华南理工大学

South China University of Technology

硕士学位论文

面向车联网应用的信息发布系统的设计与实现

作 者 姓 名　张洁柯

学 科 专 业　信号与信息处理

指 导 教 师　傅予力教授

所 在 学 院　电子与信息学院

论文提交日期　2014 年 4 月

Design and Implementation of Information Distribution System Based on Internet of Vehicles

A Dissertation Submitted for the Degree of Msater

Candidate：Zhang Jieke

Supervisor：Prof. Fu Yuli

South China University of Technology

Guangzhou, China

分类号：TN915 **学校代号：10561**

学 号 ：201120108588

华南理工大学硕士学位论文

面向车联网应用的信息发布系统的设计与实现

作者姓名：张洁柯 指导教师姓名、职称： 傅予力 教授

申请学位级别：工学硕士 学科专业名称：信号与信息处理

研究方向：通信信号处理

论文提交日期：2014 年 4 月 25 日 论文答辩日期：2014 年 6 月 日

学位授予单位：华南理工大学 学位授予日期： 年 月 日

答辩委员会成员：

主席：_____

委员：_____

华南理工大学

学位论文原创性声明

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写的成果作品。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律后果由本人承担。

作者签名：日期：年月日

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属华南理工大学。学校有权保存并向国家有关部门或机构送交论文的复印件和电子版，允许学位论文被查阅（除在保密期内的保密论文外）；学校可以公布学位论文的全部或部分内容，可以允许采用影印、缩印或其它复制手段保存、汇编学位论文。本人电子文档的内容和纸质论文的内容相一致。

本学位论文属于：

□保密，在年解密后适用本授权书。

□不保密,同意在校园网上发布，供校内师生和与学校有共享协议的单位浏览；同意将本人学位论文提交中国学术期刊(光盘版)电子杂志社全文出版和编入CNKI《中国知识资源总库》，传播学位论文的全部或部分内容。

（请在以上相应方框内打“ √ ”）

作者签名：　　日期：

指导教师签名：　　日期

作者联系电话：　　电子邮箱：

联系地址(含邮编)：

摘要

车联网技术隶属物联网技术的范畴，主要应用于智能交通系统（ITS，Intelligent Transport System）。作为智能交通系统的典型运用，车联网技术综合使用了信号与信息处理技术、消息分发技术、遥感与识别技术、数据挖掘技术、网络接入与控制技术等，把道路和车辆的各种动、静态信息通过互联网连接起来，从而实现多个维度的系统大规模、大数据的信息融合，最终起到减少道路事故、优化交通资源配置的作用。车联网典型的应用服务主要分为交通道路安全服务、交通数据采集分发服务以及娱乐信息服务三大类。但是目前基于车联网的信息发布系统普遍存在通信质量差、传输速率低、系统并发访问差，业务单一等问题。智能交通领域上缺乏提供完善服务的车联网信息发布系统。

首先，本文研究了两种有助于提高车联网应用服务质量的关键技术，即WAVE(Wireless Access in Vehicular Environment)通信协议栈技术以及 LAMP（Linux+Apache+MySQL+Python）架构的网络服务平台架设技术，并简要的分析这两种关键技术的特性。其次，在针对车联网信息数量多、数据量大、通信时间短的特点，提出了三层架构的车联网信息发布系统的构建方案；针对车联网通信环境快速多变、通信质量差的特点，设计了以 WAVE 协议栈作为信息发布系统的底层通信协议的方案；针对车联网用户数量庞大，车联网网络中心并发访问需求高的特点，设计了基于 LAMP 架构的 Web 网络中心的方案。最后，再根据本文提出的系统设计方案，详细描述信息发布系统的网络中心、路侧单元 RSU（Road Side Unit）和车载单元 OBU(On Board Unit)三个层次中各个功能子模块的具体实现方法。

与传统信息发布系统相比，车联网信息发布系统摒弃了此前常用的 C/S 架构，使用本文提出的适合车联网环境的三层架构体系，以 WAVE 协议栈作为系统通信基础，采用高性能高并发量的 LAMP 构建 Web 网络中心。经测试实验证明，本文所设计的雏形系统，其点对点通信传输带宽约为 12 Mbits/s，单服务器可满足 200-500 个车载单元 OBU 同时在线访问。该车联网信息发布系统能满足面向车联网应用的服务需求，对构建新型的智能交通具有一定的实践意义。

关键词：智能交通系统；车联网；WAVE； LAMP 架构

Abstract

Internet of Vehicles (IOV) is one of the main branches of Internet of Things (IOT), which is generally used in the Intelligent Transportation System (ITS).In order to connect all the static and dynamic information of cars and roads via the Internet, signal and information processing technology, message distribution technology, sensing and recognition technology, data mining technology and network communication technology are synthetically applied to IOV. As a result, a large-scale, large-capacity and multiple- dimension data fusion of the system can be achieved and thus the number of road accidents reduces and the efficiency of traffic resource allocation is improved. Typical IOV services are mainly divided into three categories, including traffic safety services, traffic data collection and distribution services, and entertainment information services. However, there are still open challenging issues in IOV at present, such as unsatisfying communication quality, low data rate, conflicted concurrent access and unattactive business model. An efficient information distribution system is highly demanded as a fundemantal to solve the aforementioned issues.

