。

2.1 AETA 系统业务特点分析

传感探头、数据处理终端和数据接收服务器构成了 AETA 系统的数据采集链路。其中传感探头和数据处理终端的硬件由实验室自主研发，数据接收服务器采用阿里云服务器，可通过官网选型快速购买不同配置的服务器。本节主要讨论传感探头和数据处理终端的业务特点。

传感探头负责完成模拟信号到数字信号的转化，并将经过 ADC 采集到的二进制数据进行打包并发送至数据处理终端。由 1.1.2 节可知，AETA 系统传感探头 ADC 的采样率最高可达 150kHz。由于传感探头采样率高的特点，因此传感探头内的主控单元需满足处理速度快的要求，且包含定时器、支持 SPI 协议等用于 ADC 数据的采集。此外，由于传感探头对存储数据的要求较小，因此对主控单元的存储性能要求较低。由于传感探头需要完成的任务较为单一，不需要多任务并发执行，因此主控单元选用计算性能合适的单片机即可。此外，为屏蔽外界干扰，传感探头多数安装在地下两米处，但数据处理终端需要放置在具有网络和电力供应的位置。探头与终端的分离设计也是 AETA 系统的一大特点，需要选取一种稳定的通讯方式用于传感探头和数据处理终端之间的数据交互。

数据处理终端既要完成电磁探头和地声探头的数据接收，同时又要完成数据的解包、压缩、滤波等处理工作，最后将处理得到的数据发送至服务器，是一个多任务并发执行的系统，因此数据处理终端需要选取带有操作系统的主控单元。此外，分布于全国各地的数据处理终端设备需要将数据发送至服务器，因此数据处理终端主控单元还需要支持以太网接口。

2.2 AETA 系统的硬件选型

2.2.1 传感探头的硬件选择

由 2.1 节可知，可选择一款计算性能较高的单片机作为传感探头的主控单元。当前，比较常用的单片机有 51 单片机，AVR 单片机，MSP430 单片机，STM32 单片机等。通过实际调研发现，51 单片机功能简单，运行速度较慢，适用于对性能要求不高的场合；AVR 和 MSP430 单片机性能强，功耗低，开发工具使用便捷，但是成本较高；STM32 单片机是近些年发展迅速的一类单片机，具有高速度、高性能、低功耗等优点，功能强大，能够满足大部分单片机设计需求，且成本适中。因此，通过比较，选择 STM32 系列单片机中的 STM32F407 作为传感探头的主控单元。此外，由于探头与终端的分离设计，还需要为传感探头和数据处理终端之间选择一种通信方式。目前短距离通信方式有蓝牙、Wi-Fi 技术、ZigBee 技术、有线网络通信技术等。其中蓝牙和 ZigBee 技术的数据传速度较慢，无法达到 300KB/s 的传输速度，而 Wi-Fi 技术传输速度较快，但安全性较低，因此选用有线网络通信方式用于探头与数据处理终端之间的通信。需要选取一款网络模块传输芯片用于探头和数据处理终端之间的网络传输，使得双方之间通过 TCP/IP 协议进行通信。

STM32F407 是一款 32 位，采用 LQFP100 形式封装的微处理器，FLASH 和 RAM 的大小分别为 1MB 和 192KB，主频为 168MHz，工作电流为 38.6mA。STM32F407 有五个时钟来源，分别为 HIS(高速内部时钟源),HSE(高速外部时钟源),PLL(锁相环),LSE(低速外部时钟源),LSI(低速内部时钟源)。STM32F407 性价比高，正常工作时系统功耗低，能够很好地符合 AETA 项目的需求。

W5300 是一款用作网络模块传输的芯片，它内部集成了 10/100M 以太网控制器，MAC 和 TCP/IP 协议栈。W5300 内置了 TCP/IP 内核，通过将 WIZnet 网络协议的硬件进行数字逻辑化实现的。该内核含有 PPPOE 协议、ARP 协议、IP 协议、ICMP 协议等，能够有效地解决数据的网络传输问题。通过 STM32F407 外接 W5300 芯片，可以实现传感探头和数据处理终端的有线网络通信。

2.2.2 数据处理终端的硬件选择

根据 AETA 系统的业务特点，数据处理终端需要选取一款带有操作系统，支持以太网接口，且具有一定运算速度的主控单元。ARM 嵌入式核心板在物联网领域应用广泛，它能够很好地满足 AETA 系统的设计需求。通过市场调研，最终选取了广州致远电子公司的 M3352 工控板作为数据处理终端的主控单元。

M3352 工控板中的 ARM 核心板的关键技术指标为：32 位的 Cortex-A8 处理器，

主频为 800MHz，128MB 的 DDR2 内存，256MB 的 NAND Flash，此外还具有两路以太网通信接口。两路以太网通信接口分别负责终端与探头、服务器之间的网络通信。终端适配器的输出电压为 24V，电源模块负责给 M3352 工控板和交换机提供适配的工作电压。由于一个标准台站通常具有电磁和地声两个探头，而 M3352 工控板只能提供一路以太网接口用于与探头进行通信，另一路以太网接口用于与服务器进行通信，因此需要交换机将两个传感探头的数据进行汇总，经由以太网接口发送至 Cortex-A8 处理器。

终端采用标准的 2U 机箱设计，可以方便地放置于机柜之中。终端面板左边是由液晶、单片机和存储器三部分组成的 GPU45A 串口屏。终端面板右边为复位键和电源键，复位键旁边是 USB 插口。终端背部从左至右分别是两个探头的网线插口，网络通信接口，2 个 USB 插口，一个调试串口和一个数据串口。机箱内部由 M3352 工控板、电源模块电路和交换机组成。

2.3 AETA 软件系统的需求分析

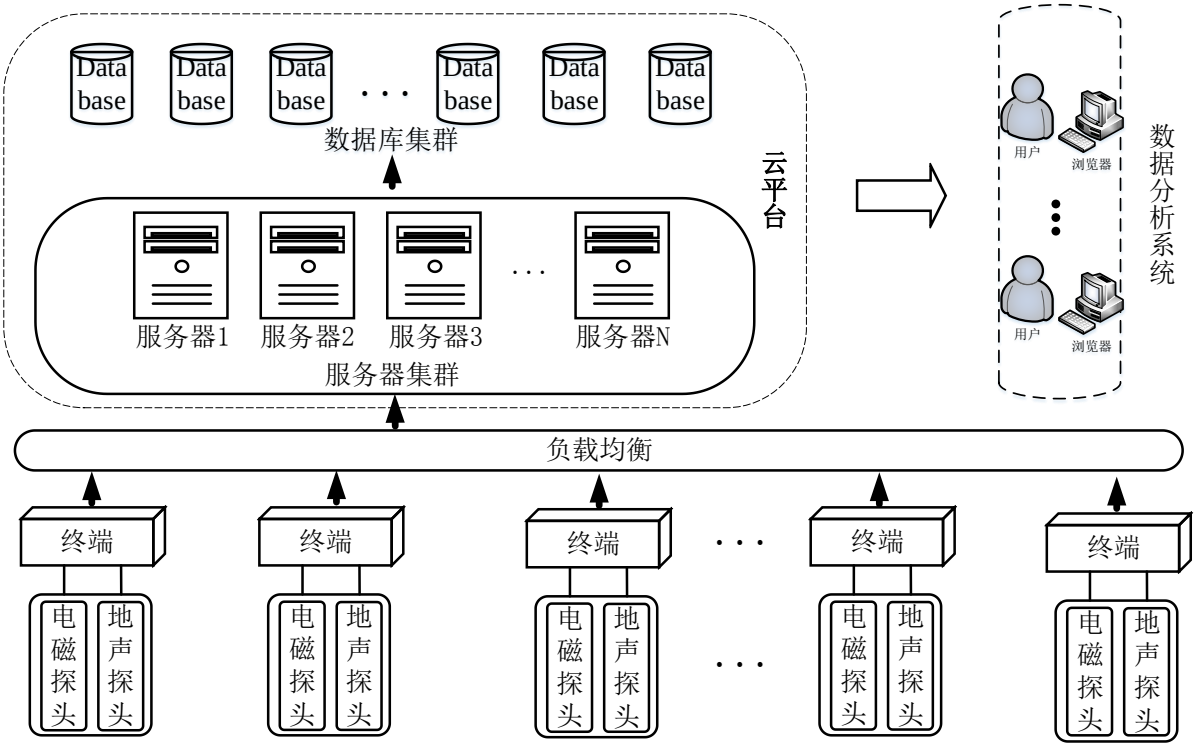

图 2.1 AETA 系统框架

图 2.1 为 AETA 系统的框架。由图 2.1 可以看出，AETA 系统软件主要分为 3 个部分，运行于传感探头的探头数据采集软件，运行于数据处理终端的终端软件，运行于云

服务器端的 server 应用程序软件。其中数据采集软件和终端软件由 AETA 项目组自行研发，server 应用程序由 AETA 项目的合作伙伴武汉大学计算机学院相关实验室研发。

2.3.1 探头数据采集软件的需求分析

结合 AETA 系统的业务特点、硬件资源，提出了 AETA 系统探头数据采集软件的需求，关键点如下：

(1) 按照电磁传感探头 30k/Hz,地声传感探头 150k/Hz 的采样率，从 ADC 中读取采样数据。

(2) 对采集到的数据按照 1024 个点一个包，按照约定好的探头与终端之间的数据传输协议进行打包，每满一个包，立即发送至终端。

(3) 除了向终端发送传感数据外，每隔一定时间向终端上传自身运行的状态、参数等，包括软件版本，参数同步间隔，调零放大系数。

(4) 接收来自终端软件的命令，包括重启、参数变更等。

(5) 探头软件要具备升级功能。探头每次重启，需要检查当前版本的探头软件是否为最新版本，如果当前版本为旧版软件，则需要完成软件的升级操作之后，再执行业务功能。

2.3.2 数据处理终端软件的需求分析

由图 2.1 可知，数据处理终端在整个 AETA 系统中处于中间环节，是整个系统的关键节点,所以一个完善合理的终端软件需求关乎整个 AETA 系统的正常运作。结合 AETA 系统的业务特点、硬件资源，提出了连续采集数据处理终端软件的设计需求，关键点如下：

(1) 数据处理终端软件与探头之间采用 C/S 模式设计，基于 TCP/IP 这一传输层协议，按照约定好的应用层数据协议进行通信。具备支持多个探头并发请求的能力，且为不同探头设置不同的缓冲区存储待解包数据。能够向探头下发命令、参数等消息。

(2) 数据采集方式为连续采集。需要有全频连续采集和低频连续采集两个版本。对于全频连续采集，终端将接收到的所有数据进行压缩处理后发送至 server 应用程序。对于低频连续采集，终端将接收到的所有数据进行滤波后的到的低频数据全部发送到 server 应用程序。

(3) 数据处理终端软件要与 server 应用程序进行通信。终端软件需要主动向 server 应用程序发送数据，且要支持日志上传、命令查询与请求、配置请求、状态更新、时间同步和实时发送警告等功能。终端软件采用 HTTP 协议，按照约定好的数据协议向 server 应用程序发起不同类型的数据请求。

(4) 按照终端与探头之间的应用协议进行数据解包；不同探头的数据应该按照数据类型严格分隔开，互不影响。为防止数据处理流程阻塞，解包速度要大于数据的接收速度。

(5) 数据滤波。由于滤波算法的输入数据为数据流，所以滤波算法要求连续滤波，保证滤波后得到的数据仍然为低频连续数据。

(6) 数据压缩。由于全频连续采集数据量较大，对数据传输带宽和数据存储要求较高，因此需要设计一种合适的压缩与解压缩算法，将接收到的数据进行压缩后再传输到 server 应用程序。

(7) 弱网络环境下可将数据存储在终端设备中的 SD 卡内，且在网络环境优良的条件下，能够取出 SD 卡内的数据进行补传。由于 SD 卡内要存储不同时间不同探头的数据，所以 SD 卡存储模块在设计时要满足数据读取时能够快速识别数据属性（探头 ID、数据类型、时间戳等）。

(8) 终端显示屏需要直观地显示探头和终端的关键运行信息，因此屏幕界面需要进行合理地规划和设计。

(9) 支持软件在线升级。终端程序在重新启动的时候，需要向服务器下载版本信息文件并与终端内的信息文件进行对比，如果需要更新，则从服务器端下载相应的更新文件并替换指定目录下的旧文件。为避免软件升级失败的情况下导致终端服务停止，需要具备软件回滚到旧版本的功能。

(10) 监控数据处理终端软件的运行状况。在数据处理终端软件卡死或者奔溃的情况下可以自动重新启动程序或者操作系统。

2.4 AETA 系统软件的框架分析与设计

2.4.1 传感探头软件的框架分析与设计

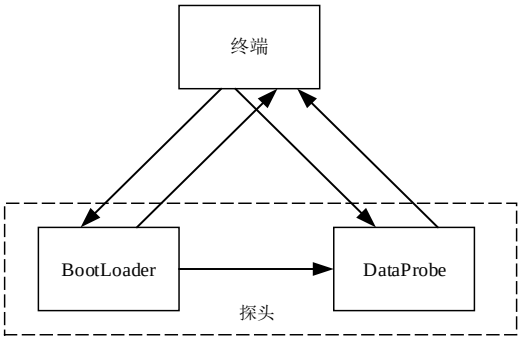

图 2.2 探头软件总体设计

在对探头软件的需求进行分析后，设计了图 2.2 所示的探头软件框架。其中 BootLoader 和 DataProbe 是运行在探头的核心计算元件 STM32F407 中的两个程序，均存储在 Flash 中。其中 BootLoader 为引导程序，主要完成对 DataProbe 的升级更新操作；DataProbe 为数据采集程序，完成传感数据的采集和发送功能。

整个软件运行流程如下：

(1) 探头上电启动后，启动 Flash 中的 BootLoader 程序。

(2) BootLoader 向终端下载探头软件版本信息文件。

(3) BootLoader 读取探头软件版本信息并与存储在 Flash 中的版本信息对比，确定是否需要更新 DataProbe。

(4) 如果需要更新 DataProbe，则从终端下载新版本的 DataProbe 并替换 Flash 中的旧固件，然后再运行 DataProbe。

(5) 如果不需要更新 DataProbe，则直接跳转运行 DataProbe。

2.4.2 数据处理终端软件的框架分析与设计

按照 2.3.2 节的需求，设计了图 2.3 所示的连续采集终端软件框架。其中 ShareMemory 为 SynApp 进程创建的共享内存区，主要用于 SynApp 和 DataTerminal 两个进程间的通信，存放进程的心跳信息和共享 sessionID。由于终端需要完成探头数据的接收、处理与发送，软件的远程升级，系统的自我监控以及日志数据的发送、时间同步等三大块功能。根据这三个功能，分别设计了三个应用进程（DataTerminal、UpdateApp、SynApp）来完成相应的任务。其中 DataTerminal 是终端内的主应用进程，主要完成对探头数据的实时接收、准确处理与安全发送三大任务。UpdateApp 主要负责终端文件的远程升级任务。SynApp 一方面要完成日志数据的处理以及时间同步等与业务数据相关

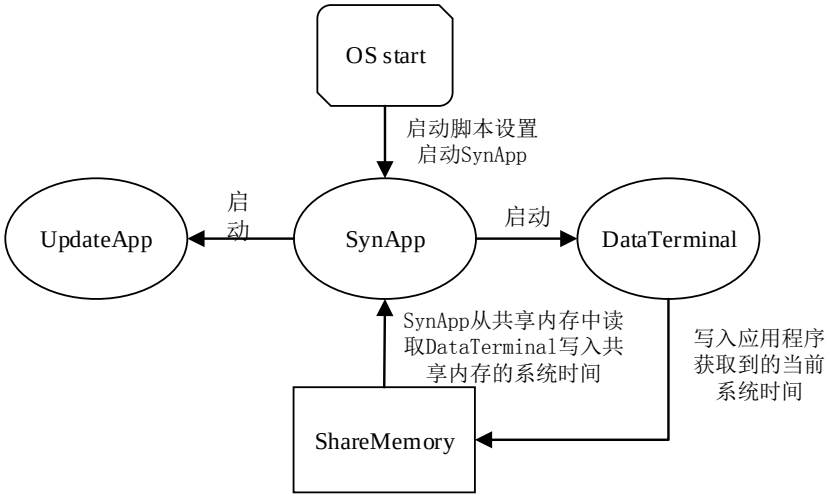

图 2.3 连续采集数据处理终端软件总体设计

性不大的任务，另一方面要负责对 DataTerminal 程序的监控。DataTerminal 是终端内最重要的应用程序，一旦程序运行发生异常崩溃或者卡死，SynApp 会将进程 kill 掉并重新启动 DataTerminal 进程。

结合图 2.3 分析整个终端系统的运行过程如下：

(1) 终端电源接通后，内嵌于核心板中的 Linux 系统开始启动。系统启动之后，按照提前部署好的启动脚本，如图 2.3 所示，启动 SynApp 进程，使得 SynApp 在系统启动后能够自动运行。

(2) SynApp 进程启动后，创建共享内存 ShareMemory，并进行自身的初始化和系统时间的同步之后，SynApp 查询系统进程表中是否存在 UpdateApp 和 DataTerminal 的进程 ID，如果不存在，则执行(3)，否则执行(4)。

(3) 若 UpdateApp 和 DataTerminal 的进程 ID 号不存在，则说明这两个进程没有启动或崩溃了。SynApp 则会启动 DataTerminal 和 UpdateApp。UpdateApp 进程在确定软件不需要更新或者在更新完终端软件之后，进程会自动释放，因此不需要向共享内存中写入心跳信息，受 SynApp 的监控。

(4) 若系统中存在 UpdateApp 和 DataTerminal 的进程 ID 号，则 SynApp 一方面去完成日志数据发送功能，另一方面周期性地从共享内存区中读取 DataTerminal 进程写入的心跳信息（即系统时间）。如果 SynApp 前后两次从共享内存区中读取到的系统时间不变，则说明 DataTerminal 进程卡死或者崩溃，SynApp 则会将 DataTerminal 进程 kill 掉，然后重新拉起 DataTerminal 进程。若相邻两次读取的系统时间在变化，则说明 DataTerminal 进程在正在运行。

(5) DataTerminal 进程在运行之后，会专门分配一个线程定时地往共享内存中写入系统时间，写入时间间隔要小于 SynAPP 读取共享内存的时间间隔。图 2.3 所展示的系统框架将系统功能分为三个子进程来完成，通过对上述三个进程之间运行逻辑的分析与介绍，可以论证，该框架设计能够满足终端系统的设计要求。

2.5 本章小结

本章首先分析了 AETA 系统的业务特点，然后根据 AETA 系统业务特点选择了合适的传感探头主控单元和数据处理终端主控单元，并对主控单元的硬件资源进行了介绍。结合 AETA 系统业务特点和硬件资源，提出了传感探头软件和数据处理终端软件的功能需求，并列出了软件要实现的功能。最后根据软件需求，设计了适合 AETA 系统的软件框架，并分析了基于这个软件框架下的系统运行流程。

第三章　AETA 系统软件的设计与实现

AETA 地震监测系统软件包含探头数据采集软件、数据处理终端软件和位于云服务器端的 server 应用程序。软件的稳定运行是 AETA 地震监测系统数据来源的可靠保证。结合系统软件运行四年多以来的经验，以及项目组对地震预报研究的需要，对 AETA 系统软件进行了优化，并进行了代码的重新设计和实现。本文主要介绍数据处理终端软件和探头数据采集软件的设计与实现。

3.1　原始数据的数据量分析与采集方式的选择

AETA 多分量地震监测预测系统的原始数据来源于传感探头。目前，传感探头按照 ADC 的采样频率分为高频和低频两个版本。第一章的表 1.1 列出来高频版本探头的相关技术指标，而低频版本的电磁探头和地声探头的采样率分别降低为 10kHz 和 30kHz。表 3.1 分别对不同种类的探头的数据量进行了计算与统计。

表 3.1　探头数据量统计

探头类型	采样率 (kHz)	数据量（MB/分）	数据量（GB/时）	数据量（GB/天）	数据量（GB/月）	数据量（GB/年）
高频电磁探头	30	3.43	0.201	4.82	144.6	1735.2
高频地声探头	150	17.15	1.005	24.1	723	8676
低频电磁探头	10	1.14	0.067	1.61	48.2	578.4
低频地声探头	30	3.43	0.201	4.82	144.6	1735.2

AETA 系统的每一个标准监测站点均由一台数据处理终端、一个电磁探头和一个地声探头组成。由表 3.1 可知，如果数据处理终端将接收到的高频探头数据全部发送至 server 应用程序并进行存储，每个监测站点每年需要接近 10TB 的带宽流量和存储资源；如果将 AETA 设备进行大区域高密度地布设，带宽和存储的成本将非常高。因此，在不影响监测数据分析的情况下，降低数据传输的带宽和存储要求，是系统设计必须要考虑的因素。

在系统设计之初，进行了小批量的现场实验。通过对现场实验采集到的电磁和地声数据进行分析，发现电磁和地声信号的变化是一个分钟级别的变化量。因此，在后来的软件设计中，采用抽样采集的方式，将接收到的数据按十分钟为一个时间窗口进行截取，即每十分钟抽取 1s 钟的连续数据作为全频数据发送至服务器，取 1 分钟全频数

据进行滤波得到低频数据并发送至服务器。通过这样处理，电磁和地声每小时的全频数据量大小为 351.56kB 和 1.72MB,极大程度的降低了系统对带宽和存储资源的要求。随着项目的深入研究，为了获取到更多的监测数据，在系统成本可承受的范围内，将抽样采集每十分钟一个抽样窗口变为 3 分钟一个抽样窗口，抽样方式不变，数据量变为原来的 3 倍。

AETA 系统按照上述抽样采集方式，运行了 3 年左右，基于这种抽样采集方式采集的数据在很多震例前反映出了异常。虽然地震前兆异常是分钟级别以上的变化量，但是抽样采集的方式在每一个抽样窗口丢失了 2 分钟的数据（每 3 分钟一个抽样窗口），丢失的数据中很可能包含有价值的地震前兆信息。此外，随着研究工作的推进，发现 AETA 系统对电磁和地声的监测数据在断裂带的测量工作中能够发挥重要的作用，但是对数据的连续性要求较高。因此，有必要将数据采集方式由抽样采集改成连续采集。然而，由表 3.1 可知，探头采集的数据量庞大，如果将数据采集方式改成连续采集，将对带宽和存储资源提出较大的挑战。随着数据分析工作的积累和深入研究，发现 AETA 系统电磁和地声的前兆异常主要集中在低频范围内（0~200Hz）。此外，国内外也有相关地震学者指出，由于低频信号穿透能力强，传输范围远，能够保持较宽的有效频带，因此低频信息在地震勘探中具有重要的作用[71]。基于此，在数据处理终端中，对接收到的探头数据进行连续滤波，可以得到数据量较小的低频连续数据，解决了连续采集的传输带宽和存储资源的问题。本文主要介绍连续采集软件的设计与实现。

3.2 数据处理终端软件的设计方案

为了进一步开展地震断裂带活动的研究，获取在时间序列上较为连续的监测数据，并且对抽样采集在软件性能上的不足进行优化，提高软件的鲁棒性，本文设计了连续采集数据处理终端软件 DataTerminal。连续采集终端软件最大的特点在于实时接收并处理探头数据，不再采用时间窗口抽样的方式采集数据，对所有接收到的数据进行滤波或者压缩处理后，按照 1 分钟一次的发送频率向 server 应用程序发送数据。DataTerminal 分为全频连续采集和低频连续采集两个版本。全频连续采集主要适用于采样率较低的低频版本探头，对接收到的探头数据进行压缩后，直接发往 server 应用程序，不过滤任何信息。低频连续采集既适用于高频版本的传感探头也适用于低频版本的传感探头，对接收至探头的数据，进行连续滤波后得到低频数据，然后将所有的低频数据发送至服务器。由于地震预测仍然是一个全球性的难题，无法确定地震前兆异常信号一定集中在低频（0~200Hz）范围内，因此在一些重点台站内部署全频连续采集版本的软件，与低频连续采集形成比对，以防丢失了重要的前兆异常信号。而低频连续采集版将在

AETA 地震监测预测系统中大规模推广使用。

3.2.1 连续采集数据处理终端软件的设计方案

按照 2.3.2 小节的需求分析，设计了图 3.1 和图 3.2 所示的连续采集数据处理终端软件 DataTerminal 的设计方案。由图 3.1 和图 3.2 可以看出，低频连续采集和全频连续采集在设计方案上基本一致，唯一的区别在于是否对接收到的数据进行滤波和压缩。以低频连续采集为例介绍 DataTerminal 的设计方案。探头与终端之间通过 TCP/IP 传输层协议进行通信，监听线程为 DataTerminal 的主线程，在指定端口持续监听是否有探头请求连接。一旦探头向监听端口发起连接请求，监听线程会创建一系列线程来服务于这一个连接，包括接收线程，解包线程和滤波线程；接收线程调用系统 API 从建立好的连接套接字中不断接收来自探头的传感数据，并将接收的数据放入接收流缓冲中。然后通过信号量（进程间通信的一种方式）通知解包线程从接收流缓冲中解包数据。解包线程按照约定好的数据格式对接收流缓冲中的数据进行解包，将解包好的数据放入解包流缓冲中；对于滤波线程，每次取 1s 钟的数据作为滤波算法的输入。因此，在解包流缓冲中的数据达到 1s 钟的数据量时，解包线程通过信号量通知滤波线程取解包流缓冲中的数据进行滤波；滤波线程每次取 1s 数据滤波得到低频数据，由于在系统设计时要求终端每一分钟向服务端发送一次数据，因此当滤波得到的低频数据达到 1 分钟

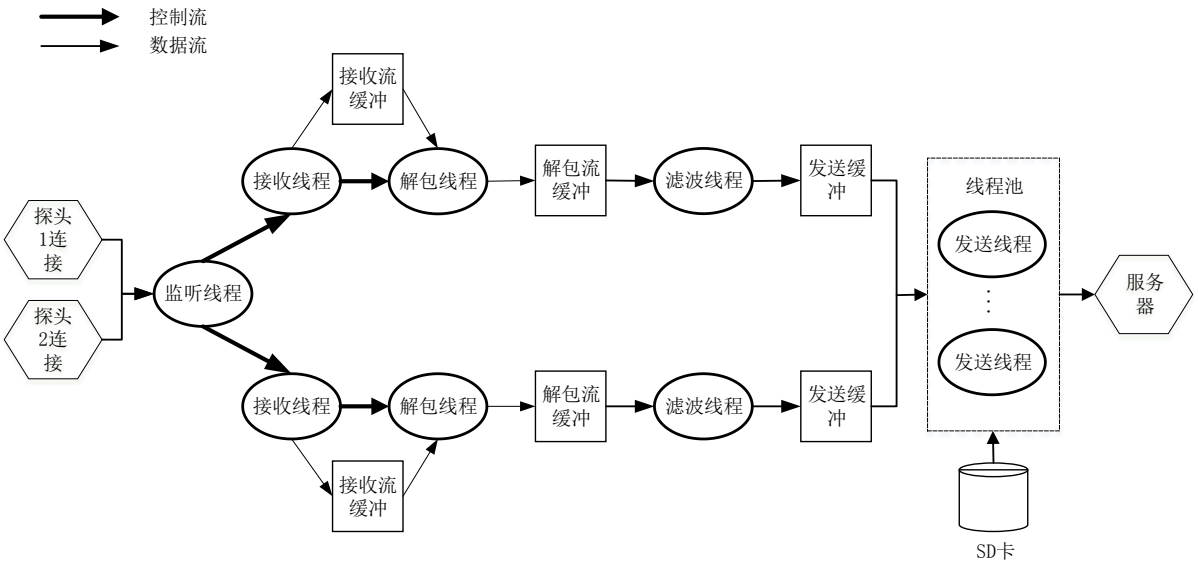

图 3.1 低频连续采集数据处理终端软件的设计方案

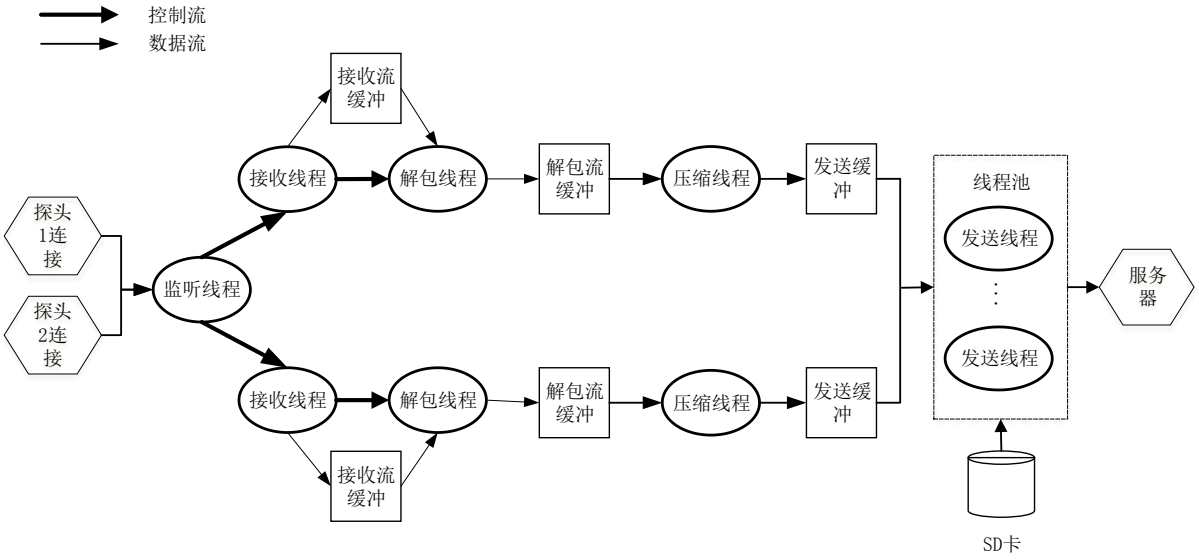

图 3.2　全频连续采集数据处理终端软件的设计方案

的数据量时，滤波线程唤醒发送线程池中的一个线程，将 1 分钟的低频数据发送至服务器。发送线程与 server 应用程序之间基于 HTTP 协议通信，采用线程池技术，可以确保有足够长的时间发送实时数据，避免整个软件数据流链路的阻塞。

3.2.2　连续采集与抽样采集数据处理终端软件设计方案的对比

图 3.3 为抽样采集数据处理终端软件 EqTerminal 的设计方案。主程序为监听线程，若在指定端口监听到有探头请求连接，则为该请求创建配置探头线程、监控线程、定时线程、接收线程、解包线程、滤波线程、发送线程。其中配置探头线程、监控线程、定时线程一旦创建则开始工作，而其它线程则在创建完成后自动挂起，等待其它线程的唤醒。

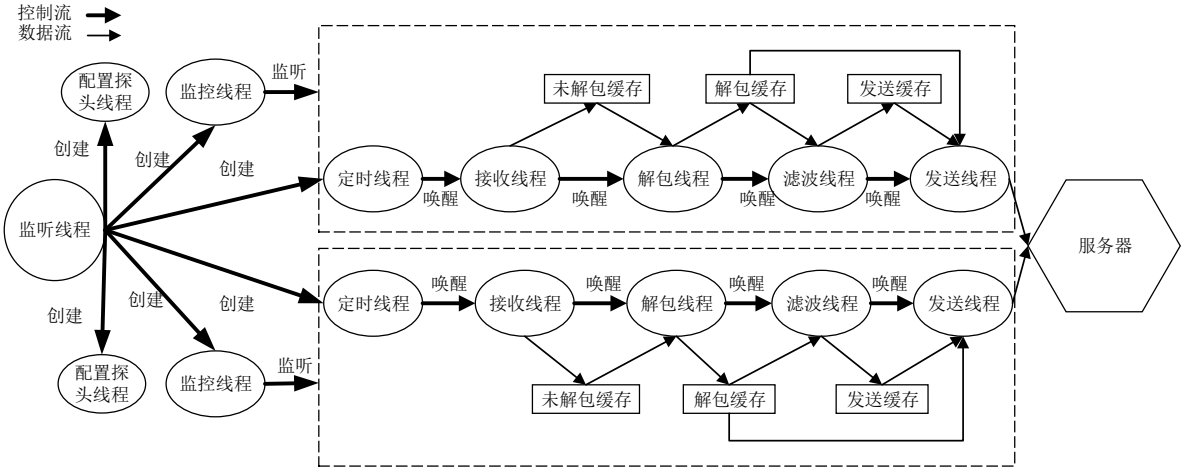

图 3.3　抽样采集数据处理终端软件的设计方案

抽样采集终端软件按照每 3 分钟一个时间窗口，对接收到的探头数据截取 1s 钟的全频

数据和 1 分钟的低频数据。基于这一设计要求，数据处理程序以 3 分钟为一个周期，按照时间片进行划分，错开各个线程之间的工作，避免线程之间的资源竞争。如定时线程在整 3 分钟时唤醒接收线程，进行数据接收。当接收线程接收到的足够的数据量时，再唤醒解包线程进行解包，然后自身挂起；解包线程完成对数据的解包任务后，唤醒滤波线程进行低频滤波，然后将自身挂起；滤波线程完成滤波工作后，唤醒发送线程进行数据的发送，自身再挂起。发送线程完成对全频和低频数据以及其它与 server 应用程序交互的相关任务后，将自身挂起。等待下一个 3 分钟的循环。

在线程结构上，抽样采集与连续采集数据处理终端软件均采用多线程设计，不同的是，连续采集数据处理终端软件为实时流式处理模式，各个线程并行运行，而抽样采集数据处理终端软件的各个线程运行方式则类似于串行工作，例如接收线程完成接收任务后，唤醒下一个解包线程，同时将自身挂起；与抽样采集相比，由于连续采集数据处理终端软件的各个线程同时运行，无需定时线程进行调度，因此取消了定时线程和监控线程；此外，在探头端进行数据的少量处理与计算，在某种程度上会丢失信号的频率和幅值信息。因此，将探头中的数据处理（调零放大）转移到了服务器端的 server 应用程序中，使得终端上传的数据为除滤波或者无损压缩外，没有经过其它任何数据处理的原始数据。由于探头端不需要再接收计算参数，因此在连续采集数据处理终端软中，取消了探头配置线程；此外，在缓存设计上，由于抽样采集数据处理终端软件每次仅有同一类型的线程运行，存入缓存中的数据量可控，因此缓存采用简单数组实现，而对于连续采集数据处理终端软件，各个线程并行运行，实时填充缓存，为防止内存溢出，缓存采用了循环队列的设计。

3.3 连续采集终端软件的设计与实现

3.3.1 数据交互协议的设计

数据处理终端位于 AETA 系统软件的中间环节，既要与探头进行数据交互，又要与 server 应用程序通信。数据处理终端与传感探头之间通过 TCP/IP 协议通信，与 server 应用程序之间通过 HTTP 协议通信。由于数据处理终端和 server 应用程序需要识别数据来源、数据类型、数据产生的时间、数据长度等信息，因此，在软件的设计中，需要设计出一种适用于探头与终端之间、终端与 server 应用程序之间的数据传输协议。

传感探头在与终端建立 TCP 连接之后，直接向终端发送数据。由于探头的数据发送速率较快，因此探头在数据发送出去后，在应用层无需等待终端对数据的确认回复即可继续发送下一条数据。探头除了发送从 ADC 采集的电磁和地声传感数据外，还需定时向终端发送探头自身运行状态，如探头软件版本号、状态发送间隔等，而终端也可

以向探头下发控制命令，如复位等，因此探头还需要接收来自终端的控制信息。探头与终端之间的交互协议如图 3.4 所示。

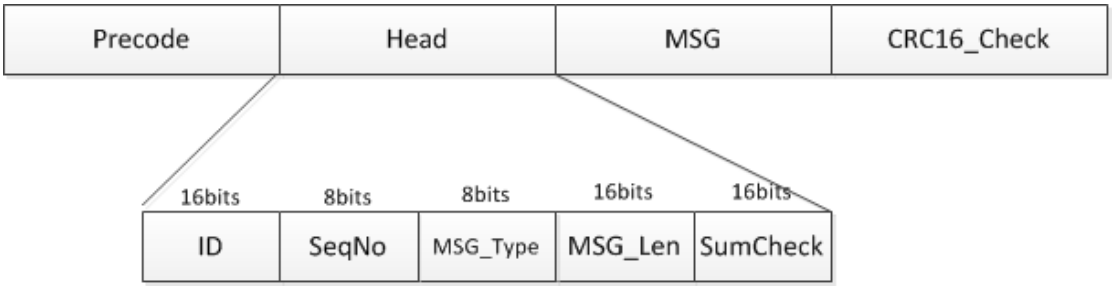

图 3.4　数据终端与传感探头之间的数据协议

下面对各个部分做介绍：

(1) Precode 前导码

前导码位于数据包的最开始位置，字节偏移为 0。先导码尽量要与数据包的其他部分相异，将其暂定为 0xFFFF，为两个字节大小。

(2) Head 头

Head 头中 ID 表示探头的设备编号；SeqNo 为数据包的序列号，每生成一个数据包，则 SeqNo 加 1；MSG_Type 用于区分消息类型， 0x01 表示数据， 0x02 表示命令；MSG_Len 代表 MSG 的长度，即 MSG 中包含有多少个 16bits 的数据；SumCheck 为包头的和校验。

(3) MSG 消息

在连续采集版本中，MSG 消息分为数据和命令两种，格式如图 3.5 和图 3.6 所示：

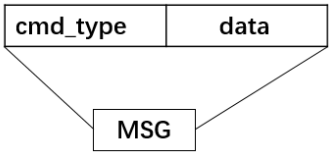

图 3.5　命令的消息格式

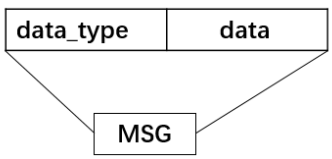

图 3.6　数据的消息格式

传感探头上传到终端的数据包含由电磁探头数据、地声探头数据、温度数据、探头状态和探头日志等不同的数据类型，表 3.2 展示了不同的 data_type 所代表的含义。对于命令形式的 MSG 消息，目前只有终端发往探头的命令消息，Cmd_type 为 0x0002，后面的 data 为 0x0000，表示不让探头复位，0x0001 表示复位，0x0002

表 3.2　data_type 数据含义

data_type	说明	数据说明
0x0001	电磁数据	18 位数据低两位不要，只传高 16 位，不间隔连续传输，一次传输 1024 个点，数据为有符号数据
0x0003	地声数据	18 位数据低两位不要，只传高 16 位，一次传输 1024 个点，数据为有符号数据
0x0007	温度数据	16 位，温度数据 1min 才一个值，一次传输 1 个温度值，换算时除以 100 则可得到温度值。
0x0009	探头状态	ID：16 位；探头类型：8 位，1 为地声，2 为电磁，其他保留；软件版本：24 位,A.B.C；参数同步间隔周期:16bits；零漂补偿值：16bit；放大倍数：16bit。顺序为 ID+类型+软件版本+NULL_复位+参数同步间隔周期+零漂补偿值+放大倍数。
0x000a	探头日志	0x0001　探头重启 0x0004 表示 crc16 校验出错，0x0005 表示和校验出错，0x0006 表示前导码错误

表示让探头停止发送数据，0x0003 表示让探头发送数据。此外，命令形式的 MSG 消息还可以根据需求进行拓展，使得运维人员可以远程通过终端达到控制探头的目的。

数据处理终端与 server 应用程序之间采用 HTTP 协议进行通信，终端向服务器发送 http 请求，以 POST 的方式传输数据和参数。参数列表以 key1=value1&key2=value2 的形式传输。如 terminalId=201801&password=123456。大量的数据如日志和原始数据以文件的形式上传。

终端向服务器上传文件，文件名由时间戳、终端号、数据类型、协议版本号构成，例如：1501659843_25_1_1，其中 1501659843 代表时间戳，代表 25 号终端，数据类型为 1，协议版本号为 1。

因为 http 请求是无状态的短连接，为了能保证终端能够一次登陆多次上传数据，需要引入 http 的 session 机制。http 报文头中需要加一对参数：sessionid=value。每个终端都有一个唯一的 sessionid。每次发送请求需要将 sessionid 放在报文头中传递过来，用于验证终端身份以及是否超时。初始化的终端，sessionid 应为””，第一次登陆成功之后服务器会返回一个 sessionid 值，终端需要将该值保存在本地，之后每次请求都需要将 sessionid 附加在报文头中，如果服务器未检测到 sessionid 将拒绝终端的请求。终端可能会因为断电或断网长时间未向服务器发送数据，导致 sessionid 超时。当终端恢复连接向服务器重新发送请求时，如果服务器判断超时，则需要重新登陆。当非登陆请求

的 session 超时，则服务器会返回状态码为 200 的响应。表 3.3 所示为 session 超时时，服务器的响应内容。

表 3.3 session 超时响应内容

参数名	参数值
sessionError	TimeOut(超时) NotExist(服务器不存在该终端的session) NotSend(请求头中不包含sessionId) Invalid(sessionI无效)

3.3.2 软件的模块划分

根据 3.3.1 节对连续采集终端软件的整体框架分析，将软件划分为图 3.7 所示的九个模块，分别为接收模块、显示模块、解包模块、数据处理（滤波或压缩）模块、发送模块、SD 卡存储管理模块、与硬件交互模块、流缓冲模块、发送缓存模块。由于流缓存模块和发送缓存模块功能类似，因此将其统一归类为缓存模块进行描述。

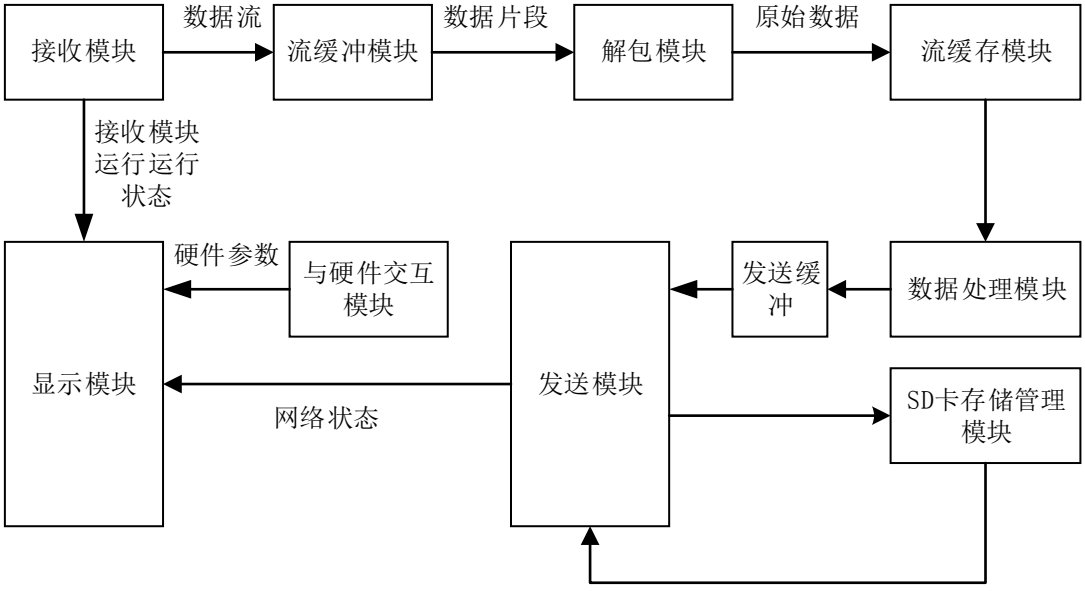

图 3.7 DataTerminal 的模块划分

3.3.3 软件各个模块的设计与实现

1. 接收模块

数据处理终端软件与探头软件之间是基于 CS 模式设计，探头是客户端，终端是服务器。服务器的设计一般有三种模式：多进程设计、多线程设计和 I/O 复用设计。AETA

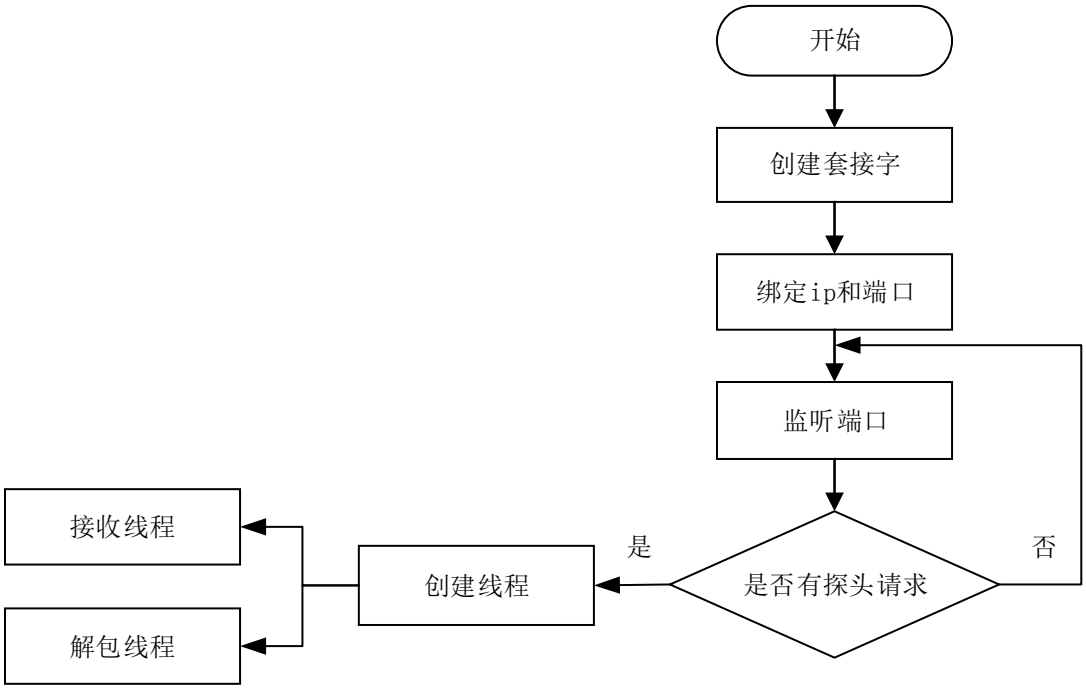

图 3.8　接收模块流程图

系统监测的地震前兆信号分量为电磁信号和地声信号，在硬件设计上为终端预留了两个网络接口，分别用于连接电磁探头和地声探头。对于每一个终端而言，最多只有两个传感探头向终端请求连接，终端不需要处理高并发请求的问题。因此，接收模块采用多线程模式。

图 3.8 为接收模块流程图，主程序在创建并打开套接字后，监听指定端口，一旦有探头请求连接，则创建接收线程和解包线程服务于这一连接，接着主线程继续监听这个端口；若该指定端口没有探头请求连接，则主线程阻塞在监听状态，直到有新的请求连接将这一线程唤醒。

2.　解包模块

传感探头按照 3.3.1 节设计的协议对数据进行打包并发送到终端。解包线程需要从接收流缓冲中读取未解包的数据并按照数据协议对探头数据进行解包。图 3.9 为解包流程示意图。

解包线程在流缓冲中按照字节偏移的方法找到前导码，然后读取完整的数据包头并验证包头是否正确。若包头验证通过，则根据包头中消息的长度来读取完整的原始数据。若包头验证不通过，则直接丢弃这个包的数据，重新寻找前导码解包。在解包的过程中，对包头的验证采用和校验的方式，而对数据的校验则采用 CRC16 校验。