WAVE (Wireless Access in Vehicular Environment) communication protocol stack technology and LAMP (Linux + Apache + MySQL + Python) architecture, which plays an important role in improving the service quality of IOV, are studied in this paper. The analysis of their characteristics is also presented. The design of system can be divided into three steps: 1) considering a huge quantity information and a short communication time, a three-tier design scheme for the IOV information distribution system is proposed; 2) regarding the IOV rapidly changing communications environment and the poor communication quality characteristics, the WAVE stack is used as a underlying communication protocol; 3) for the IOV huge number of Internet users and the high concurrent access, a LAMP-based Web network centers is proposed. Based on the proposed system design scheme, the implementation of IOV information distribution system's network center, road side unit RSU and vehicle unit OBU and their respective sub-module is investigated.

Compared with the traditional information systems, IOV information distribution system proposed in this paper discards the traditional C/S architecture and uses the three-tier system which is more suitable for the IOV environment. The WAVE communication protocol stack is used as a basis communication protocol in IOV information distribution system, while the high-performance and high-concurrency LAMP architecture is used as building Web network Center. The system is built and verified. The results show that the prototype system can achieve good performance, in which point- to-point communication transmission bandwidth is about 12

Mbit/s, the server meets 200-500 onboard unit OBU online access at the same time. The IOV information distribution system, as an integral part of ITS, has a practical significance to new intelligent transportation systems.

Keywords: Intelligent Transport System; Internet of Vehicles; Wireless Access on Vehicular Environments; LAMP Architecture

目录

第一章 绪论

1.1. 研究背景及意义

随着车联网技术研究的深入，IEEE 已经在 802.11 无线通信规范的基础上制订了专门用于车载通信的协议标准----WAVE[1]。WAVE 协议栈主要制定了车联网中车-车（V2V，Vehicle to Vehicle）和车-路（V2I，Vehicles to Infrastructure）的无线通信规范[2]。作为智能交通系统的典型运用，车联网技术综合运用了各种高新技术，如信号与信息处理技术、射频遥感识别技术、传感器监控技术、互联网与网络接入控制技术等，利用设置在道路两侧或者车辆内部的红外传感器、射频识别设备、视频及音频感应设备、GPS 系统等信源感知及传输设备，按照预先设定的协议，把道路以及道路上车辆的各种动、静态信息通过现有的网络互联互通起来[3][4][5]。从这个维度上讲，车联网就是通过互联网把现有的多层次系统大规模的融合起来，从而达到对每一条道路和道路上的车辆进行全时空的监测感应，以起到减少交通事故、提高交通资源配置作用的新型应用技术。它能够有效地解决目前城市普遍存在的道路监管能力低下、机动车辆增长过快、交通管理水平不足等问题。因此，研究车联网相关技术，开发车联网相关的核心产品，借此发展与其息息相关汽车类电子产业，推动相关领域的社会和经济变革，势必将产生不可估量的社会经济效应[6][7][8]。

为了确保在快速移动的车联网通信环境中，车载单元 OBU 之间，车载单元 OBU 与路侧单元 RSU 之间，可以进行稳定的高质量通信，美国电气和电子工程师协会（即 IEEE 协会）开始在无线传输的 802.11 技术[9][10]（一种为无线传输服务的通信协议）的基础上定制 ASTM -DSRC 标准，并将该新协议标准重新命名为 802.11p 协议，其主要涵盖物理层和数据链路层管理规范。车联网的通信基础 WAVE 协议栈就是在 802.11p 协议[11]的基础上，包含了对通信协议中的 MAC 层、网络层、传输层以及资源的安全管理的各种规范协议的集合。WAVE 车联网技术能提高车辆在高速移动中的数据传输能力，并且保证通信系统的稳定性、安全性和可靠性。基于 WAVE 协议栈的车联网系统可以作为新型的车辆导航辅助系统和车辆道路服务平台的核心，应用到未来的智能交通、智能公路等领域，拥有十分广泛的应用前景。目前，IEEE 已经相继推出了 IEEE802.11p 标准、1609.3 规范、1609.4 规范等相关车联网通信的规范标准[12][13]，并将确定应用于包括车辆自身安全、交通管理服务以及商业娱乐服务中。