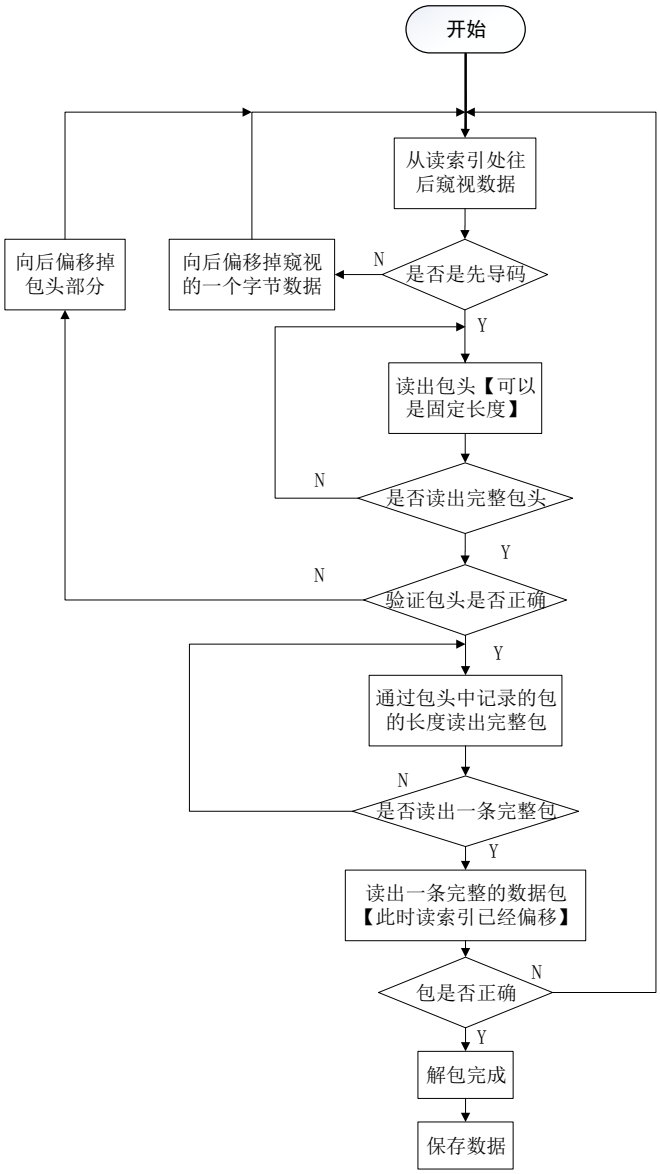

图 3.9　解包模块流程图

3. 数据处理模块

连续采集终端软件分为全频连续采集和低频连续采集两个版本，由 3.2.1 节可知这两个软件版本的唯一区别在于数据处理方式的不同。其中全频连续采集对原始数据进行压缩处理，低频连续采集对数据进行滤波处理。两种处理方式的目的都是为了降低数据量，减少传输带宽和存储资源的成本。对于低频连续采集，数据处理模块即为滤波线程。滤波器为 64 阶滤波器，需要 64 个点才能计算出一个低频数据。滤波线程每次取 1s 钟的数据作为滤波算法的输入，进行滤波，对于因未凑够 64 个点而无法滤波的数据将与下 1s 钟的数据进行拼接，然后再输入滤波算法中进行滤波。这种滤波方法即为平滑连续滤波，可以保证滤波之后得到的数据仍然未连续的。对于全频连续采集，数据处理模块为压缩线程。压缩线程对采集到的 1 分钟的数据进行压缩后放入到发送

缓冲中，由发送线程将压缩后的数据直接发送到服务器。压缩算法的实现将在第四章中详细描述，下面主要介绍滤波器参数的计算。

AETA 系统目前存在高频和低频两个版本的传感探头。以高频版本的探头传感数据为例，数据处理终端需要将接收并解包得到的原始数据通过低通滤波器滤波得到0.1Hz~200Hz 的低频信号，对应的数据采样率为 500Hz。由表 3.1 可知，探头采样率均大于 10kHz，为避免丢失较多信息，设计了一个 64 阶两级滤波算法，采用降采样方法将全频原始数据滤波为 500Hz。两级滤波算法的相关参数如表 3.4 所示。

表 3.4　滤波参数

信号类型	滤波次序	滤波类型	输入信号的采样率（Hz）	通频带（Hz）	输出信号采样率（Hz）
地声	第 1 级	低通	150k	3k	10k
	第 2 级	低通	10k	200	500
电磁辐射	第 1 级	低通	30k	1k	3k
	第 2 级	低通	3k	200	500

由于采用两级滤波算法，每一级滤波均是一个滤波器，因此针对电磁和地声数据总共需要设计四个滤波器。本文采用 Matlab 中的辅助工具 FDAtool 来设计滤波器，并最终得到滤波器的滤波系数。根据 Matlab 计算得到的窗函数系数，在 Linux 系统下，采用 C++语言实现了该滤波算法，并将其封装成了一个类。数据处理终端软件 DataTerminal 每次取 1s 钟的数据作为滤波器的输入，循环滤波。为保证滤波后低频数据的连续性，需要采用平滑滤波的方式，即上一秒钟未参与滤波计算的数据与下 1 秒钟即将进行滤波的数据进行拼接，将拼接后的数据作为滤波器的输入数据。

4. 缓冲模块

(1) 流缓冲区的设计

流缓冲的本质为一个循环队列，由图 3.2 可知，DataTerminal 中使用了接收流缓冲和解包流缓冲。接收流缓冲用于存放接收线程实时接收的数据，解包流缓冲用于存放解包线程从接收流缓冲中实时解包获取的原始数据。根据 DataTerminal 的设计需求，该模块需具备以下两个特性：

a) 支持循环读写，流缓冲的存储空间要大于 2 个最长的探头数据包所占用的空间大小。

b) 对外提供读和写两个接口。

在 DataTerminal 的代码编写过程中，将流缓冲封装成了一个类，如下所示：

```
class StreamBuffer{
public:
    StreamBuffer(int buff_size);
    ~SteamBuffer();
public:
    int WriteData(char* buff, int len);
    int ReadData(char* buff, int len);
    int MoveReadIndex(int n);
private:
    char* m_pBuffer;
    unsigned int m_nBufferLen;
    unsigned int m_nWritePos;
    unsigned int m_nReadPos;
};
```

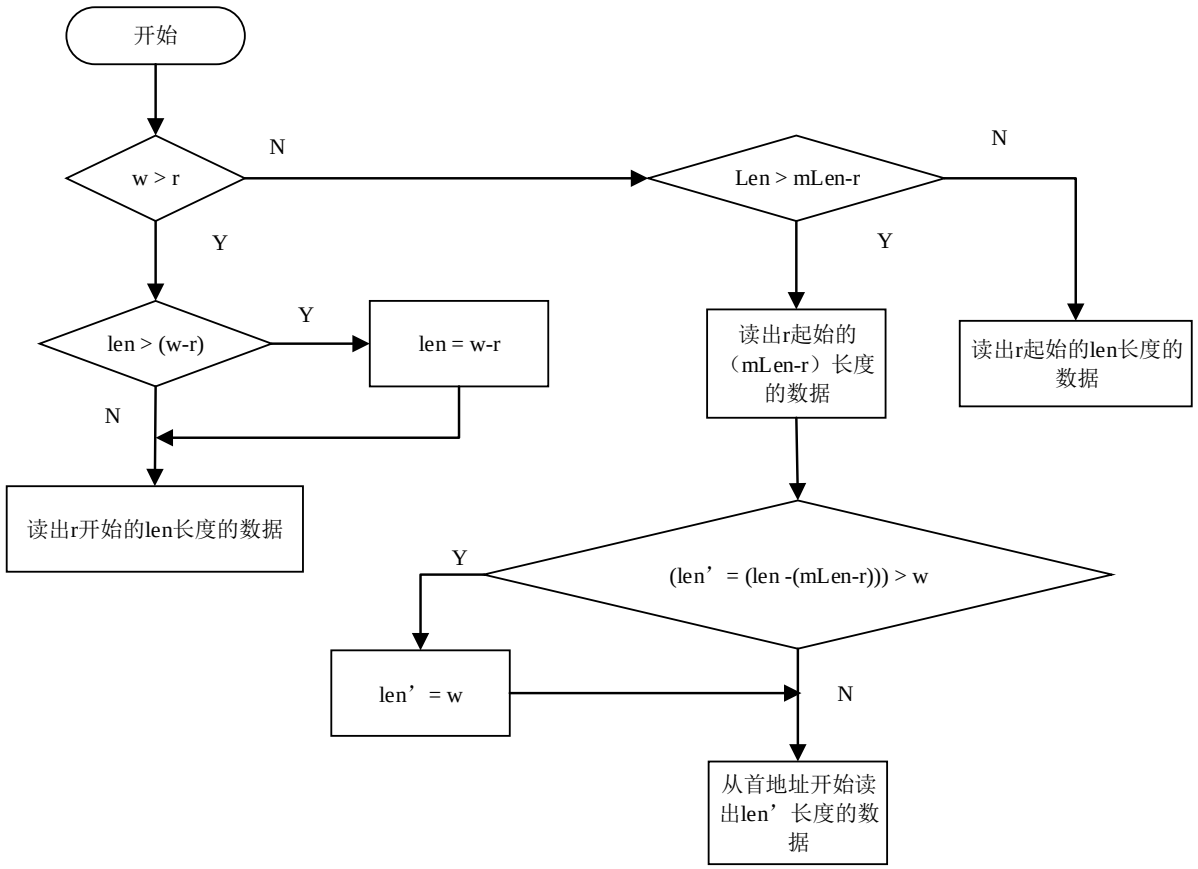

图 3.10　流缓冲区读接口流程图

在上述流缓冲类中，写接口 int WriteData(char* buff, int len)主要负责将数据存储进 m_pBuffer 数组中。当需要存入的数据长度大于 m_pBuffer 剩余空间时，将剩余未存放

进数组的数据从 m_pBuffer 数组的起始地址继续存储，即循环使用数组 m_pBuffer，其原理与循环队列一致。图 3.10 为流缓冲模块的读数据接口处理流程。流程图中 w 为写索引 m_nWritePos 的简写，r 为读索引 m_nReadPos 的简写。len 是读接口函数的第二个形参，表示接口者调用者希望读出的数据长度，mLen 表示数组 m_pBuffer 长度。当读索引靠近数组 m_pBuffer 的末尾且 mLen-m_nReadPos<len 时，读接口需要读两次数据，len'则代表第二次从数组起始地址开始要读取长度。

(2) 发送缓冲模块

发送缓冲模块主要用来存放进行滤波或压缩后的数据，发送线程从发送缓冲中取数据直接发送至 server 应用程序。发送缓冲的数据结构如下所示：

```
typedef struct
{
    bool fullflag;
    int probeID;
    int timestamp;
    int length;
    unsigned short int data_type;
    char send_data[MAX_SENDDATA_LENGTH];
}SEND_BUFFER;
```

其中 fullflag 表示是否存满指定数据量的数据，probeId 表示这个缓冲区的数据来源于哪个探头，timstamp 表示数据的 unix 时间戳，length 表示数据的长度，data_type 表示数据的类型，send_data 用于存放数据。

5. 硬件交互模块

为了能够直观的了解终端硬件的工作状态，需要在终端显示屏上显示终端硬件的工作电压和工作电流。M3352 工控板上自带 ADC，连接到了硬件模块，可将电流和电压转化成数字值。硬件交互模块完成的工作便是从 ADC 中读取电流电压的值，并将其转化为模拟值电压值传送给显示模块，屏幕显示模块将来自于硬件交互模块的电流电压值显示在屏幕上。

6. 发送模块

发送模块主要负责与服务器端 server 应用程序进行数据通信。发送模块完成登陆验证、数据发送、状态发送、命令查询等功能。图 3.11 为发送模块流程图。其中登陆验证为终端和服务器之间的一种身份认证与识别机制，避免其它未知流量对服务器的攻击；数据发送功能即终端将解包并处理好的探头传感数据发送至 server 应用程序，由 server 应用程序进行特征值的提取；状态发送功能指终端定时发送终端软件与探头

数据采集软件的运行状态；命令查询功能即终端支持接收并来自 server 应用程序下发的 Linux 命令，使得运维人员可以远程控制终端。

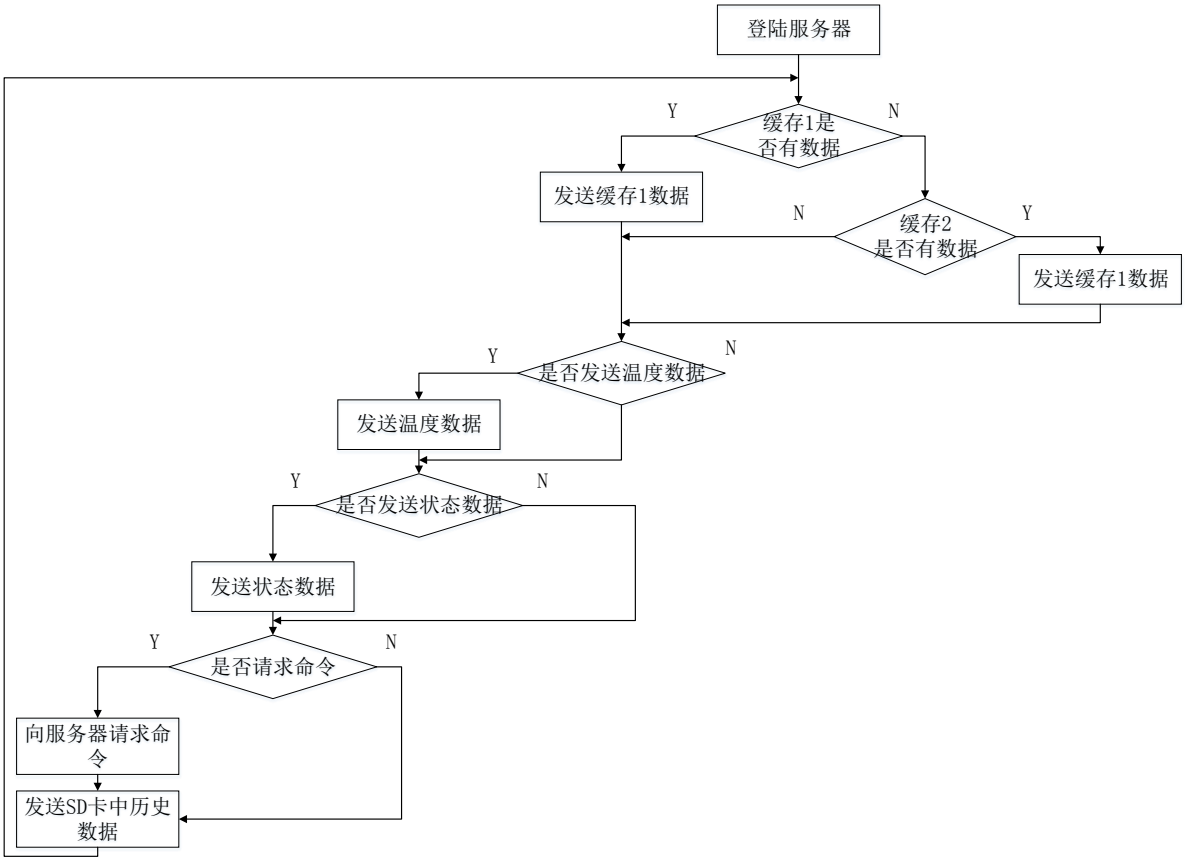

图 3.11　发送模块流程图

7. 屏幕显示模块

该模块负责将终端和探头的必要信息显示在显示屏上，包括探头 ID、终端 ID、网络状态、终端的工作电压和电流等。显示屏模块为一个独立的线程，接收其它线程提供的显示参数，并实时刷新，使得屏幕上显示的数据为最新的。

8. SD 卡存储模块

AETA 系统多数安装在地震频发的地域，如四川、云南等地。由于地理位置等原因，AETA 系统安装站点的网络可能不太稳定，在数据无法通过网络传输到服务器上时，需要将数据存储在终端本地，待网络恢复稳定之后，再读取历史数据重新发送至服务器。由于 M3352 工控板上的 ARM 核心板硬盘只有 256MB，容量较小，无法存储较长时间的数据，因此在终端上外挂了一张 32GB 大小的 SD 卡，可存储一台终端一个月以上的监测数据。为方便 SD 卡内数据的读写操作，设计了 SD 卡存储模块，提供读写两个接口。

3.4 在线升级程序的设计与实现

升级程序完成对文件包括升级程序自身的升级。升级程序只在系统启动时执行，执行完后，升级程序自动结束。

在服务器上，每一个终端都有对应的一个升级文件夹，里面存放有升级信息描述文件 NewRelease.ini 和升级文件压缩包 Update.zip。其中 NewRelease.ini 文件中包含终端 ID 号，NewRelease.ini 的版本号，升级文件压缩包名，脚本文件名，以及服务器端计算的 Update.zip 文件的 CRC 校验值；Update.zip 文件包含要升级的文件清单，升级文件和脚本文件。

升级程序的升级逻辑是:首先从服务器下载升级信息描述文件 NewRelease.ini，判断是否需要进行升级。如果需要升级，则下载升级文件压缩包 Update.zip 到一个临时目录，校验成功后解压缩该文件，然后找到压缩包中的脚本文件，给脚本文件加可执行权限后，执行脚本文件。后续的所有工作比如将文件放到指定的目录下或执行其它操作均由脚本文件完成，脚本文件完成所需工作后，重启终端。升级过程中产生的日志以.txt 的文件格式保存，然后经过 FTP 上传至服务器该终端 ID 所对应的升级目录下。图 3.12 即为升级程序的流程图。

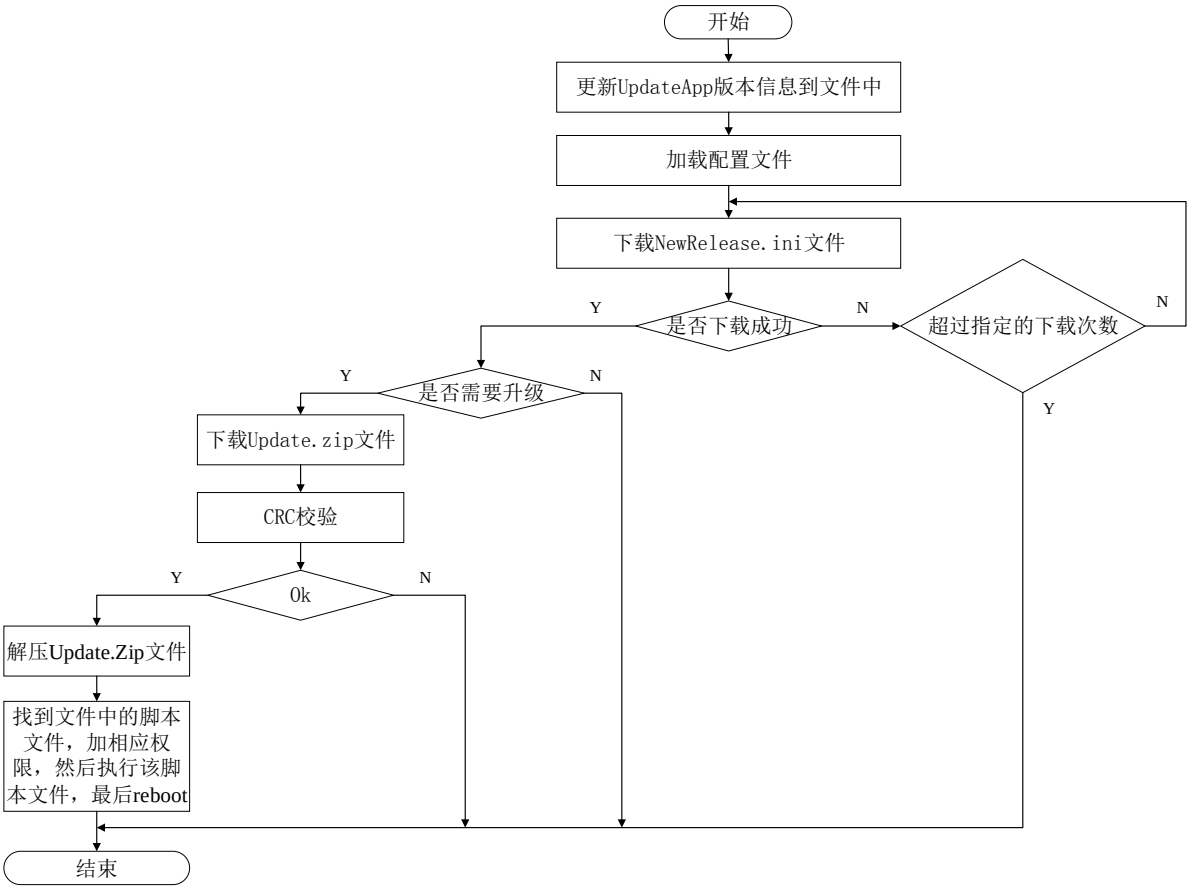

图 3.12 在线升级程序设计流程图

3.5 日志管理程序的设计与实现

日志管理程序 SynApp 主要完成对 Dataterminal 程序的监控以及时间同步和日志处理这三个功能。SynApp 主线程在初始化完成后，创建一个线程，用于完成时间同步和日志处理的功能，由主线程来对 SynApp 进行监控。主线程定时读取共享内存中的心跳信息来判断 Dataterminal 进程的运行状态。图 3.13 为完成时间同步和日志处理功能的线程运行框图。

该线程首先执行登陆，如果登陆失败，则每隔一分钟持续登陆，直到登陆成功。登陆成功后，执行时间同步和发送日志。之后就是一个循环，每隔一小时执行一次时间同步和发送日志。

图 3.14 为时间同步模块，时间同步成功，直接跳出该模块，时间同步失败，可以再次进行时间同步，如果时间同步多次失败，则跳出该模块。首先向服务器端请求时间同步，然后判断时间同步是否成功，如果成功，则跳出时间同步模块，如果失败，则写相

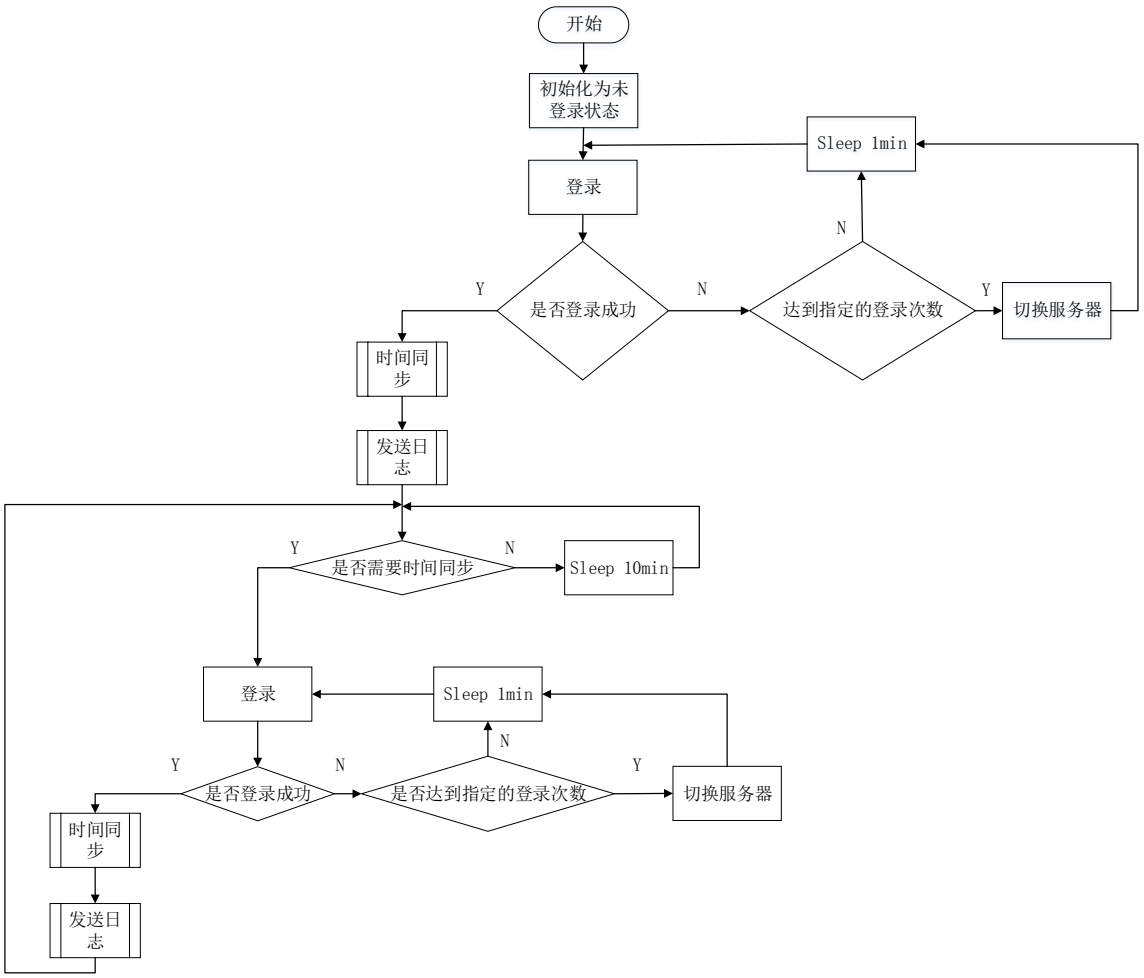

图 3.13 日志管理程序流程图

应日志，然后判断有没有超过时间同步的次数，如果没有则再次同步，如果超出了，则说明在指定同步次数内同步失败，则直接跳出该模块。

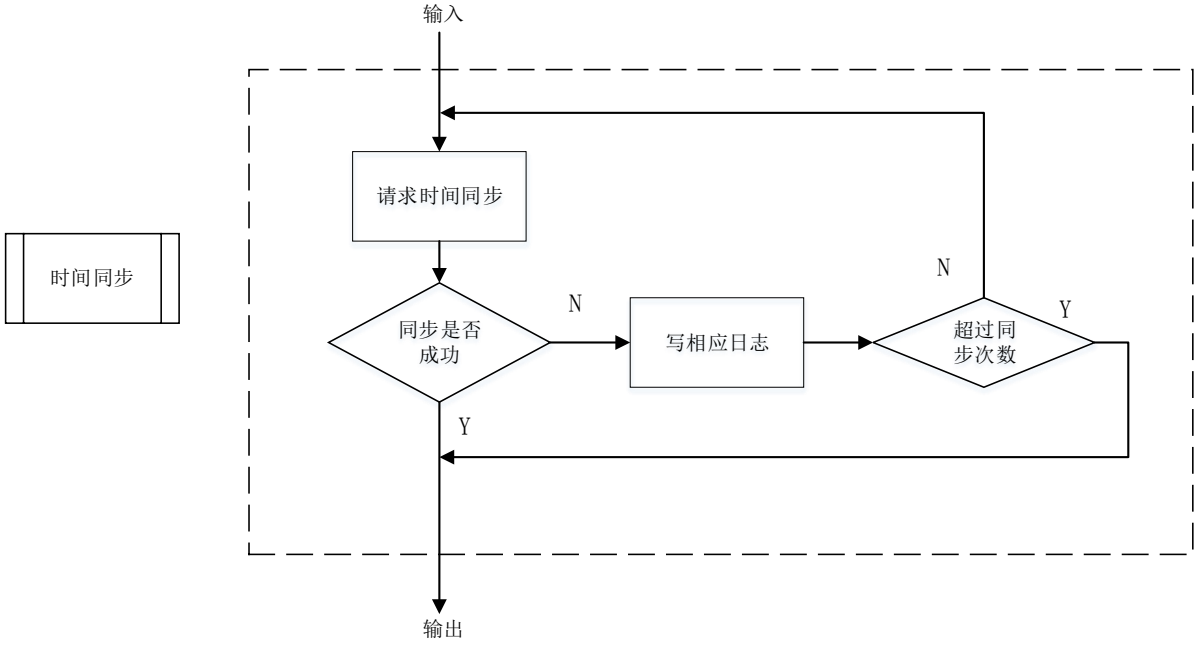

图 3.14　时间同步模块

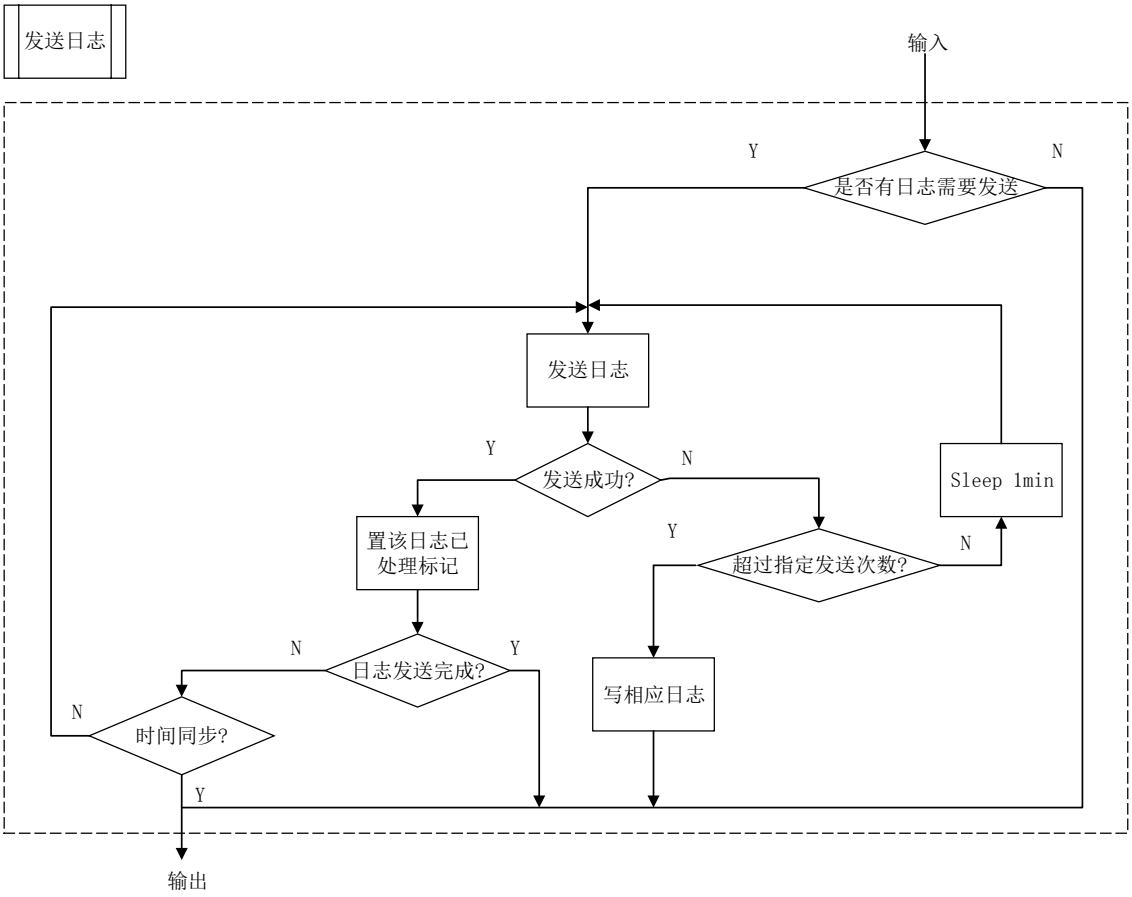

图 3.15　日志发送模块

图 3.15 为发送日志、警报模块。所有的日志、警报均写入 SD 卡内，由于发送日志、警报和发送数据是在不同的进程进行，因此 SD 卡内数据和日志需要分开存储。在发送一次日志失败后，可以再次发送，如果在指定的发送次数内依旧发送失败，则跳出该模块。其次，由于在网络不好的情况下，SD 卡内会存在大量日志，因此，为了防止该程序一直发送日志，影响时间同步，所以在该模块内还需要判断是否到了时间同步的时间，如果需要进行时间同步，则退出发送日志模块。

3.6 数据采集探头软件的设计与实现

数据采集探头中运行有两个程序，分别为 BootLoader 引导程序和 DataProbe 数据采集程序。本小节主要介绍这两个程序的设计与实现。

3.6.1 传感探头引导程序的设计与实现

引导程序本质上是一个通用的 STM32 应用程序，特殊之处是它的功能：上电时引导系统。引导程序是为了实现引导功能，其部分程序是与系统硬件设计细节密切相关的。系统上电或者复位后，首先运行引导程序，并且预留交互接口，以便让用户或者系统管理员，进行系统相关的工作，如升级系统固件。

传感探头引导程序文件命名为 STM32F4xx_TFTP_BootLoader.bin，位于 STM32 内部 FLASH 首地址，因此上电时刻，引导程序率先运行，然后初始化其自身要用到的相关外设，如时钟、串口、W5300 网络芯片。紧接着对地面终端的 Linux TFTP Server 发起固件升级请求，下载版本信息文件，判断是否需要进行固件升级工作。如果需要，则进行固件升级，如果不需要，则执行采集板应用程序。

系统固件升级流程：

(1) 在终端的 tftpd 工作目录下存放两个新的 stm32f4xx_earthquake_version.txt 和 stm32f4xx_earthquake_app.bin 文件，其中 txt 文件为版本信息控制文件，避免探头每次重启时都进行升级。

(2) 在传感探头上电或者复位重启时，BootLoader 作为 tftp client 与终端的 tftp server 进行通信，下载版本信息控制文件 stm32f4xx_earthquake_version.txt，然后跟存放于本地 FLASH 的版本信息控制文件内容进行对比，判断是否需要进行升级。若版本号一致，则退出升级程序，执行 DataProbe 数据采集程序。

(3) 若版本号不一致，则 BootLoader 将对固件进行升级。BootLoader 从终端的 tftp 目录中下载新的程序固件 stm32f4xx_earthquake_app.bin，并将其写入存放 application 固件的 FLASH 地址，然后再更新 FLASH 中存储的版本信息。升级操作完成后，退出

升级程序，执行新的 DataProbe 数据采集程序。

因此，STM32F4xx_TFTP_BootLoader 最主要的功能就是系统固件升级，而且只能升级 stm32f4xx_earthquake_app.bin 固件和 stm32f4xx_earthquake_version.txt 版本信息，而它自身固件升级的功能没有实现，因为引导程序一般成型之后，几乎不会再有变动了。这样设计的好处就是，引导程序功能简化，固件尺寸很小，预留了更多的 FLASH 空间给 application 程序，最重要的，减小其自身固件丢失的风险，除了出厂时的一次性烧录之外，没有其他途径去擦除引导程序固件，除非不可意料的情况。

3.6.2 传感探头数据采集程序的设计与实现

传感探头数据采集程序 DataProbe 主要完成传感数据的采集和发送、软件运行日志及状态的生成与发送、命令的接收与处理三个任务。其中数据采集和命令的接收与处理分别是通过 ADC 中断和 W5300 中断来实现，日志则是在软件运行过程中根据事件生成。此外，探头每隔 255s 向终端发送一次运行状态。整个 DataProbe 程序主函数在数据发送和命令执行两个函数之间循环执行。发送函数和命令执行函数通过标志位来判断是否执行某项操作，而标志位则是在中断函数中进行修改的。下面首先介绍两个中断函数的实现逻辑，然后介绍主函数的实现逻辑。

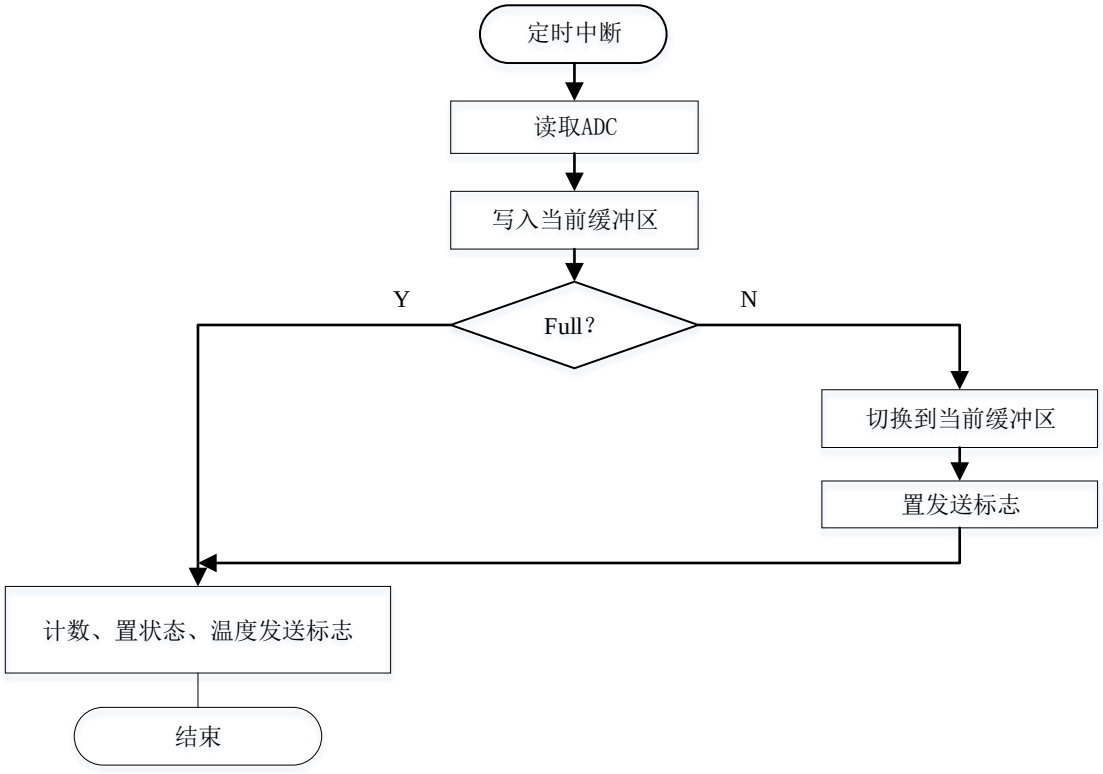

图 3.16　ADC 中断函数实现逻辑

图 3.16 为 ADC 中断函数的实现逻辑.ADC 中断被触发，程序进入 ADC 中断函数

处执行，读取 ADC 数据（2 字节）存入当前缓冲区中。缓冲区采用双缓冲设计，当前缓冲区写满后切换到另一个缓冲区，并置发送标志位。当中断函数执行完毕退出时，程序切换到中断被触发那一刻的状态，继续执行，数据发送函数检测到发送数据标志位后即开始执行发送任务。

图 3.17 为 W5300 中断函数流程图。W5300 中断函数逻辑较为简单，只需将网络标志位置 1 即可。

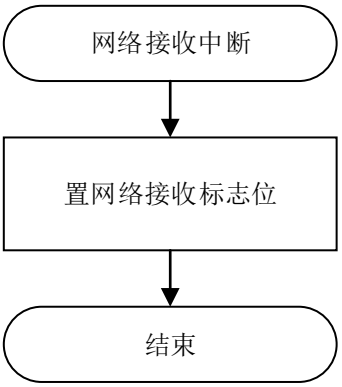

图 3.17　W5300 中断函数流程图

图 3.18 为 DataProbe 程序的 main 函数执行流程图。Main 函数循环执行数据发送和执行命令这两个函数，这两个函数通过判断标志位来决定是否执行相关操作。

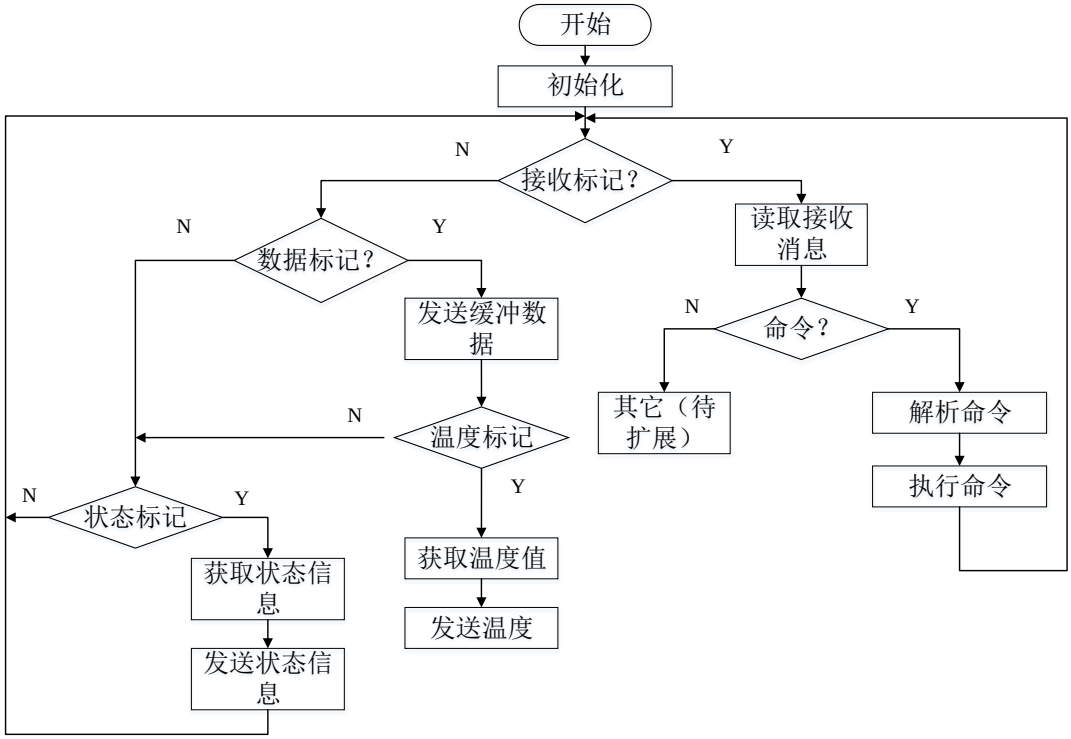

图 3.18　主函数流程图

在数据发送函数中，除了发送传感数据外，还需要发送温度和状态信息；而命令执

行函数中目前只需解析并执行命令，但保留了执行其它任务的可拓展性。

3.7 本章小结

本章首先对 AETA 系统的原始数据数据量进行分析，并对旧版 AETA 系统的数据采集方式进行了介绍，提出在当前项目研究进展下，需要对数据处理终端软件进行优化与重构，将数据采集方式升级为连续采集。针对连续数据采集方式，设计了连续采集数据处理终端软件的软件方案，并与抽样采集数据处理终端软件的方案进行了对比。最后详细描述了连续采集终端软件与数据采集探头软件的设计与实现细节。

第四章 AETA 系统软件的优化

本章将详细阐述 AETA 系统软件在设计与实现中所做的优化，包括 AETA 系统数据的传输与存储优化，数据处理终端软件的线程结构优化，AETA 系统日志处理的优化，AETA 系统软件的稳定性优化与服务器端 server 应用程序的优化。

4.1 AETA 系统数据的传输与存储优化

AETA 系统低频版本的电磁传感探头采样率为 10kHz，地声传感探头采样率为 30kHz。全频连续采集软件对接收到的数据不做任何过滤处理，主要应用在传感探头为低频版本的监测台站内。按照低频版本传感探头的采样率计算，一个标准监测站点（包含地声传探头和电磁传感探头）一个月产生的原始数据量达到 197.7GB,这对数据的传输和存储提出了较高的要求。布设于全国各地的 AETA 系统台站通过有线或无线网络将采集到的数据发送至 AETA 系统后台服务器，后台对数据进行处理与存储。随着 AETA 系统布设规模的不断扩大，AETA 系统的数据量急剧提升，数据的传输带宽和存储成本也不断增大。因此，在连续采集版数据处理终端软件中引入压缩算法，既能够节省数据存储成本，又可以降低 AETA 系统对带宽的要求，同时提升数据传输的稳定性。

数据压缩算法是一种可以减少数据信息冗余的方法，它通过特殊的编码方式将原始数据进行编码压缩，从而降低原始数据文件占用存储空间的大小[72,73]。常用的无损数据压缩算法有 RLE 压缩算法、哈夫曼压缩算法、Rice 压缩算法等，由于 AETA 系统的数据采用率较高，数据在较短的时间内的均方值较小，针对这一特征，设计了一种位重组与 RLE 相结合的地震数据压缩算法。

4.1.1 AETA 系统数据的压缩

AETA 系统数据压缩的算法的核心思想在于数据的位重组。位重组的逻辑可描述为：依次获取待压缩数据中每一个数据的最高位，然后是次高位，如此循环，直到提取完最后一个数据的最低位，将提取到的数据位按照顺序重新组合成新的数据；AETA 系统采集的每一个数据均为 16 位，以 16 位无符号整型数为例，按照从最高位到最低位的顺序，依次提取待压缩数据中每一个数据的第 16 位，第 15 位，…，第 1 位，然后按照提取的顺序，将获取到的数据位重新组合成新的 16 位数字。

位重组的目的是为了提高 0 在二进制数据中出现的次数。对于 AETA 地震监测数据，大部分数据的高位均为 0，通过位重组的方法，可以将每一个数据的高位合并在一

起，组合成新的 16 位数据，而新的 16 位数据中将包含大量的 0 值，可以极大地方便数据的压缩。对于 AETA 地震监测数据而言，由于传感探头采样率较高，相邻两个数据之间的差值非常小。因此，保留数据流中的第一个数据，对其它相邻两个数据作差，可以极大的提高每一个数据二进制形式中 0 的个数，同时也能通过第一个原始数据和差值完全恢复原始数据。然而，对相邻数据作差之后，数据中既包含正数，又包含负数，而负数的高位为符号位 1，正数的符号位为 0，在做了差值的数据中，两者出现的概率相等，很大程度上降低了位变化的效果。因此需要将做差值之后的数据进行数据类型转换，即将负数转换为正数。

数据类型转换可以通过公式（1）所示的奇偶与正负互相映射的方式来实现，即将负数 n 映射为第 n 个奇数 2n-1，正数 m 映射为第 m 个偶数 2m。奇偶与正负的转换公式为：

$$f(x)=\begin{cases}2x, & x\geq 0\\ 2|x|-1, & x<0\end{cases} \qquad (1)$$

其中f(x)表示转换之后的无符号整数，x 表示有符号整数，|x|表示无符号整数 x 的绝对值。

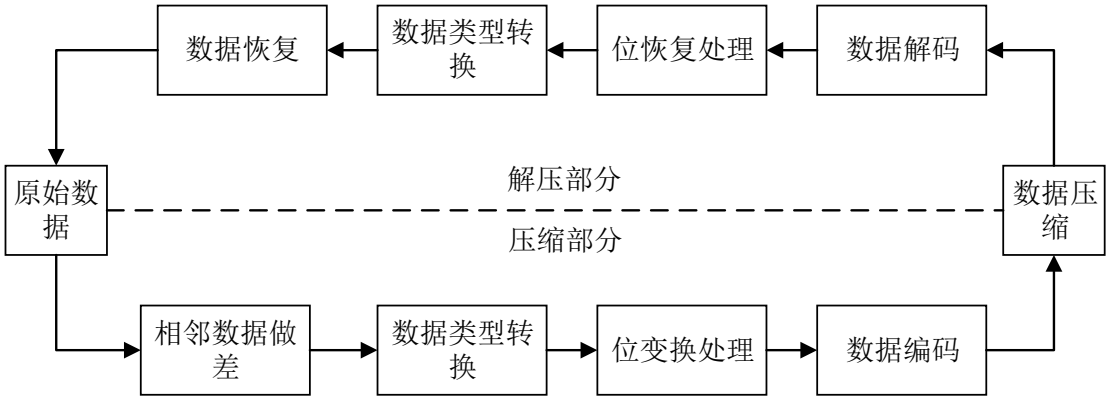

图 4.1　AETA 系统数据压缩/解压缩流程

通过上述介绍，可以得出基于位重组的数据压缩算法实现步骤如图 4.1 所示。数据处理终端每一分钟向服务器发送一次数据，因此，终端软件每次取 1 分钟的原始数据进行压缩。首先对这 1 分钟的原始数据相邻值作差，保留差值和第一个原始数据。然后对转换为差值的数据进行数据类型转换，即按照公式（1）的方法，将所有的有符号整数转换为无符号整数，然后对无符号整数进行位重组处理，得到一组新的数据，对新的数据进行数据编码压缩。数据编码方式借鉴 RLE 编码方式，即采用最简单的计数方式，从头开始遍历位重组过后得到的新数据，统计连续出现多少次 0 值，并用个数和两个标志位来表示这一串的 0 值。对于非 0 值，则保留其原始的数据格式。数据解压

缩是将压缩后的数据还原成未压缩状态的过程。对于压缩过后的数据，首先按照数据编码的规则，找到编码标志位和 0 值长度，恢复 0 值的个数，对于非编码标志位的数据则直接拷贝即可。数据解码之后可以得到经过数据类型转换的数据，再根据奇偶正负的规则将数据恢复成位重组后重新组合的数据，然后对数据进行位恢复即可得到原始数据。采用位重组与 RLE 编码相结合的压缩方式，可以有效地对 AETA 系统数据进行压缩，压缩速度快，压缩效果良好且易于实现。

4.1.2 AETA 系统数据的压缩效果

根据 4.1.1 节设计的 AETA 系统数据压缩算法，采用 C++语言在 Linux 平台上编写了相应的压缩和解压缩程序。本文利用编写好的基于位重组的压缩算法对 AETA 系统的电磁和地声数据进行压缩与解压缩测试，计算了数据的压缩因子，并对比分析了其它几种无损压缩方法的效果。

表 4.1 地声数据压缩效果

台站名称	终端号	探头号	LZW 编码	算术编码	WinRAR 软件	本文方法
泸定气象局	22	30025	2.90	3.16	3.25	3.28
康定姑咱山洞	26	30017	3.12	3.21	3.30	3.35
九龙县乃渠乡	130	30135	3.26	3.39	3.63	3.62
石棉挖角乡	33	30023	3.02	3.12	3.24	3.27
石棉山洞	35	30022	1.63	1.96	2.36	2.39
宝兴县灵关中学	93	30079	3.12	3.22	3.31	3.38
名山区安吉村	124	30141	3.23	3.36	3.60	3.59
西昌气象局	32	30019	2.92	3.13	3.26	3.29
冕宁防震减灾局	38	30021	2.95	3.23	3.27	3.31
德昌防震减灾局	41	30133	3.11	3.31	3.44	3.47
盐源县盐塘乡	78	30138	3.21	3.35	3.49	3.50
雷波县地震台	84	30123	3.29	3.41	3.54	3.66
木里县防震减灾局	101	30139	1.1	1.57	1.73	1.81
西昌小庙山洞	132	30020	3.06	3.26	3.29	3.31
都江堰中学	19	30027	2.16	2.52	2.86	2.94
筠连县气象局	47	30055	3.31	3.47	3.65	3.71

表 4.2　电磁数据的压缩效果

台站名称	终端号	探头号	LZW 编码	算术编码	WinRAR 软件	本文方法
泸定气象局	22	24	1.13	1.30	1.34	1.36
康定姑咱山洞	26	56	1.27	1.65	1.72	1.78
九龙县乃渠乡	130	112	1.31	1.43	1.53	1.61
石棉挖角乡	33	31	1.25	1.20	1.23	1.24
石棉山洞	35	30	1.21	1.26	1.27	1.34
宝兴县灵关中学	93	106	1.13	1.40	1.42	1.45
名山区安吉村	124	133	1.02	1.11	1.15	1.23
西昌气象局	32	19	1.12	1.29	1.33	1.45
冕宁防震减灾局	38	22	1.07	1.10	1.15	1.21
德昌防震减灾局	41	99	2.9	3.42	4.06	3.59
盐源县盐塘乡	78	123	1.30	1.52	1.63	1.68
雷波县地震台	84	118	1.33	1.86	1.96	2.03
木里县防震减灾局	101	125	1.12	1.20	1.22	1.30
西昌小庙山洞	132	18	1.23	1.83	1.85	1.86
都江堰中学	19	20	1.19	1.33	1.45	1.48
筠连县气象局	47	77	1.23	1.45	1.50	1.62

本文的测试数据是从 AETA 系统实际布设的监测台站中获取的同一时间的数据，总共获取了 16 个台站的电磁和地声数据进行数据压缩测试，电磁和地声数据的压缩效果如表 4.1 和表 4.2 所示。从表 4.1 和 4.2 可得，对于均方值较小的 AETA 系统地震监测数据，无论是电磁还是地声，采用本文设计的压缩算法对数据的压缩效果整体上要好于其它几种压缩方法的压缩效果。

此外，由表 4.1 和表 4.2 可计算，电磁数据的平均压缩倍数为 1.67，压缩率为 0.63; 地声数据的平均压缩倍数为 3.34，平均压缩率为 0.3。按照这个压缩效果来计算，一个标准监测站点每个月得数据量可以从 197.7GB 降低为 75.5GB，节省了 122.2GB 的传输流量和数据存储空间，极大地优化了 AETA 系统数据的传输与存储。此外，由于传输数据量的减小，使得部分网络条件不太稳定的台站也能够正常地传输数据，降低了 AETA 系统对优质网络环境的依赖。

4.2 AETA 系统数据处理终端软件的线程结构优化

4.2.1 基于生产者消费者模型的线程结构优化

生产者消费者模型为面向过程编程中的一种常用设计方法。图 4.2 为 AETA 系统连续采集终端软件中的生产者消费者结构，接收线程作为生产者，实时接收传感探头中的数据并将数据放入接收流缓冲；解包线程既作为消费者从接收流缓冲中取出原始数据进行解包，也作为生产者将解包后得到的裸数据放入解包流缓冲中；同理，滤波线程也作为消费者从解包流缓冲中取数据进行连续滤波，也作为生产者将滤波后得到的低频原始数据存入发送缓冲中。

在未采用生产者消费者模型对线程结构进行优化之前，解包线程需要不断轮询接收流缓冲中是否有数据待解包。同样，滤波线程也需要不断轮询解包流缓冲中是否存满 1s 钟的全频数据。由于解包线程解包速度要远大于接收线程接收数据的速度，因次解包线程和滤波线程在循环查询流缓冲是否准备好了待处理数据的过程中，大多数时间处于无效的运行状态，浪费了 CPU 计算资源。

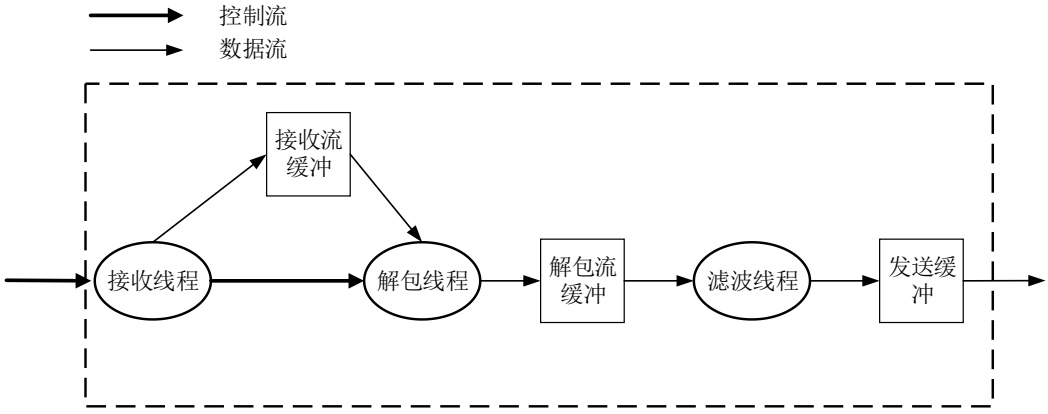

图 4.2 AETA 系统连续采集终端软件的中的生产者消费者结构

优化为生产者消费者模型后，接收线程和解包线程通过条件变量和互斥锁进行互斥与同步，解包线程一旦发现接收流缓冲中没有足够的数据解包，则自身会挂起，等待条件变量的变化；接收线程每次接收到数据往接收流缓冲存放之后，通过条件变量唤醒解包线程进行解包。同样，解包线程在解完一个包后，将数据存入解包流缓冲，待解包流缓冲中存够 1s 钟的数据量后，通过条件变量唤醒滤波线程对数据进行滤波。通过对代码进行生产者消费者模型的优化后，很大程度上降低了 DataTerminal 进程对嵌入式系统 CPU 的占用率。

4.2.2 基于线程池的发送模块优化

在连续采集数据处理终端软件 DataTerminal 中，发送模块完成向服务器发送实时

数据、终端和探头状态、请求命令等功能。数据处理终端与 server 应用程序之间采用 HTTP 协议通信，终端软件 DataTerminal 向 server 应用发送的每一个请求，均有设置一个超时时间，在规定的超时时间内，server 应用程序没有返回相应的响应，则代表这个请求失败。

在未采用线程池来设计发送模块时，发送模块只有一个独立的线程。根据设计需求，数据处理终端每一分钟需要向服务器发送一条数据，因此，发送线程需要在 1 分钟的时间内将低频地声数据、低频电磁数据、终端状态和探头状态全部发往服务器。在单线程处理多个发送任务的情况下，如果设置发送超时时间过长，将会导致整个软件数据处理的阻塞，即下一分钟的数据无法及时得到处理，长此积累，将导致数据丢失。服务器端对终端的请求处理方式是，先将请求放入请求队列中，然后由相应线程从请求队列中取出请求进行处理。若数据处理终端设置的超时时间小于服务器端请求队列的超时时间，在服务端接的请求数量超过一定阈值时，将导致对部分请求的处理过慢，但请求尚未超时，而终端则认为该请求处理超时，将会重新发送该请求，最终会造成雪崩效应，导致服务端累积大量重复无效的请求，造成系统资源耗尽，服务不可用。

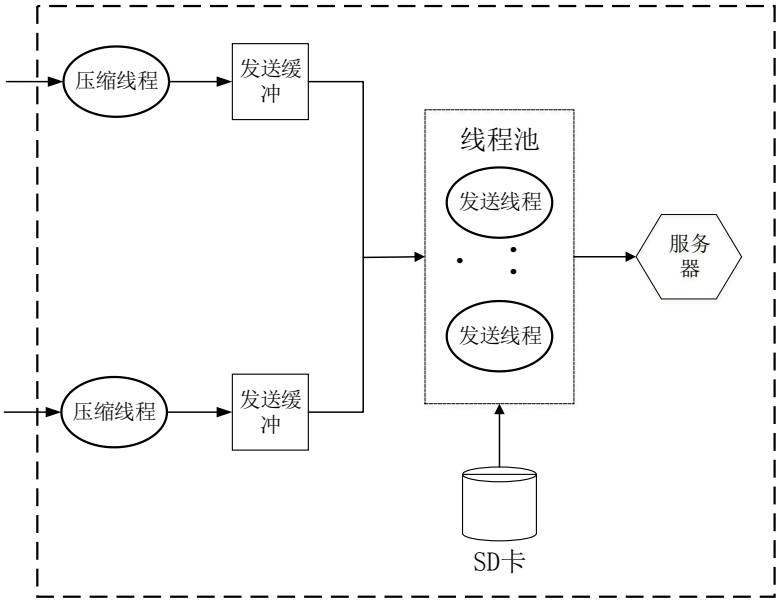

图 4.3　AETA 系统连续采集终端软件发送线程池

采用线程池技术对发送模块进行优化，可以将每一个 HTTP 请求的超时时间设置为一个较长的合理值。在上一分钟的数据还未发送完，下一分钟的数据就已经到来的情况下，可以从线程池中唤醒一个空闲线程来处理新的数据，而上一分钟的数据则继续发送，两者互不影响。

图 4.3 为 AETA 系统连续采集终端软件的发送线程池。在系统初试化时创建好一定数量的发送线程，初始状态为空闲状态。当压缩或滤波线程填满发送缓冲区时，创建

一个新的发送任务加入到任务队列中，从线程池中取一个空闲的发送线程来处理该任务。此时该线程转化为忙碌线程，任务处理完成之后，该忙碌线程又自动挂起，处于空闲状态。线程的唤醒采用条件变量和互斥锁来实现。

经过测试，单线程情况下的发送线程，HTTP 请求的超时时间不能超过 10s，即要求终端软件在 10s 内完成一个发送请求。然而，1 分钟的全频连续采集地声数据压缩后（按平均压缩因子计算）大小约为 1.05MB，对于部分网络环境较差的 AETA 站点，在 10s 内完成 1 分钟的地声数据的发送任务较为困难。采用线程池技术来设计发送模块，对线程池设置合理的发送线程数量，既可以减小线程创建与销毁带来的开销，又能够延长 HTTP 请求的超时时间，提高数据传输的稳定性。

4.3 AETA 系统日志处理的优化

日志可以记录下系统运行过程中所产生的关键信息，并且按照某种规范表达出来，有助于运维人员和开发人员了解系统运行状态，快速定位系统问题[74-79]。

AETA 系统软件自上线以来就设计了日志功能，然而，在连续采集软件之前，AETA 系统软件对日志的处理较为简单，将产生的单条日志不做任何处理就发送至服务器。当 AETA 系统规模较小时，这种日志处理方式能够简单快捷地收集日志数据。然而，AETA 系统已在全国各个地震高发区域大规模布设，经过统计和分析发现，目前单台服务器每天需要处理 100 万条以上的日志，而大量的日志信息和数据并发量过多地挤占了服务器的处理能力和处理时间。因此，为了改变这种现状，设计一套日志系统，规范化日志的描述、采集和存储，并且对收集到的日志进行统计分析，提高服务器的资源利用率，降低系统维护人员的运维成本。

AETA 日志处理系统是基于 AETA 系统软件设计的，由日志数据采集层、日志数据汇集层、持久化存储层和日志数据展示层组成，如图 4.4 所示。日志数据采集层负责对设备的日志信息和状态信息进行采集，并将数据规范化处理，然后按照约定的应用层传输协议将数据传输至日志数据汇聚层。日志数据采集层对日志进行了合并与分级。日志合并的目的是为了降低由日志数据对服务器带来的高并发量；由于日志合并需要等待多条日志才能进行合并处理，降低了日志的实时性，因此对日志数据进行分级处理，保证部分关键的日志能够实时发送至服务器。

AETA 系统日志汇聚层主要分为两个部分：任务管理部分和数据分拣中心。任务管理部分包括任务分拣中心、各类任务调度中心和任务管理中心，负责管理采集层的设备（终端和探头），检测设备的运行状态，实现设备任务分解、协同等调度工作；数据分拣中心负责接收采集层上传的电磁数据、地声数据、日志数据和状态数据，将数据分

类处理，然后将处理好的数据送入入库队列，由入库队列统一入库；

AETA 系统日志持久化存储层采用 Mysql 数据库和 Linux 文件系统存储数据，其中日志数据全部存储在 Mysql 数据库中。持久化存储层通过对数据库中的数据进行分析生成事件，告警，视图等元素的基本信息。展示层通过 B/S 方式展现用户界面，通过数据预取技术将用户可能关心的数据预先缓存到用户本地，提高展示的效率，方便用户的使用。此外，还可为用户提供配置任务、优先级别、阈值和域名的接口，为用户提供个性化服务。

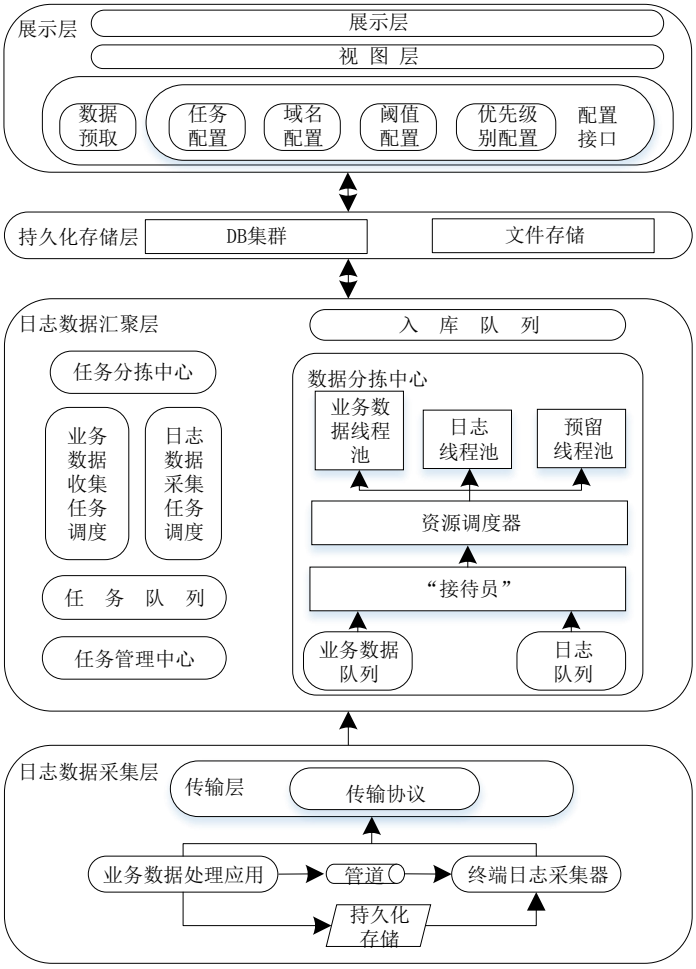

图 4.4　AETA 系统日志处理框架图

通过对 AETA 系统日志的规范化处理，AETA 系统能够高效地处理来约 230 个台站的日志数据，运维人员定位故障平均 1 小时，极大地提高 AETA 系统的运维效率。

4.4　AETA 系统软件的稳定性优化

软件的可靠性是指软件在规定的时间和条件下完成指定的功能的能力[80]；AETA 系统的监测设备遍布于全国各地，部分区域经常面临电力、网络的中断以及终端设备卡

死等情况，为提高 AETA 系统软件运行的稳定性，对安装于各地的终端软件设计监控机制，同时针对网络环境下数据的传输与存储进行了优化。

4.4.1 数据处理终端的自我监控机制

在一个软件系统中，软件或者硬件的微小波动均会引起系统的运行故障，在遇到故障时，系统能否自动恢复运行是衡量一个系统容错性的重要指标。因此，为 AETA 系统软件设计了监控机制。图 4.5 为数据处理终端的监控机制。

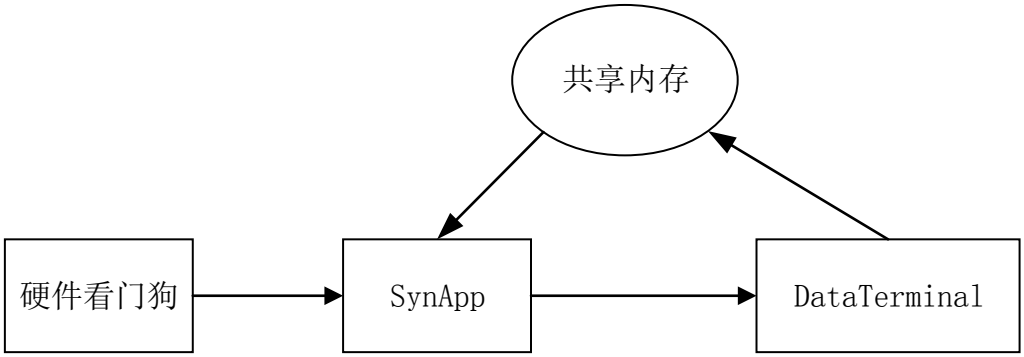

图 4.5 数据处理终端的监控机制

数据处理终端中包含有 UpdateApp、SynApp 和 Dataterminal 三个应用程序，其中 UpdateApp 仅在终端系统启动时运行，运行完进程自动结束，因此不需监控。SynApp 在运行后，启动 Daterminal 进程，DataTerminal 进程定时向共享内存中写入系统当前时间作为 DataTerminal 进程的心跳信息。SynApp 周期性地从共享内存中读取 DataTerminal 的心跳信息，验证 DataTerminal 进程是否正常运行。如果 SynApp 前后两次从共享内存中读取到的心跳信息均不变，则说明 DataTerminal 进程卡死或者奔溃，这时 SynApp 将释放 DataTerminal 进程，并重新启动该进程。

SynApp 由硬件看门狗来进行监控。如果 SynApp 卡死或者奔溃，则硬件看门狗会对整个系统进行复位。硬件看门狗的本质是利用一个连接到系统复位端的定时器电路来实现的，定时器电路每隔一定时间产生溢出信号，对系统进行复位。如果应用程序每个一定时间对定时电路清零，即俗称“喂狗”操作，定时器达不到溢出的状态，因此，无法对系统进行复位。一旦应用程序卡死或者奔溃，停止“喂狗”操作，定时器电路将达到溢出状态，将系统进行复位。

AETA 系统软件中，传感器探头软件也是由硬件看门狗来对程序进行监控。而 server 应用程序运行在阿里云服务器上，对系统进行重启将会影响其它服务的运行。因此，对 server 应用程序的监控采用脚本监控的方式。监控脚本每隔一定时间查询服务器进程表中是否存在 server 应用程序，一旦发现 server 应用程序进程 ID 不存在，则通过短信和电话等方式通知运维人员进行处理。

4.4.2 弱网络环境的处理

AETA 系统监测站点大多部署在地震频发的地域，这些区域中很多地方由于地理位置原因导致网络条件不稳定，经常出现断网或者网速较慢等情况。因此，AETA 系统软件在设计的时候需要考虑在弱网条件下数据的传输问题。本章 4.1 节中描述的 AETA 系统地震数据压缩可以很大程度上减小数据传输对网络质量的要求，提高弱网络环境下数据传输的稳定性。然而，在断网情况下，数据的传输仍然是个问题。数据处理终端内预留了 SD 卡接口，AETA 项目的每一台终端均配备了一张 32GB 容量的 SD 卡，可以存储一个标准台站（低频连续采集）一个月以上的数据。因此，在弱网络环境下，数据无法正常传输到 server 应用程序的情况下，数据处理终端软件可以将数据存储在 SD 卡中，并在网络稳定的条件下取出 SD 卡内的数据发送至服务器。

在 AETA 系统项目启动时，数据的安全性是最重要的，因此在旧版的数据处理终端软件中，SD 卡模块仅有存储数据和数据补传的功能，不含数据删除功能。因此补传完的数据不能自动删除，SD 卡容量将逐渐减小直至用完。当 SD 卡容量用尽之后，只能将 SD 卡寄回，将数据备份后清除，然后再寄回。随着软件运行的稳定性逐渐提高，数据存储与数据补传功能已能够较稳定地运行。因此，在新版连续采集数据处理终端软件中，对 SD 卡模块进行了优化。

在 SD 卡模块中设计了数据存储，数据读取，数据删除，SD 卡容量查询等功能，同时也重新设计了 SD 卡的目录结构，使得 SD 卡目录中的数据结构清晰易读。图 4.6 为 SD 卡持久化存储的目录结构设计，按天创建数据文件夹，文件夹内文件名按序列号从 001 开始命名，每小时加 1，每天最多 24，每个小时内的数据写入同一个文件内。对于每一个日志文件都有相对应的 index 文件，详细记录了每条数据的地址、长度和读取标志，用来描述数据文件中数据的读取位置，是否已读等信息。

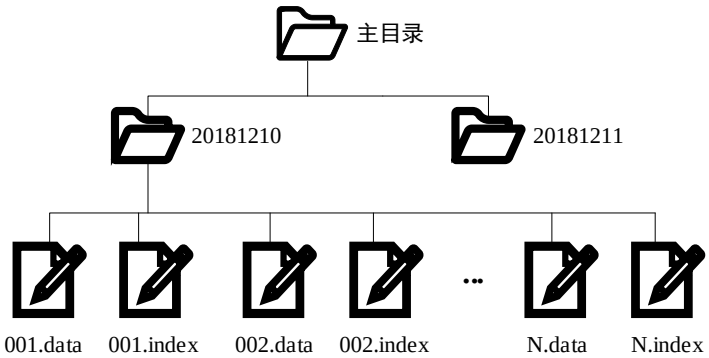

图 4.6 SD 卡持久化存储目录结构

通过对 SD 卡模块的重新设计，在网络中断的情况下，终端软件调用 SD 卡模块的存储接口，将数据存入 SD 卡中；网络恢复后，调用 SD 卡模块的读接口，将数据从 SD

卡中读出发送至服务器，然后清除已发送的数据，可以避免人工进行数据的删除操作。

4.5 AETA 系统服务器应用程序的优化

AETA 系统服务器 Server 应用程序部署在阿里云服务器主机上，负责接收所有数据处理终端发往服务器的数据，并对接收到的数据进行特征值提取和数据入库处理。AETA 系统台站布设规模越来越大，目前台站数据量已接近 230，server 应用程序面临的高并发压力也越来越大。Server 应用程序优化之前，软件针对每一个接收到的 HTTP 数据请求均创建一个线程来处理，线程处理完之后自动释放。在系统并发数量达到一定阈值之后，server 应该程序创建的线程数据达到服务器的上限，会将系统资源耗尽，造成服务器宕机。因此有必要对 Server 应用程序进行优化。

优化后的服务器系统的功能分为两部分：运维监控功能和数据接收服务功能。

运维监控功能主要是记录服务器磁盘余量、内存使用率等信息，和配置数据接收服务可处理的业务并发量阈值和业务的优先级别；数据接收服务会根据服务器的资源使用情况和配置好的线程建立规则（名词解释如表 4.3 所示）决定如何响应请求。

表 4.3 名词解释

名词	解释
资源使用情况	服务器磁盘余量、内存使用率等
线程建立规则	业务优先级别、业务并发量阈值

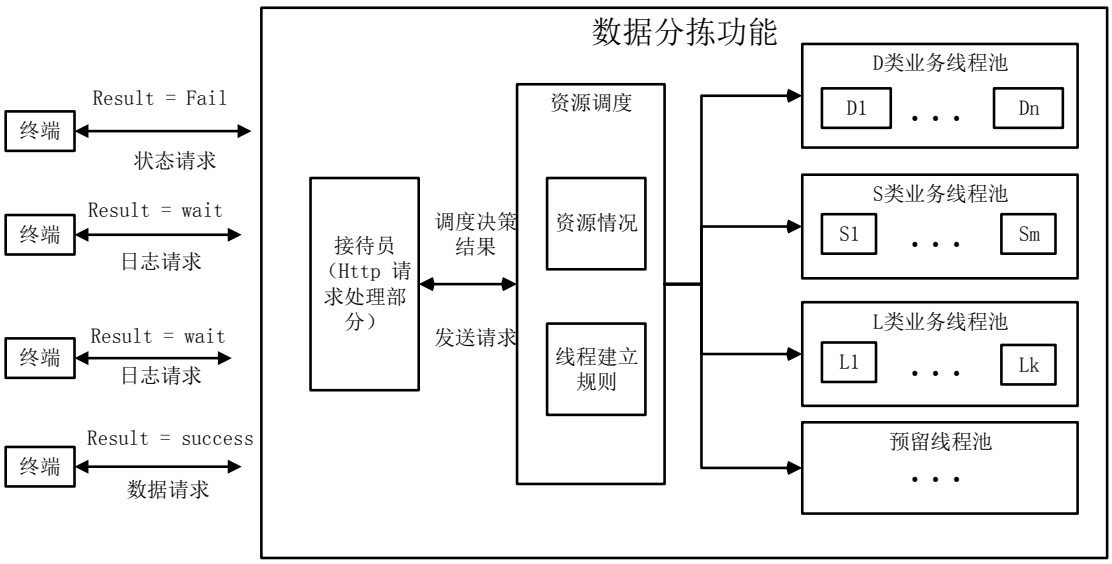

图 4.7 数据分拣功能逻辑架构图

图 4.7 为数据分拣功能逻辑框架图，对数据处理终端上传的数据请求进行分类并交由对应的线程池进行处理。下面对数据分拣功能进行介绍。

(1) 数据分拣功能的业务类别

数据分拣功能的业务类别如表 4.4 所示：

表 4.4 业务类别描述表

类别	业务描述	接收频率	简称
电磁地声数据	低频电磁、低频地声	1 分钟	D
终端状态数据	——	3 分钟	S
终端日志数据	——	1 小时	L

(2) “接待员”

该模块用于将各种请求分类和判断请求是否符合约定。

a) 将请求归类为某一种业务类别：如数据请求中含有 logfile 的请求被归为终端日志数据处理与入库的业务类别；数据请求中包含 datapost 的请求被归为电磁地声数据处理与入库的业务类别。

b) 判断此请求格式和参数是否符合约定，对不符合约定的请求直接返回“result=fail”。

(3) 线程池

拟采用四个线程池来完成三种业务的处理，预留线程池是在各类业务线程池已经超过阈值的情况下，处理优先级别高的请求。

(4) 资源调度

如图 4.8 所示，当“接待员”将分类后的请求发送个资源调度器之后，以电磁地声类业务为例：

a) 资源调度器会根据电磁地声类线程总数 x1、电磁地声类线程池线程阈值 n1 判断是否在电磁地声类线程池中建立线程处理此请求中的数据。如果 x1<n1，则在电磁地声类线程池中建立线程处理数据。

b) 如果 x1>=n1，则继续判断此请求的优先级别 a1 是否可以进去预留线程池，如果 a1 级别较低不能进去预留线程池，则为终端返回“result=wait”。

c) 如果预留线程池中的线程总数 x4 小于预留线程池线程数阈值 n4，则在预留线程池中建立线程处理数据。

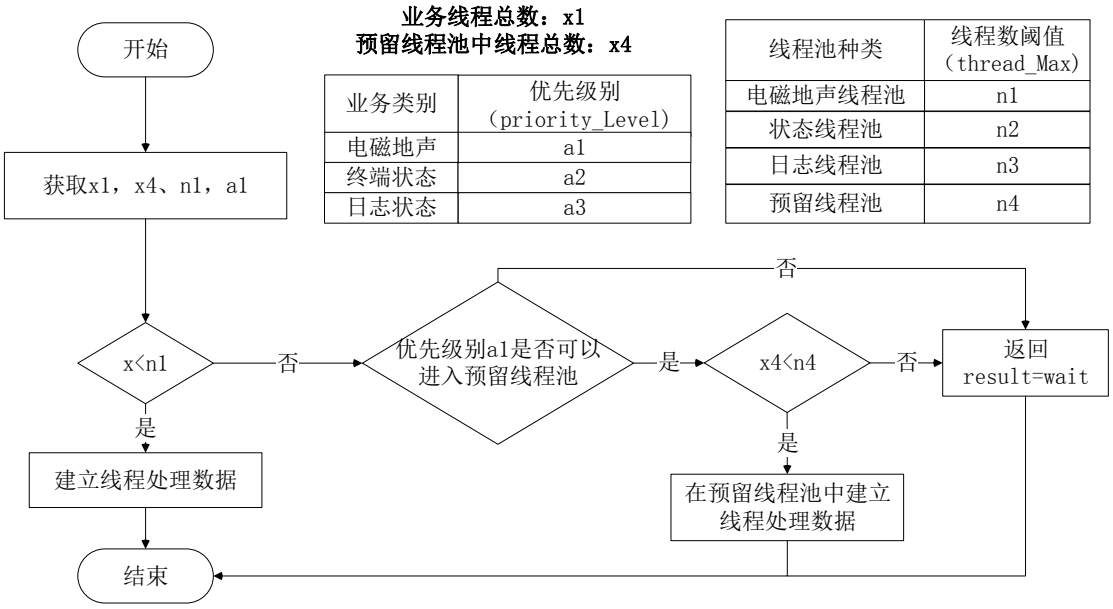

图 4.8　资源调度流程图

d)　如果 x4>=n4，则表示资源与不足以支持处理新请求，则为此终端返回“result=wait”。

(5)　运维监控

为了使数据接收服务能够正常运行， 定期采集服务器系统盘磁盘余量和内存使用情况等信息并记录这些信息，达到监控数据接收服务运行环境的目的，并且数据接收服务也可以根据服务器的资源使用情况来优化资源配置。

a)　定期获取磁盘余量内存使用情况等资源信息，并记录资源使用信息。

b)　设置磁盘余量和内存使用情况阈值，资源使用超过阈值为运维人员预警。

c)　运维人员配置，各个业务线程阈值和各类业务的优先级别。

4.6　本章小结

本章主要介绍了 AETA 系统软件在设计与实现上所做的优化。采用位重组压缩算法对数据进行压缩，可以节省数据的传输和存储成本；基于生产者消费者模型对接收模块、解包模块、滤波模块进行设计，可以提高 CPU 的利用率；基于线程池技术对发送模块进行优化，可以增大 HTTP 数据传输的超时时间，提高数据传输的稳定性，同时降低 server 应用程序的压力；为应对终端的高并发请求，对 server 应用程序进行了优化，通过调度模块将不同的业务请求调度到不同的线程池中进行处理。此外，为提高 AETA 系统软件运行的稳定性，设计了数据处理终端软件的自我监控机制，重新设计了 SD 卡模块以应对网络中断的情况。

第五章 AETA 系统软件的测试与上线

软件测试是通过人工测试或者计算机自动化测试的手段来检验它是否满足软件的设计需求，软件测试的主要目的是在软件上线之前检测软件是否存在故障，降低软件上线的风险。软件测试是软件质量的保证。本文为 AETA 系统软件的测试开发了相关测试工具，并提出了相应的测试方案，对 AETA 系统软件做了单元测试与集成测试，最后将软件进行上线并分析了线上的运行效果。

5.1 测试工具的设计与开发

为了验证 AETA 软件系统数据链路的正确性，开发了模拟探头和模拟终两个测试工具。图 5.1 为模拟探头软件界面。模拟探头是基于 MFC 框架，采用 visual c++语言在 windows 系统下的 visual studio 平台编写而成。图 5.1 中，ip 地址为探头软件的固定 IP 地址，端口号为探头与终端之间数据传输的端口号，探头 ID 为探头的唯一标识，传感器编号用来区分该电磁模拟探头和地声模拟探头，频率为指定的模拟探头输出正弦波信号的频率。最大值和最小值为模拟探头输出信号电压幅度的范围，点数表示模拟探头发送的一个数据包中包含有多少个模拟数据，发送间隔为模拟探头数据包的发送间隔。

通过在模拟探头中设置对应的数值，可以产生幅度和频率可知的模拟正弦波信号。将模拟正弦波波信号相关参数的计算结果与经过整个 AETA 系统数据处理链路并最终得到的数据进行比较，可以判断 AEAT 系统数据处理链路的正确性。

模拟探头是为了给软件系统指定的输入激励，验证数据处理终端和 server 应用程序对数据的处理是否正确，保证探头之上整条数据链路的正确性。

图 5.2 为模拟终端软件的界面。通过模拟探头，可以验证终端软件和应用服务软件数据流通道的正确性，但是无法验证实际传感探头数据是否合理。因此，开发了模拟探头，用于接收并实时绘制探头软件的数据波形。模拟终端按照终端的部分需求，利用 visual C++ 基于 MFC 框架在 Windows 系统下的 visual studio 平台编写而成，用于接收实际传感探头的数据，并实时计算探头的信号最大值和最小值，峰值频率和峰值幅度。此外，模拟终端还可以将接收到的数据保存到 PC 端磁盘中，用于后期对数据的详细分析。

由图 5.2 可以看出，模拟终端软件主要分为连接区、配置区、数据接收状态、接收区、显示区和位于显示界面右侧的功能区六个部分。其中连接区设置模拟监听的端口

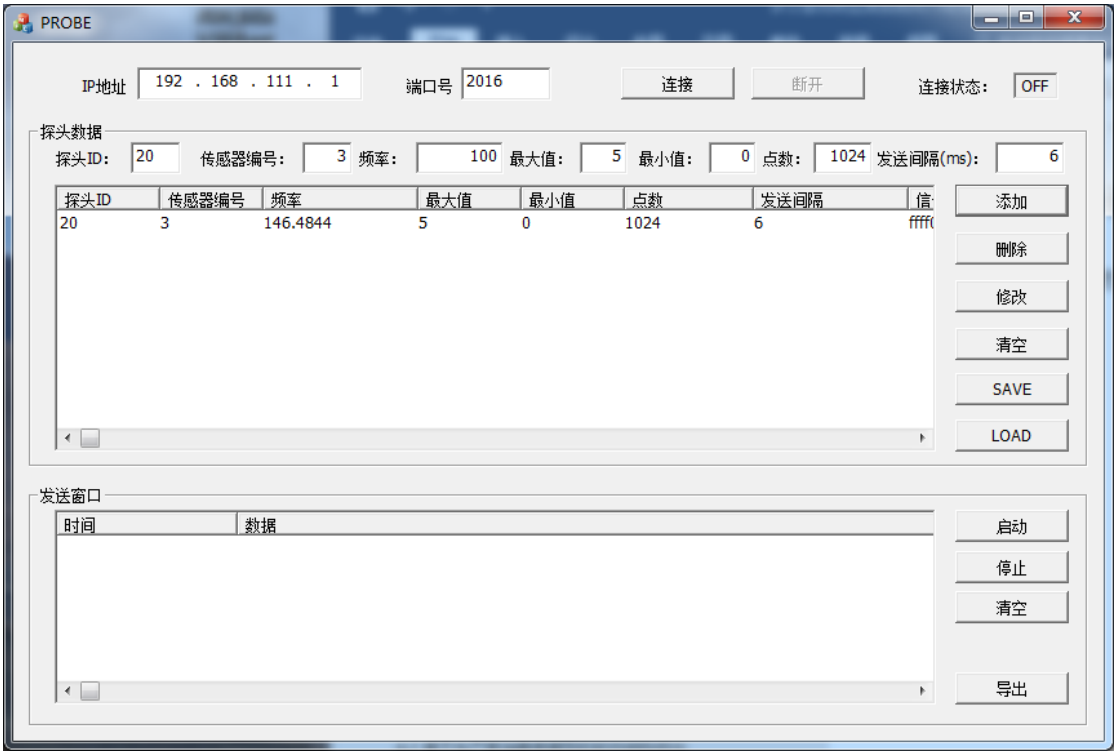

图 5.1　模拟探头软件界面

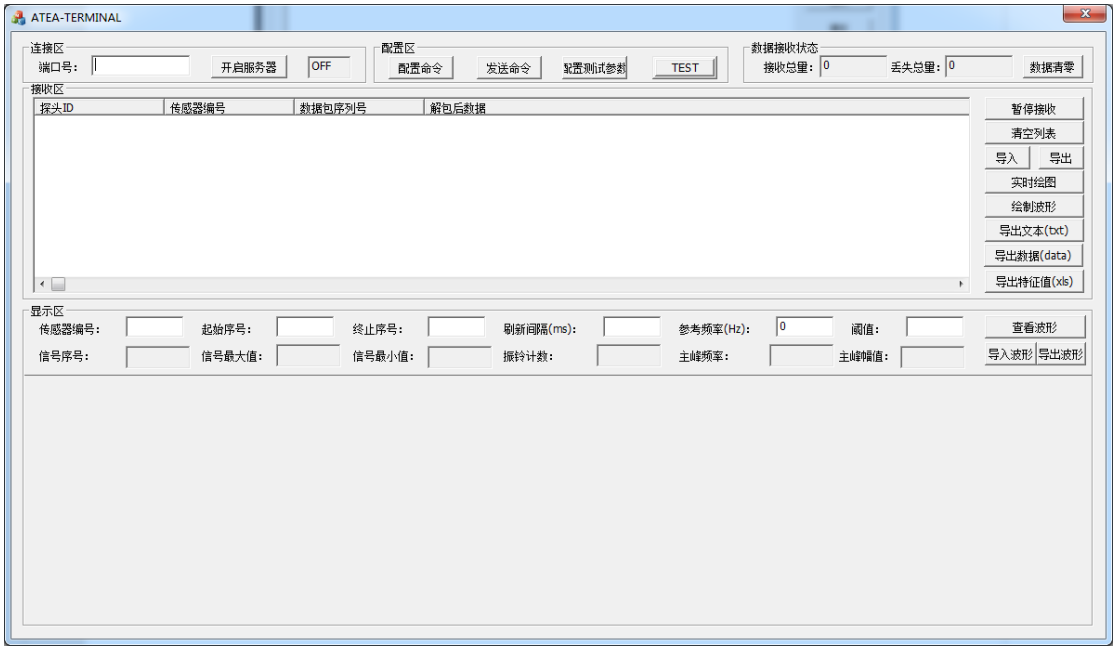

图 5.2　模拟终端界面

号，并且设有是否开始接收数据的按钮；配置区主要用于向传感探头下发命令和其它配置参数；数据接收状态主要用于显示模拟终端的接收数据总量和丢包数据总量。接收区用于显示接收到的传感探头数据，包括探头号，传感器编号，数据包序列号和解包后的原始数据。显示区主要用于显示传感数据的一些特征值，比如峰值频率，峰值幅度

等。此外，显示区下方的空白区域可以用来绘制实时数据波形。位于软件界面右侧的功能区，主要包括实时绘图、绘制波形、将数据导出为文本格式（txt）或者数据格式（data）、导出特征数据等功能。通过模拟终端，可以从原始数据分析、数据波、数据特征值等角度来验证模拟探头软件对数据的处理是否正确。

5.2 AETA 系统软件的测试方案及评价标准

本文主要对传感探头和终端内的软件进行测试，采用单元测试与集成测试相结合的测试方案进行测试。单元测试主要对 AETA 系统软件的各个单元接口进行测试，发现单元内可能存在的错误，此外还辅以代码审查的方式，对各个接口的代码进行详细的检查，包括程序格式、命名方式、内存管理、异常的处理等方面；

集成测试主要是验证 AETA 系统软件是否符合需求的设计，采用白盒测试与黑盒测试来对 AETA 系统软件进行集成测试；其中白盒测试主要对 AETA 系统的数据流进行测试，确保 AETA 系统数据处理链路的正确性；黑盒测试主要根据系统的需求，对 AETA 系统软件的功能进行测试。

AETA 系统软件只有经过测试且达到测试标准之后才能上线运行。AETA 系统软件的测试评价标准主要有以下三点：

(1) AETA 系统软件的各项功能能够正确执行。
(2) AETA 系统软件能够在连续的较长时间内稳定运行。
(3) 数据处理终端软件的内存占用率达到良好（≤70%）,CPU 使用率达到良好（≤70%）。

5.3 AETA 系统软件的单元测试

AETA 系统软件的单元测试采用静态测试方法，测试步骤如下：

(1) 对软件的各个接口按照 MISRA 规范进行代码审查。
(2) 对软件的各个接口进行测试，确保功能正确，并且能够进行异常处理。

AETA 系统软件的代码审查本节不作过多描述，表 5.1 列出部分需要进行测试的各个接口及测试结果。

表 5.1 AETA 系统软件部分测试接口

软件名称	接口名称	接口说明	测试结果
ProbeApp	Data_Receiving()	接收来自终端的数据	接口正常
ProbeApp	Data_Sending()	发送传感数据至终端软件	接口正常
DataTerminal	OpenGpio3_8()	打开 Gpio3_8	接口正常
DataTerminal	ChangeServer()	切换登录的服务	接口正常
DataTerminal	WriteLogToSd()	将日志写入 SD 卡	接口正常
DataTerminal	Datacompress()	数据压缩接口	接口正常
DataTerminal	Data_decode()	数据解压缩接口	接口正常
DataTerminal	MagnFir_H()	高频电磁滤波接口	接口正常
DataTerminal	MagnFir_L()	低频电磁滤波接口	接口正常
DataTerminal	SoundFir_H()	高频地声滤波接口	接口正常
DataTerminal	SoundFir_L()	低频地声滤波接口	接口正常
DataTerminal	Input()	FIFO 输入接口	接口正常
DataTerminal	Output()	FIFO 输出接口	接口正常
DataTerminal	HttpPostDataByFile()	通过文件发送数据接口	接口正常
DataTerminal	HttpPostDataByParamete()	通过参数发送数据接口	接口正常
DataTerminal	SD_Write_Data()	SD 卡存数据接口	接口正常
DataTerminal	SD_Write_Log()	SD 卡存日志接口	接口正常
DataTerminal	SD_Fetch_Data()	SD 卡读数据接口	接口正常
DataTerminal	SD_Fetch_Log()	SD 卡读日志接口	接口正常
SynApp	TimeSyn()	时间同步接口	接口正常
SynApp	PostLogFromSD()	发送日志接口	接口正常

5.4 AETA 系统软件的集成测试

采用白盒测试和黑盒测试两种方法对 AETA 系统软件进行集成测试。

5.4.1 AETA 系统软件的白盒测试

数据是整个 AETA 系统的核心资源，确保数据的正确性至关重要。5.1 节介绍了系统测试的相关工具。本节利用该工具搭建了如图 5.3 所示的测试平台，并利用这一个测试平台对 AETA 系统的数据链路进行了测试。

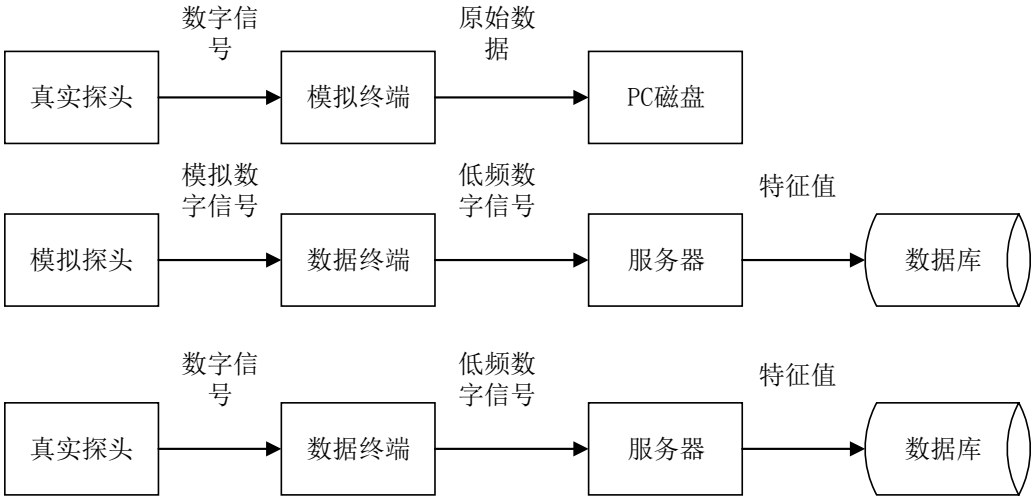

图 5.3　测试平台框图

表 5.2　部分测试结果

输入信号			理论结果				实际测试结果			
频率（Hz）	幅度（V）	直流分量（V）	全局峰值幅度（V）	全局峰值频率（Hz）	振铃计数（次/s）	均值（V）	全局峰值幅度（V）	全局峰值频率（Hz）	振铃计数（次/s）	均值（V）
0.9115	5	0	5	0.9115	0	3.18	4.9544	0.92	0	3.1543
14.6484	5	0	5	14.6484	14	3.18	4.9561	14.65	14	3.1551
58.59	5	0	5	58.59	58	3.18	4.9563	58.59	58	3.1547
117.1875	5	0	5	117.1875	117	3.18	4.9421	117.19	117	3.1443
234.375	5	0	5	234.375	234	3.18	4.9546	234.39	234	3.1542
468.75	5	0	5	468.75	468	3.18	4.9545	468.75	468	3.1515
937.5	5	0	5	937.5	937	3.18	4.9545	937.5	937	3.144
7500	5	0	5	7500	7500	3.18	4.9545	7500	7500	3.143
937.5	1	0	1	937.5	937	0.636	0.9908	937.5	937	0.6287
937.5	2	0	2	937.5	937	1.272	1.9817	937.5	937	1.2576
937.5	4	0	4	937.5	937	2.544	3.9636	937.5	937	2.5152
937.5	6	0	6	937.5	937	3.816	5.9456	937.5	937	3.7729

首先采用真实探头连接模拟终端进行测试，确保探头对数据的采集与处理是正确的。通过在电磁探头外加磁场激励或者在地声探头外加声音激励，模拟终端将接收到

的实时数据绘制成数据波形，并将数据存储在 PC 磁盘上。通过对数据波形的观测与原始数据的分析来判断探头对数据的处理是否正确。在确保探头对数据的采集与处理的是可靠的情况下，采用模拟探头连接数据终端，输入确定幅度和频率的正弦波信号，对数据终端、服务器、数据库整条数据链路进行测试。表 5.2 为采用模拟电磁探头对数据链路进行测试的结果。

在做特征值提取时，由于傅里叶变换分辨率的原因导致实际计算结果与理论值存在细微的误差。测试结果表明，在误差允许的范围内，数据处理终端软件和 server 应用程序软件对数据的处理流程是正确的。

5.4.2 AETA 系统软件的黑盒测试

(1) 登陆功能测试

数据处理终端与服务器端 server 应用程序之间采用 HTTP 协议通信，通过 POST 的方式传输数据和参数。数据处理终端首次与 server 应用程序建立连接 HTTP 连接后，server 应用程序需要对数据终端进行身份验证，确保该连接是合法请求连接。身份验证的方式是数据终端向 server 应用程序发送登陆请求，请求参数包括终端号和登陆密钥。server 应用程序接收到登陆请求后对终端号和登录密钥进行确认，在两者均正确的情况下，数据处理终端才能和 server 应用程序正常通信。由于 HTTP 协议是短连接，为了避免每次数据通信都需要进行登陆验证，采用 session 的方式。数据处理终端在第一次登陆服务器成功后，server 应用程序向数据处理终端返回一个 sessionID，以后每次数据处理终端与 server 应用程序通信只需带上 sessionID 即可。若 server 应用程序接收到登陆请求后发现终端号不合法或者登陆密钥错误，则 server 应用程序会返回一个登陆失败的响应。

测试 1：登陆密钥设置为“1234567890”，DataTerminal 向 server 应用程序发送的登陆请求参数为：terminalID=95&password=1234567890。

测试结果：server 应用程序的响应参数为：result=success&sessionId=95_ 1554015505。响应参数表示 95 号终端登陆服务器成功。图 5.4 为终端登陆成功的打印信息。

```
Starting login server...
contents:&terminalID=95&password=1234567890
url:http://120.25.154.201:8069/login
Login sucessfully!
Login finished!
exit loginmodule
```

图 5.4 登录成功打印信息

测试 2：设置错误的登陆密钥，DataTerminal 向 server 应用程序发送的登陆请求参数为：terminalID=95&password=12345。

测试结果：server 应用程序的响应参数为：result=success&faileCode =FC_002&failReason=terminalID or password was not correct。这条响应参数的意思即为 95 号终端登陆服务器失败，失败原因可能是终端号不合法或者登陆密钥错误。

(2) 状态上传功能测试

状态包括探头运行状态和终端运行状态。探头每隔 255s 向终端上传一次运行状态，而终端每 3 分钟向服务器上传一次状态。因此对状态的测试只需让 AETA 系统正常运行 5 分钟以上，观测数据库中是否更新了终端和探头的状态即可。测试结果如下图 5.5 和 5.6 所示。

TerminalID	ServerSysTime	TerminalSysTime	TerStartTime	TerSoftVersion	SDStatus	FreeSpace	NetworkStatus	DataRate	BadPack	StationID
18	2019-01-21 10:09:17	2019-01-21 10:09:14	2019-01-16 11:27:24	3.3.2	0	99.72	1	0	0	60018
20	2019-03-31 15:44:48	2019-03-31 15:44:46	1970-01-01 08:00:32	3.3.2	-1	0	1	389365	0	20
25	2019-03-31 15:46:48	2019-03-31 15:46:45	2019-03-21 13:35:20	3.3.2	0	99.72	1	405727	0	25
37	2019-03-31 15:46:49	2019-03-31 15:46:47	2019-03-31 00:52:47	3.3.2	0	99.72	1	405232	0	60037
44	2019-03-31 15:44:39	2019-03-31 15:44:36	2019-03-22 10:38:58	3.3.2	-1	0	1	397069	0	44
45	2019-01-11 17:13:07	2019-01-11 17:13:04	2019-01-10 17:22:03	3.3.2	0	99.25	1	408521	0	45
46	2019-03-26 02:31:45	2019-03-26 02:31:43	2019-02-19 10:33:18	3.3.2	-1	99.57	1	391925	0	46
52	2019-03-31 15:46:27	2019-03-31 15:46:24	2019-03-17 07:49:57	3.3.2	0	43.22	1	393870	0	52
53	2019-03-31 15:44:10	2019-03-31 15:44:07	2019-03-26 17:41:19	3.3.2	0	99.31	1	391310	0	53
54	2019-03-31 15:44:42	2019-03-31 15:44:40	2019-03-27 23:53:44	3.3.2	0	99.71	1	392632	0	54
58	2019-03-31 15:44:41	2019-03-31 15:44:38	2019-02-26 01:07:00	3.3.2	-1	0	1	392953	0	58
60	2019-03-31 15:46:24	2019-03-31 15:46:22	2019-03-15 11:43:55	3.3.2	0	99.25	1	399254	0	60
61	2019-03-31 15:45:47	2019-03-31 15:45:44	2019-02-19 11:08:33	3.3.2	0	99.45	1	391070	0	61
62	2019-03-31 13:27:34	2019-03-31 13:27:31	2019-03-31 01:20:30	3.3.2	-1	0	1	390543	0	62
63	2019-02-27 16:06:56	2019-02-27 16:07:51	2019-02-27 16:04:49	3.3.2	-1	0	1	393078	0	63
65	2019-03-31 15:45:07	2019-03-31 15:45:04	2019-03-10 04:33:33	3.3.2	-1	0	1	392441	0	65
66	2019-03-31 15:46:42	2019-03-31 15:46:39	2019-02-28 16:17:55	3.3.2	0	99.26	1	390922	0	66

图 5.5　终端状态显示

ProbeID	ProbeType	TerminalID	ServerSysTime	TerminalSysTime	ProSoftVersion	DataEnable	SyncInterVal	ZeroCorrectValue	DataAmpTimes	StationID
258	2	176	2019-03-31 15:48:48	2019-03-31 15:47:57	3.0.1	1	255	0	0	176
30161	1	176	2019-03-31 15:48:48	2019-03-31 15:47:22	3.0.1	1	255	0	0	176
49	2	61	2019-03-31 15:48:46	2019-03-31 15:47:05	3.0.1	1	255	0	0	61
30086	1	61	2019-03-31 15:48:46	2019-03-31 15:46:29	3.0.1	1	255	0	0	61
67	2	92	2019-03-31 15:48:34	2019-03-31 15:44:45	3.0.1	1	255	0	0	92
30126	1	92	2019-03-31 15:48:34	2019-03-31 15:44:44	3.0.1	1	255	0	0	92
54	2	79	2019-03-31 15:48:32	2019-03-31 15:44:44	3.0.1	1	255	0	0	79
30059	1	79	2019-03-31 15:48:31	2019-03-31 15:48:13	3.0.1	1	255	0	0	79
43	2	70	2019-03-31 15:48:24	2019-03-31 15:48:08	3.0.1	1	255	0	0	70
30087	1	70	2019-03-31 15:48:24	2019-03-31 15:48:06	3.0.1	1	255	0	0	70
30078	1	74	2019-03-31 15:48:20	2019-03-31 15:47:19	3.0.0	1	255	0	0	74
111	2	196	2019-03-31 15:48:19	2019-03-31 15:44:07	3.0.1	1	255	0	0	100
30119	1	196	2019-03-31 15:48:19	2019-03-31 15:44:05	3.0.1	1	255	0	0	100
175	2	222	2019-03-31 15:48:15	2019-03-31 15:46:56	3.0.1	1	255	0	0	222
30226	1	222	2019-03-31 15:48:15	2019-03-31 15:46:55	3.0.1	1	255	0	0	222
68	2	72	2019-03-31 15:48:12	2019-03-31 15:46:25	3.0.1	1	255	0	0	72
30090	1	72	2019-03-31 15:48:12	2019-03-31 15:46:23	3.0.1	1	255	0	0	72
88	2	65	2019-03-31 15:48:08	2019-03-31 15:46:59	3.0.1	1	255	0	0	65

图 5.6　传感探头状态显示

连续采集版本的 AETA 系统软件已经运行半年左右的时间，状态功能一直运行正常。

(3) 命令通道测试

在数据库命令配置表中对应的终端条目处填写好给该终端下发的命令，并将 Newflag 置为 1，表示需要给该终端下发命令。终端每隔一定会时间向 server 应用程序

请求命令。server 应用程序在接收到终端的命令查询请求后，会向数据库查询是否有给该终端的命令，若有，则从数据库中读取命令并将它作为终端请求的响应参数发送给终端，终端解析 server 应用程序的响应参数得到命令信息，并根据命令类型执行相应的操作，命令执行结果会以日志的形式记录下来，由 SynApp 发送至 server 应用程序并最终存入数据库。

测试步骤：

在数据库 command 表中 95 号终端条目处，添加命令“ls /opt/”，并将对应的 Newflag 置为 1。为尽快查看终端执行结果，可从终端打印信息中观察终端的命令执行结果。

测试结果：

测试步骤中的结果如图 5.7 所示。此外在数据库中也可查到相应日志为：

03-31,16:23:51,ls /opt/:19_20.data 19_30027.data 22_24.data 22_30025.data 24_17.data 26_30017.data 26_56.data 29_23.data 29_30016.data 32_30019.data Config.ini ConfigureTmp HardCompati。

通过比较可以发现，图 5.7 打印信息中显示的命令执行结果与数据库中记录的命令执行结果一致。此外，通过 xshell 软件进入终端系统中，输入 “ls /opt/”命令，系统输出的命令执行结果与图 5.7 打印信息中的结果一致。

```
Receive cmd:ls /opt/
device = 1
deviceID = 95
start to run cmd
ls /opt/
the logtime is 1554020571: and the device is 1:
starting write log to the sd card
cmd result:
ls /opt/:19_20.data 19_30027.data 22_24.data 22_30025.data 24_17.data 26_30017.data
.data Config.ini ConfigureTmp HardCompatible.ini HardTm
```

图 5.7 终端接收命令打印信息

(4) 数据上传功能测试

数据上传是终端的核心业务功能，对该功能进行测试主要是验证数据处理终端是否能够向 sever 应用程序正常发送探头传感数据，server 应用程序接收终端发来的数据，并对数据进行处理后，是否能够将原始数据和特征数据正常入库保存。

测试步骤：将电磁探头和地声探头接上终端设备，并保证终端网络的正常。

测试结果：图 5.8 中方框处的打印信息表示终端向 server 应用程序发送数据且数据库成功接收的数据。在数据库中可以查看到这两条数据的特征数据，原始数据存储在服务器磁盘中，并由数据下载脚本定时下载到原始数据存储磁盘阵列。图 5.9 为在数据库中查询到的特征数据。

结合图 5.8 和 5.9 可知，终端能够正常接收探头数据并将数据发送至 sercer 应用程序，server 应用程序能够正常接收终端上传的数据，并将数据存入数据库。

```
65 lowbanddata_1min is: 60000
starting to send real time data
Get time[1554021121] probe[65] component[1] raw full-freq data 60000 bytes
probe[65] lowband length is:60000
probe[65] Post 1 type data...
result:success
postTime[1554021121] No.1 data sucessfully!
probe[65] there is no low_band_data need to be written to sd card!
probe[32000] -16384
probe[65] -7082
probe[32000] -16384
probe[65] -10166
probe[32000] -16384
probe[65] -15243
probe[32000] -16384
32000 lowbanddata_1min is: 60000
probe[65] -11967
starting to send real time data
Get time[1554021125] probe[32000] component[3] raw full-freq data 60000 bytes
probe[32000] lowband length is:60000
probe[32000] Post 3 type data...
result:success
postTime[1554021125] No.3 data sucessfully!
probe[32000] there is no low_band_data need to be written to sd card!
```

图 5.8　数据发送打印信息

Time	TerminalID	ProbeID	average	Ring_down_count	peak_frequency	peak_amplitude	TimeSave
1554021121	95	65	3.4193	50	49.99	4.9991	2019-03-31 16:32:04

Time	TerminalID	ProbeID	average	Ring_down_count	peak_frequency	peak_amplitude	TimeSave
1554021125	95	32000	2.2529	0	0.24	0.1013	2019-03-31 16:32:08

图 5.9　数据库中存储的特征数据

(5) 弱网络环境处理测试

在网络环境较差或者断网时，数据处理终端会将无法发出去的数据在终端 SD 卡中做持久化处理，并且在网络恢复之后，从 SD 卡中读取数据，重新发送至服务器。

测试步骤：

a) 断开数据处理终端的网络，通过终端打印信息和日志检测终端是否将数据存储进 SD 卡中；

b) 使得数据处理终端重新连接网络，通过打印信息和日志检测终端是否重新上传历史数据；

测试结果：

步骤 a)的测试结果如图 5.10 所示，在断网数据处理终端的网络，之后，数据处理终端将数据写入了 SD 卡中，进入 SD 卡目录中，发现数据确实被写入 SD 卡中。

```
starting to write Low-band data to sd
Write time[1554024071] probe[32000] component[3] low-band data 60016 bytes to SD
the logtime is 1554024072: and the device is 1:
starting write log to the sd card
probe[65] 3093
65 lowbanddata_1min is: 60000
probe[32000] -16384
probe[65] 15498
probe[32000] -16384
Starting login server...

contents: &terminalID=95&password=1234567890
url: http://120.25.154.201:8069/login
Send login request failed, return value = 7.
the logtime is 1554024074: and the device is 1:
starting write log to the sd card
starting to write Low-band data to sd
Write time[1554024073] probe[65] component[1] low-band data 60016 bytes to SD
the logtime is 1554024074: and the device is 1:
starting write log to the sd card
```

图 5.10　数据写入 SD 卡打印信息

连接网络后，如图 5.11 和图 5.12 所示，数据处理终端从 SD 卡中重新读取这两条数据并发送至服务器；

```
starting get data to send from SD card
data length from sd card is:60016
Read ID[95] time[1554024071] probe[32000] component[3] data 60016 bytes from SD
probe[32000] Post 3 type data...
probe[65] 12453
probe[32000] -10416
result:success
postTime[1554024071] No.3 data sucessfully!
```

图 5.11　终端补传地声数据打印信息

```
starting get data to send from SD card
data length from sd card is:60016
Read ID[95] time[1554024073] probe[65] component[1] data 60016 bytes from SD
probe[65] Post 1 type data...
result:success
postTime[1554024073] No.1 data sucessfully!
```

图 5.12　终端补传电磁数据打印信息

在数据库中对数据和日志进行查询，如图 5.13、图 5.14 和图 5.15 所示。

Time	TerminalID	ProbeID	average	Ring_down_count	peak_frequency	peak_amplitude	TimeSave
1554024073	95	65	3.8066	50	49.99	3.6331	2019-03-31 17:23:06

Time	TerminalID	ProbeID	average	Ring_down_count	peak_frequency	peak_amplitude	TimeSave
1554024071	95	32000	2.4885	0	0.12	0	2019-03-31 17:22:05

图 5.13　数据库中存储的特征数据

```
03-31,17:22:14,Write time[1554024073] probe[65] component[1] low-band data 60016 bytes to SD
03-31,17:22:12,Write time[1554024071] probe[32000] component[3] low-band data 60016 bytes to SD
```

图 5.14　数据存入 SD 卡日志

```
03-31,17:24:03,Send time[1554024073] probe[65] component[1] data successfully from SD.
03-31,17:16:43,Changed server url to http://120.25.154.201:8069.
03-31,17:16:43,Send login request failed, return value = 7.
03-31,17:23:03,Send time[1554024071] probe[32000] component[3] data successfully from SD.
03-31,17:23:03,Login server successfully!
```

图 5.15　数据补发日志

通过上述测试分析，数据处理终端在弱网络环境下能够将数据存入终端 SD 卡内进行持久化存储，待网络环境恢复良好之后，数据处理终端软件会从 SD 卡中读取历史数据重新发送到服务器。

(6) 时间校准测试

终端硬件之间的差异是不可避免的，这将导致不同终端设备之间可能在系统时间上存在差异，而数据的时间戳是在终端软件 DataTerminal 上添加的。因此，为保证全国各个台站终端系统时间的一致性，终端需要定时与同一台服务器的时间进行同步。终端每隔 1 小时向 server 应用程序发送终端当前的系统时间，server 应用程序接收到时间同步请求后，将获取 server 应用程序所在服务器的系统时间，并作为响应发送至终端程序 SynApp，SynApp 接收到 server 应用程序的响应时间后，修改系统当前时间。

测试步骤：

启动 SynApp 进程，server 应用程序在启动时会向 server 应用程序发送时间同步请求，请求参数为：time= 1554033185 。

测试结果：

SynApp 接收到 server 应用程序的响应为：time=1554033186 ，终端系统的 UNIX 时间戳为 1554033185，转化为日期即 2019-03-31 19:53:05，server 应用程序响应的服务器时间为 1554033186，转化为日期即 2019-03-31 19:53:06，两者相差 1s。SynApp 在接收到 server 应用程序的时间同步响应后，解析响应参数中的服务器时间，并将终端时间修改为服务器时间，并将时间同步操作通过日志记录。

在进行时间同步测试之后，查看数据库的日志如下：

Local time is modified from 2019-03-31 19:53:05 to 2019-03-31 19:53:06。说明时间同步成功。

(7) 日志上报测试

终端软件 DataTerminal 会以日志的形式记录运行过程中出现的关键事件，由 SynApp 对记录的日志进行处理并发送至 server 应用程序。

测试步骤：

由于数据处理终端软件 DataTerminal 在进程刚启动时会记录终端系统启动的相关日志，在有探头连接上终端时也会记录相关日志信息，因此对日志的测试只需将终端断电重启，然后查看数据库中是否有相应日志即可。

Time	TerminalID	Num	LogType	Length	Event
1554621690	95	58962019	5	58	04-07,15:22:30,Post terminal state sessionError->Invalid!
1554621692	95	58962017	5	42	04-07,15:22:32,Login server successfully!
1554621672	95	58962014	3	68	04-07,15:22:12,Local time is modified from 1554621672 to 1554621672
1554621449	95	58962012	5	133	04-07,15:18:29,Post Time[1554621397] No.3 data failed, code:FC_004, failreason:the account(29997) of data mismatch the dataType(3)!.
1554621399	95	58961993	5	97	04-07,15:17:39,Write time[1554621397] probe[32000] component[3] low-band data 60010 bytes to SD
1554621399	95	58961991	5	133	04-07,15:17:39,Post Time[1554621397] No.3 data failed, code:FC_004, failreason:the account(29997) of data mismatch the dataType(3)!.
1554621327	95	58961989	5	53	04-07,15:16:27,Terminal system start,version:3.3.2!)
1554621340	95	58961987	5	37	04-07,15:16:40,one probe start,fd=12
1554621337	95	58961986	5	37	04-07,15:16:37,one probe start,fd=11
1554621329	95	58961985	5	42	04-07,15:16:29,Login server successfully!
1554621323	95	58961984	3	68	04-07,15:16:23,Local time is modified from 1554621323 to 1554621323

图 5.16　数据终端日志

图 5.16 为 95 号终端重启后，在数据库 terminallog 表中查看到的日志。从图 5.16 中可以看出，终端软件系统记录了时间同步日志信息，登陆服务器成功日志信息，探头启动日志信息，终端启动日志信息，此外，还记录了一些软件运行过程中遇到的异常信息比如，sessionError 等。由此可看出，AETA 系统软件的日志上报功能运行正常，能够将终端运行过程中的关键事件记录并最终保存到数据库中。

(8) 监控功能的测试

SynApp 还具有监控 DataTerminal 运行状态的功能，同时 SynApp 自身受系统硬件看门狗的监控。对监控功能的测试主要是检测当 DaTerminal 程序或者 SynApp 程序运行奔溃或者卡死时，终端系统是否能够自行恢复正常运行状态。

测试步骤：

a) 启动数据处理终端，通过 Xshell 软件进入到终端的嵌入式 Linux 操作系统中，查看 SynApp 和 DataTerminal 的进程 ID；

b) 待终端启动 1 分钟以后，通过“kill -9 x”命令，手动结束 DataTerminal 进程，等待 1 分钟之后，再次查看系统是否存在 DataTerminal 的进程 ID。

c) 与步骤 2 类似，手动结束 SynApp 进程，观察系统是否重启；

d) 在 DataTerminal 的主函数中添加延时函数，延时 60s，让 DataTerminal 进程模拟产生卡死的状态。观察 DataTerminal 进程 ID 的变化。

测试结果：

a) 系统启动后，SynApp 和 DataTerminal 的进程 ID 均存在；

b) DataTerminal 进程被手动结束后，SynApp 检测到 DataTerminal 进程的心跳信息停止，重新启动了 SynApp 进程，在系统中又能够查到 DataTerminal 的进程 ID。

c) SynApp 进程被手动结束之后，终端会重启；

d) 处于“卡死”状态的 DataTerminal 进程将被 SynApp 进程 kill 掉，然后 SynApp 会

重新启动 SynApp 进程。

上述测试表明，终端系统具有较好的监控机制，保证终端系统的长期稳定运行。

(9) 升级功能的测试

UpdateApp 是运行在终端内的升级程序，主要完成终端内相关文件的远程升级功能。

测试步骤：在服务器端部署好需要升级的文件，然后远程重启终端，等待约 3 分钟左右，观察终端版本号的变化。

85	2019-04-07 17:07:36	2019-04-07 17:07:33	2019-03-26 05:20:03	3.3.2	
86	2019-04-07 17:09:58	2019-04-07 17:09:56	2019-04-03 09:43:06	3.3.2	
89	2017-08-25 16:56:52	2017-08-25 16:56:49	2017-08-25 16:50:39	3.0.5	
92	2019-04-07 17:08:04	2019-04-07 17:08:01	2019-04-01 16:17:13	3.3.2	
95	2019-04-07 17:10:27	2019-04-07 17:10:25	2019-04-07 17:06:07	3.2.6	
97	2019-04-07 17:07:37	2019-04-07 17:07:35	2019-03-14 09:50:30	3.3.2	
100	2019-04-07 17:09:17	2019-04-07 17:09:14	2019-04-06 18:15:15	3.3.2	
102	2018-06-18 22:00:50	2018-06-18 22:00:47	2018-06-18 10:21:45	3.3.0	

图 5.17　终端状态表（升级前）

85	2019-04-07 17:13:37	2019-04-07 17:13:33	2019-03-26 05:20:03	3.3.2	-1
86	2019-04-07 17:12:58	2019-04-07 17:12:56	2019-04-03 09:43:06	3.3.2	-1
89	2017-08-25 16:56:52	2017-08-25 16:56:49	2017-08-25 16:50:39	3.0.5	0
92	2019-04-07 17:14:04	2019-04-07 17:14:01	2019-04-01 16:17:13	3.3.2	0
95	2019-04-07 17:15:12	2019-04-07 17:15:10	2019-04-07 17:12:07	3.3.2	0
97	2019-04-07 17:13:38	2019-04-07 17:13:36	2019-03-14 09:50:30	3.3.2	0
100	2019-04-07 17:15:18	2019-04-07 17:15:15	2019-04-06 18:15:15	3.3.2	0
102	2018-06-18 22:00:50	2018-06-18 22:00:47	2018-06-18 10:21:45	3.3.0	0

图 5.18　终端状态表（升级后）

测试结果：从图 5.17 和图 5.18 可以看出，95 号终端在升级前软件版本为 3.2.6，升级后为 3.3.2，因此，终端程序升级成功，表明 UpdateApp 能够正常完成升级功能。

5.5　AETA 系统软件的上线与效果展示

5.5.1　AETA 系统软件的性能分析

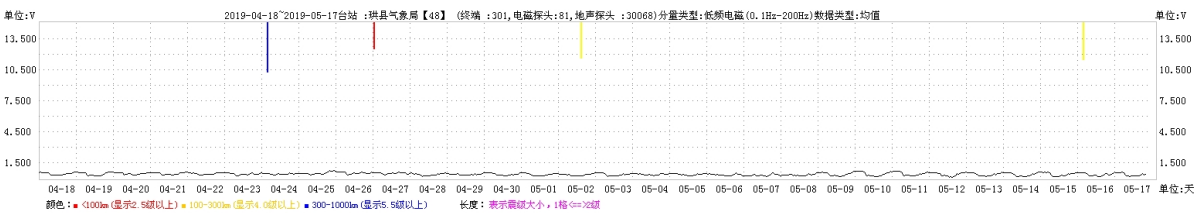

图 5.19　AETA 系统数据展示网页截图

图 5.19 为珙县气象局台站近一个月的电磁均值。由图可知，该台站近一个月的数据连续无断点，表明 AETA 系统软件可长时间稳定运行。

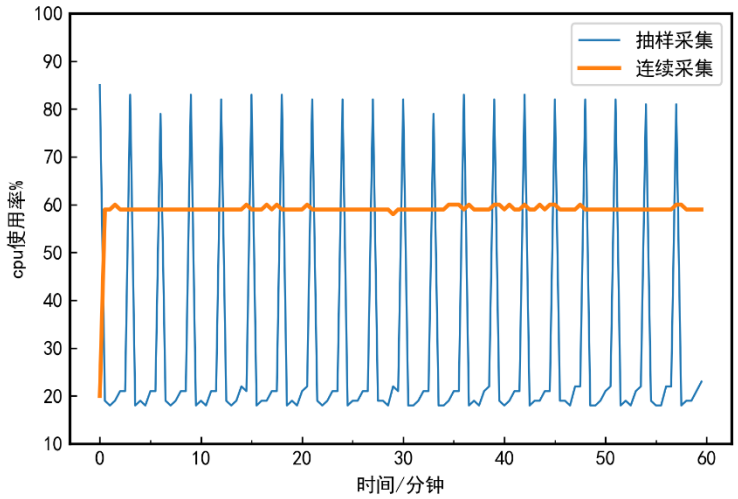

图 5.20　连续采集与抽样采集数据处理终端软件 cpu 使用率

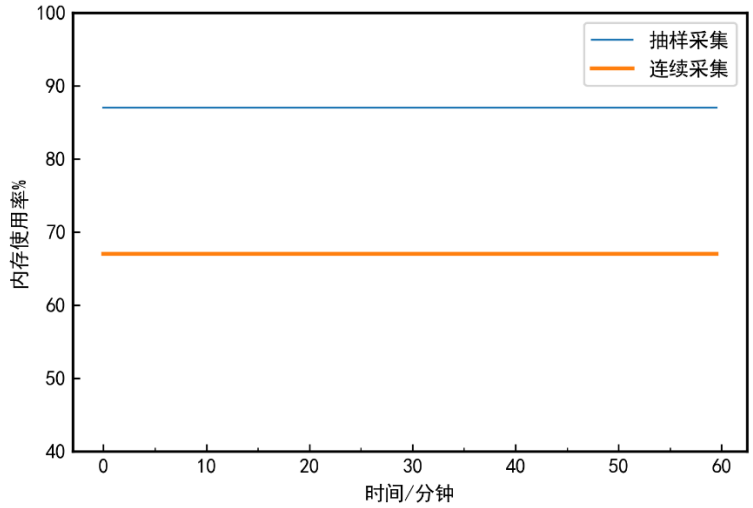

图 5.21　连续采集与抽样采集数据处理终端软件内存使用率

图 5.20 和 5.21 分别表示连续采集与抽样采集数据处理终端软件的 cpu 使用率和内存占用率。由两个图可以看出，连续采集数据处理终端软件的 cpu 占用率为 59%左右，内存占用率为 67%，符合 5.2 节提出的测试评价标准。

通过与抽样采集数据处理终端软件进行对比，可以发现，连续采集数据处理终端软件的 cpu 占用率较为稳定，而抽样采集数据处理终端软件的 cpu 占用率呈周期性变化，且峰值要远高于连续采集的 cpu 占用率；此外，连续采集数据处理终端软件的平均

内存占用率为也要远小于抽样采集。

5.5.2 AETA 系统软件的线上运行效果展示

通过对 AETA 系统软件的单元测试、集成测试以及性能分析可知，AETA 系统能够较好地完成软件的需求，同时也能在较长时间内稳定运行，且数据处理终端软件的 cpu 和内存使用情况均为良好，达到了上线标准。

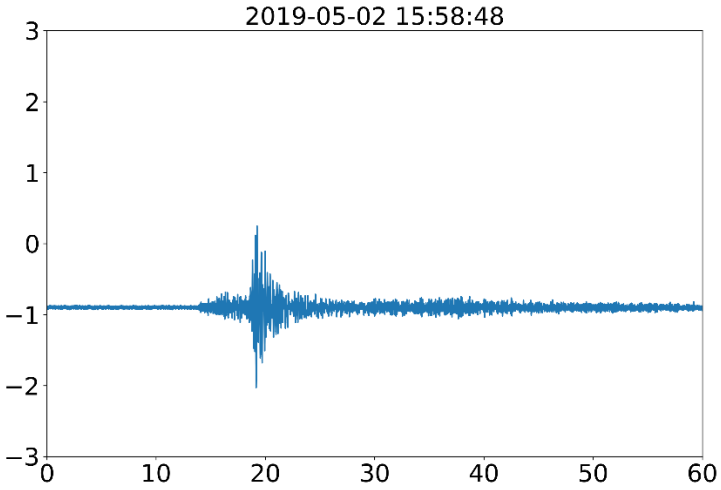

图 5.22　雅安市名山区安吉村原始数据

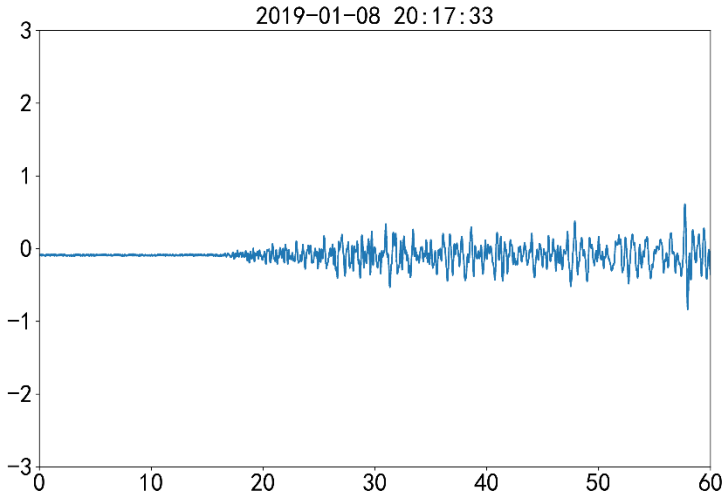

图 5.23　台湾横山观测点原始数据

图 5.22 和图 5.23 均为连续采集版本的 AETA 系统软件上线后采集到的原始数据。其中图 5.22 为 2019-05-02 15:57:58 庐山 4.5 级地震后采集到的地声原始数据，该台站距震中 29.8 公里，图 5.23 为 2019-01-08 20:16:45 台湾台东县海域 4.7 级地震后采集到

的地声原始数据，震中距 174.8km。对于抽样采集软件而言，如果地震恰好发生在数据采集窗口之外，则 AETA 系统将采集不到这部分数据。因此，AETA 系统软件升级为连续采集版本后，能够采集到质量较好的连续地震监测数据，有利于地震数据分析工作的开展。

5.6 本章小结

本章为 AETA 系统软件的测试开发了相关的测试工具，包括模拟终端和模拟探头，并提出了对 AETA 系统软件的测试方案和测试评价标准，从单元测试和集成测试两个方面对 AETA 系统软件进行测试。测试结果表明，AETA 系统软件能够达到测试评价标准。此外，本章还对抽样采集和连续采集数据处理终端软件的性能进行分析，分析表明，连续采集数据处理终端软件在 cpu 和内存占用率上要优于抽样采集数据处理终端软件。最后，展示了连续采集版本 AETA 系统软件采集到的原始数据，实践表明，连续采集版本的 AETA 系统软件能够获取质量更高的地震监测数据。

第六章 总结与展望

6.1 本文总结

面对 AETA 系统软件数据采集方式从抽样采集到连续采集升级的需求，面对 AETA 系统软件在一年多运行过程中积累的经验和表现出的不足，本文旨在对 AETA 系统软件进行全面的优化升级与全新设计。在阅读了大量相关文献和调研了最新技术之后，提出、设计并实现了连续采集版本的数据处理终端软件和数据采集探头软件，既适用于大地震临震监测系统的数据处理与通信，又可应用于断裂带活动的监测。基于本文的论述，采用 C++语言在 Linux 环境下对数据处理终端软件进行了编码实现与调试，采用 C 语言编码实现了数据采集探头软件；在软件的设计和编码过程中，对软件进行了多项优化。下面总结本文的主要工作：

(1) 本文分析了多分量地震监测预测系统 AETA 的研究背景、系统框架、数据指标和系统布设现状，在对国内外数据采集软件的研究现状进行调研后，提出了连续采集数据处理终端和数据采集探头软件的需求，并根据需求设计了探头软件和终端软件的整体框架。

(2) 设计并实现了连续采集数据处理终端软件 DataTerminal、日志处理和系统监控软件 SynApp、在线升级软件 UpdateApp 和探头数据采集软件。DataTerminal 负责实时接收传感探头的数据，并对数据进行实时解包、滤波或者压缩，最后将处理得到的数据按照约定的数据协议发送到服务器端；SynApp 完成日志数据的处理与发送，同时还负责对 DataTerminal 程序的监控，保障终端软件运行的可靠性和稳定性；UpdateApp 完成终端系统内应用文件的升级功能。数据采集探头软件负责读取传感探头 ADC 的数据并将数据打包发送至终端。

(3) 设计了基于位重组思想的地震数据无损压缩算法，用于数据的传输与存储，降低了数据传输和存储的成本。

(4) 采用生产者消费者模型设计了数据处理终端的接收、解包和滤波三个模块，提高了 CPU 的利用率。

(5) 采用线程池技术对发送模块进行设计，可以有效提高数据传输的稳定性，并降低服务器的处理压力。

(6) 设计了日志管理系统，规范化日志的描述和发送，既降低了终端数据请求的高并发量，又能够有效监测软件的运行状态，提高运维工作的效率。

(7) 对 Server 应用程序的系统逻辑框架进行了优化，按照不同的业务类型创建不同的

线程池，并由调度器来对终端的请求进行分类并交由相应的业务线程进行处理，若现有线程已达到线程池上限，则快速向终端返回无法处理的响应，避免 server 应用程序无限制创建线程。

(8) 设计并开发了测试工具，对 AETA 系统软件进行了测试，保证了系统运行的正确性和稳定性。

6.2 未来展望

连续数据采集版本的 AETA 系统软件已经稳定运行半年时间。结合 AETA 系统的实际应用情况，对今后进一步的工作进行规划和展望。

(1) 数据处理终端与 server 应用程序之间直接通过建立 HTTP 连接进行通信；随着系统布设规模的进一步扩大，单台服务器难以支撑所有终端的数据请求，因此需要考虑设计一种分布式系统架构，并采用负载均衡策略，平衡多台服务器之间的压力。

(2) AETA 系统软件升级与发布机制依旧不够完善。后期可以设计一个灰度升级系统，使得软件的升级流程自动化，升级方式可靠、可控。

(3) 在监测站点的安装时，设备的注册依旧需要大量的人工后台操作，为简化设备安装流程，需要开发相应的设备安装辅助程序。

参考文献

[1] 陈运泰. 地震预测:回顾与展望[J]. 中国科学(D 辑:地球科学), 2009, 39(12): 1633-1658.

[2] 徐纪人, 赵志新. 汶川 8.0 级大地震震源机制与构造运动特征[J]. 中国地质, 2010, 37(04): 967-977.

[3] 张国民. 我国的地震灾害和震灾预防[J]. 科学对社会的影响, 1999, (02): 44-47.

[4] 林科, 王新安, 张兴等. 一种适用于大地震临震预测的地声监测系统[J]. 华南地震, 2013, 33(04): 54-62.

[5] 王新安, 雍珊珊, 黄继攀等. 基于 AETA 监测数据的地震预测研究[J]. 北京大学学报(自然科学版), 2019, (02): 209-214.

[6] Y.Chen, D.C.Booth, 李万金. 《2008 年汶川地震:灾难剖析》第一章:汶川地震[J]. 世界地震译丛, 2019, (03): 266-302.

[7] Mantari A A, Wenceslao G, Segura Z, et al. Descriptive Study of the Ionosphere Charge Variation as a Preamble of Worldwide Earthquakes During 2016[C]. 2018 IEEE Sciences and Humanities International Research Conference (SHIRCON), 2018: 1-4.

[8] 王新安, 雍珊珊, 徐伯星等. 多分量地震监测系统 AETA 的研究与实现[J]. 北京大学学报(自然科学版), 2018, 54(03): 487-494.

[9] Sigurj, Oacute, Oacute N, et al. Importance of post-seismic viscous relaxation in southern Iceland[J]. Nature Geoscience, 2008, 1(2): 136-139.

[10] Peng Z, Peng Z. Migration of early aftershocks following the 2004 Parkfield earthquake[J]. Nature Geoscience, 2009, 2(12): 877-881.

[11] Lucazeau F, Leroy S, Bonneville A, et al. Persistent thermal activity at the Eastern Gulf of Aden after continental break-up[J]. Nature Geoscience, 2008, 1(12): 854-858.

[12] Brinkman B A, Leblanc M, Ben-Zion Y, et al. Probing failure susceptibilities of earthquake faults using small-quake tidal correlations[J]. Nature Communications, 2015, 6: 6157.

[13] Bouchon M, Durand V, Marsan D, et al. The long precursory phase of most large interplate earthquakes[J]. Nature Geoscience, 2013, 6(4): 299-302.

[14] Ammon C J, Hiroo K, Thorne L. A great earthquake doublet and seismic stress transfer cycle in the central Kuril islands[J]. Nature, 2008, 451(7178): 561-5.

[15] Wang J-C. The preliminary study of seismic energy release in Taiwan[C]. 2018 IEEE International Conference on Applied System Invention (ICASI), 2018: 220-223.

[16] Andrén M, Stockmann G, Skelton A, et al. Coupling between Mineral Reactions and Chemical Changes in Groundwater before and after Earthquakes in Iceland[C]. Agu Fall Meeting, 2015.

[17] Brodsky E E, Thorne L. Geophysics. Recognizing foreshocks from the 1 April 2014 Chile earthquake[J]. Science, 2014, 344(6185): 700-2.

[18] Green H W, Wang-Ping C, Brudzinski M R. Seismic evidence of negligible water carried below 400-

km depth in subducting lithosphere[J]. Nature, 2010, 467(7317): 828-831.

[19] Garcia A M, Aguilar A N P. Affordable Instrument Design for Seismic Monitoring, Early Warning Systems and Control Actions to Risk Mitigation[C]. 2018 13th APCA International Conference on Control and Soft Computing (CONTROLO), 2018: 143-147.

[20] Pierleoni P, Marzorati S, Ladina C, et al. Performance Evaluation of a Low-Cost Sensing Unit for Seismic Applications: Field Testing During Seismic Events of 2016-2017 in Central Italy[J]. IEEE Sensors Journal, 2018, 18(16): 6644-6659.

[21] 张肇诚, 张炜. 地震预报可行性的科学与实践问题讨论[J]. 地震学报, 2016, 38(04): 564-579+658.

[22] 刘瑞丰. 中国地震台网的建设与发展[J]. 地震地磁观测与研究, 2016, 37(04): 2+201.

[23] Li Z, An J, Yin H, et al. Study on Association Rules Between Earthquake Event and Earthquake Precursory Information Anomalies[C]. 2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), 2018: 1-6.

[24] 邱泽华, 张宝红. 我国钻孔应力-应变地震前兆监测台网的现状[J]. 国际地震动态, 2002, (06): 5-9.

[25] 邱泽华, 石耀霖. 国外钻孔应变观测的发展现状[J]. 地震学报, 2004, (S1): 162-168+176.

[26] Kong X, Li N, Lin L, et al. Relationship of Stress Changes and Anomalies in OLR Data of the Wenchuan and Lushan Earthquakes[J]. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 2018, 11(8): 2966-2976.

[27] 池顺良. 日本 9 级大震前我国钻孔应变网测到两起地块强烈受压事件[J]. 地球物理学进展, 2011, 26(05): 1583-1587.

[28] Tan Q, Liu Q, Sun Z. Research and Application of Beijing Earthquake Disaster Prevention System Based on GIS[C]. 2018 IEEE International Conference on Computer and Communication Engineering Technology (CCET), 2018: 275-279.

[29] Wang X, Dou A, Ding X, et al. The Development of Rapid Extraction and Publishing System of Earthquake Damage Based on Remote Sensing[C]. IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium, 2018: 2921-2924.

[30] Hashimoto C, Noda A, Sagiya T, et al. Interplate seismogenic zones along the Kuril–Japan trench inferred from GPS data inversion[J]. Nature Geoscience, 2009, 2(2): 141.

[31] Li X, Ge M, Zhang Y, et al. New approach for earthquake/tsunami monitoring using dense GPS networks[J]. Scientific Reports, 2013, 3(38): 2682.

[32] 孟国杰, 苏小宁, 徐婉桢等. 基于 GPS 观测研究 2010 年青海玉树 M_S7.1 地震震后地壳形变特征及其机制[J]. 地球物理学报, 2016, 59(12): 4570-4583.

[33] Khaerani D, Sarsito D A, Meilano I. Deformation of West Sumatra Due to the 2016 Earthquake (M7.8) Based on Continuous GPS Data[C]. 2018 IEEE Asia-Pacific Conference on Geoscience, Electronics and Remote Sensing Technology (AGERS), 2018: 1-7.

[34] 刘洋, 许才军, 温扬茂. 两次大柴旦 Mw6.3 地震间地表形变的 InSAR 观测及与同震破裂的联合分析[J]. 大地测量与地球动力学, 2016, 36(02): 110-114+119.

[35] Vajedian S, Motagh M, Samsonov S V. Spatiotemporal evolution of seismic slip of the 31 October 2013 Ruisui, Taiwan, earthquake[C]. IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium, 2018: 2248-2250.

[36] 张文蕾, 杨奕, 翁骋. 地震地下流体实时监测与地震预测研究[J]. 科技创新导报, 2018, 15(04): 22-23.

[37] 王广才, 沈照理. 地震地下水动态监测与地震预测[J]. 自然杂志, 2010, 32(02): 90-93.

[38] 王志敏. 地震地下水动态监测与地震预测研究[J]. 科技经济导刊, 2016, (03): 112-113.

[39] 王继军, 赵国泽, 詹艳等. 中国地震电磁现象的观测与研究[J]. 大地测量与地球动力学, 2005, (02): 11-21.

[40] Carducci L M, Alonso R, Fano W G. Data acquisition system for the study of earthquakes precursors by measuring magnetic field emissions[C]. 2017 XVII Workshop on Information Processing and Control (RPIC), 2017: 1-6.

[41] Masruri M F I, Priadi R, Nanda B M T F, et al. Analysis of Preseismic Event Using Seismo-electromagnetics Data[C]. 2018 2nd International Conference on Applied Electromagnetic Technology (AEMT), 2018: 1-7.

[42] 张梅, 本 党: 地震监测手段需要新技术[N], 2014-05-09.

[43] 杨奕, 张文蕾. 地震应急信息技术在互联网+时代的应用研究[J]. 中国高新区, 2018, (08): 229.

[44] 丁丹, 倪四道, 田晓峰等. 地震相关的声音现象研究进展[J]. 华南地震, 2010, 30(02): 46-53.

[45] 夏雅琴, 崔晓艳, 李均之等. 震前次声波异常信号的研究[J]. 北京工业大学学报, 2011, 37(03): 463-469.

[46] 朱星. 岩石破裂次声探测技术与信号特征研究[D]. 成都理工大学, 2014.

[47] 潘黎黎, 曾佐勋, 王杰. 芦山地震(M_S7.0)及玉树地震(M_S5.2)震前次声波异常信号分析[J]. 地学前缘, 2013, 20(06): 73-79.

[48] Lukovenkova O, Marapulets Y, Kim A, et al. A Complex Method for Automatic Detection of Geoacoustic Emission Pulses Preceding Earthquakes[C]. 2018 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon), 2018: 1-4.

[49] 钱书清, 任克新, 吕智. 伴随岩石破裂的 VLF，MF，HF 和 VHF 电磁辐射特性的实验研究[J]. 地震学报, 1996, (03): 69-74+136.

[50] 陈化然, 杨冬梅, 李琪等. 1980 年以来我国电磁辐射地震前兆信息的观测与研究[J]. 中国地震, 2008, (02): 180-186.

[51] 高曙德, 汤吉, 孙维怀. 盈江 5.8 级和缅甸 7.2 级地震前电磁异常[J]. 地球物理学报, 2013, 56(05): 1538-1548.

[52] 刘晨光, 王新安, 雍珊珊等. AETA 多分量地震监测系统的数据存储与安全系统[J]. 计算机技术与发展, 2018, 28(12): 7-12.

[53] 庞瑞涛, 雍珊珊, 王新安等. 地震监测系统的电磁信号的采集设计与实现[J]. 计算机技术与发展, 2018, 28(02): 27-30.

[54] 曾敬武, 雍珊珊, 郑文先等. 适用于大地震临震预测的地声传感单元[J]. 计算机技术与发展, 2015, 25(12): 133-137.

[55] 金秀如, 雍珊珊, 王新安等. 地震监测系统 AETA 的数据处理设计与实现[J]. 计算机技术与发展, 2018, 28(01): 45-50.

[56] 雍珊珊, 王新安, 庞瑞涛等. 多分量地震监测系统 AETA 的感应式磁传感器磁棒研制[J]. 北京大学学报(自然科学版), 2018, 54(03): 495-501.

[57] Du J, Li W, Guo J. Design of LabVIEW based general data acquisition system[C]. 2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), 2017: 1235-1239.

[58] Önder N M, Çakmak M C, Ünver Ö. FPGA based data acquisition and test system design for diagnostic testing[C]. 2017 IEEE AUTOTESTCON, 2017: 1-5.

[59] Lin H, Zhang Z, Guo Y, et al. A Configurable Data Acquisition System for Various Working Conditions[C]. 2018 25th International Conference on Mechatronics and Machine Vision in Practice (M2VIP), 2018: 1-5.

[60] Shao W, Guo H, Peng P, et al. Information Acquisition Structure of Internet of Things Based on Intelligent Gateway between ZigBee and Ethernet[C]. 2018 International Conference on Smart Grid and Electrical Automation (ICSGEA), 2018: 404-407.

[61] Pawar S, Shete V. Data Acquisition System Upgradation at Engine Test Cell[C]. 2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA), 2017: 1-6.

[62] Sarrab M, Alnaeli S M. Critical Aspects Pertaining Security of IoT Application Level Software Systems[C]. 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), 2018: 960-964.

[63] Gao P, Li Z, Li F, et al. Design of Distributed Three Component Seismic Data Acquisition System Based on LoRa Wireless Communication Technology[C]. 2018 37th Chinese Control Conference (CCC), 2018: 10285-10288.

[64] Kafadar O, Sertcelik I. A Computer-Aided Data Acquisition System for Multichannel Seismic Monitoring and Recording[J]. IEEE Sensors Journal, 2016, 16(18): 6866-6873.

[65] 黄经国, 李正媛, 陶冶等. 基于 SFTP 的 GNSS 数据采集软件的设计与实现[J]. 震灾防御技术, 2017，2017(2): 409-414.

[66] 朱倩钰. 单通道无缆地震数据采集系统低功耗研究与实现[D]. 吉林大学, 2017.

[67] 谢荣清. 高精度地震数据采集系统的嵌入式软件设计与实现[J]. 自动化与仪表, 2018, 33(06): 95-98.

[68] 汪超. 基于 GPRS 的低功耗远程数据采集终端软件设计[D]. 电子科技大学, 2017.

[69] 徐志国, 梁姗姗, 邹立晔等. SeisComP3 地震监测软件系统及其在海啸预警系统建设中的应用[J]. 科技导报, 2017, 35(07): 88-92.

[70] 李冬月, 杨刚, 千博. 物联网架构研究综述[J]. 计算机科学, 2018, 45(S2): 27-31.

[71] 张彬彬, 张军华. 地震数据低频信号保护与拓频方法研究[J]. 地球物理学进展: 1-8.

[72] Salomon D. Data Compression: The Complete Reference[M]. 2000.

[73] Schendel E R, Ye J, Shah N, et al. ISOBAR preconditioner for effective and high-throughput lossless data compression[C]. IEEE International Conference on Data Engineering, 2012.

[74] Cinque M, Cotroneo D, Pecchia A. Event Logs for the Analysis of Software Failures: A Rule-Based Approach[J]. IEEE Transactions on Software Engineering, 2013, 39(6): 806-821.

[75] Pecchia A, Russo S. Detection of Software Failures through Event Logs: An Experimental Study[C]. IEEE International Symposium on Software Reliability Engineering, 2013.

[76] 黄侠. “日志系统”在网络维护中的重要性[J]. 无线互联科技, 2012, (01): 44-46.

[77] 廖湘科, 李姗姗, 董威等. 大规模软件系统日志研究综述[J]. 软件学报, 2016, 27(8): 1934-1947.

[78] Yuan D, Park S, Zhou Y. Characterizing logging practices in open-source software[C]. Proceedings of the 34th International Conference on Software Engineering, 2012: 102-112.

[79] Ding Y, Park S, Peng H, et al. Be conservative: enhancing failure diagnosis with proactive logging[C]. Usenix Conference on Operating Systems Design & Implementation, 2012.

[80] 都军民, 戴宗妙. 软件可靠性定量指标的分配与验证[J]. 航空计算技术, 2002, 32(2): 20-23.