伴随互联网技术的快速发展，特别是 Web2.0 的广泛运用，它导致了传统由网站主导

内容产生转变为由用户主导内容产生的新型交互式的互联网产品模式的诞生[14][15][16]。互联网也从一系列的初级网络站点转变为成熟的为用户提供动态网络应用的服务平台。因此，一种以 Web 2.0 标准为基础的，用于搭建动态网站的强大应用程序平台的开发架构模型便被设计出来，其中典型的是商用软件架构 Oracle 的 J2EE 和 Microsoft 的.NET。但紧随着开源学潮的逐渐兴起，开放源码的 LAMP(Linux + Apache + Mysql + Perl/PHP/Python)架构体系凭借着诸如高简易性、高模块化、高安全性、成本低廉、并发性能优异等的出色性能，其应用的普及范围不断扩大，已从诞生之初的小型服务应用系统的扩展到复杂、大型服务平台系统上，并开始与 J2EE 和.Net 等商业软件形成分庭抗礼之势[17]。从国内外网站的流量上来说，LAMP 架构的服务平台已经占有了 70%以上的互联网访问流量，LAMP 模型已逐步发展成最热门的 Web 服务开发基础架构[18][19]。

本文将基于车联网领域中缺乏完善的信息分发系统现状的前提下，针对车联网通信环境快速多变的特点，分析车联网信息发布系统的服务需求，包括交通道路安全服务需求、交通数据采集分发服务需求以及道路娱乐商业服务需求等方面。并在这基础上，利用 IEEE 最新的针对车联网提出的 WAVE 协议栈，设计出一套三层体系的基于 LAMP 架构的面向车联网应用的信息发布系统。从通信角度上，使用 WAVE 协议栈保障了车联网可以提供可靠的服务；从业务角度上，LAMP 架构的高性能以及高稳定特性，保障了车联网中大数据、大用户量的服务需求得以快速实时的实现。并且，网络中心 LAMP 模型的实现，使得系统拥有良好的可维护升级和可扩展的能力。本文所述的面向车联网应用的信息发布系统，其第一层网络中心将采用 LAMP 架构部署服务器，作为整个车联网信息发布系统的业务核心，其第二层路侧单元 RSU 和第三层 OBU 将采用软件客户端的形式实现。RSU 和 OBU 之间利用 WAVE 协议进行通信，同时通过 Web 客户端的形式与网络中心进行信息交互。

1.2. 国内外研究现状

1.2.1. 智能交通系统研究现状

智能交通系统（ITS）是一个面向道路交通服务的智能系统，它以物联网技术为基础，涵盖了车辆的监测控制、运营调度、信息通信服务、车辆信息数据挖掘等领域[20]。智能交通系统，其核心就是通过把现代通信技术融入到传统的道路交通系统中，实现将公路、铁路、车辆、乘客以及路边的设施完全联动，并通过公路、铁路、车辆、乘客以及路边的设施之间实施准确，频繁的信息交互，达到最大限度的优化公共交通资源，为公共交

通系统的使用者和监控者提供智能化的服务的系统。目前，国际上主要由欧洲的 ERITCO、美国的 ITS America 以及日本的 VERTIS 等机构在积极的促进智能交通系统向民用化商用化演变[21]。

智能交通系统的通信方式主要分成两种，即“车-路”通信（V2I, Vehicles to Infrastructure）与“车-车”通信（V2V,Vehicle to Vehicle）。国际上对 ITS 的研究主要集中在“车-路”通信（V2I）中[22]。如日本的 VICS 系统和美国的 IVHS 系统，都是利用道路与车辆之间的信息连接，实现对交通路况的管理配置功能[23]。WAVE 协议栈作为智能交通系统中的核心通信协议簇，其核心框架结构及具体协议正逐步被标准化。符合 WAVE 协议栈的原型 RSU 和 OBU WAVE 模拟测试设备也已经逐步开发出来，其中最典型的有美国的 Sirit 公司（美国主要的 RFID 技术提供商）和日本的 RSB 公司（瑞萨半导体）已经开发出正在测试阶段的原型 WAVE 设备。台湾的 Unex 公司也在最近提出了整套符合 WAVE 协议栈标准的 WAVE 解决方案。总而言之，伴随着最新的车联网通信规范（802.11p 协议）、自组网技术、网络接入技术、传感器监测技术等技术的不断更新变革，国际上关于“车-路-人”（V2I2P）的通信规范正朝着短程化、网络化、实时化、专用化、信息化的方向发展。除此之外，国外的汽车制造商也在车联网通信方面做出了卓越的贡献，如：通用汽车、西门子、宝马等。集成车载单元 OBU 的设备 Onstar 已经在美国上市。

相比于国外迅速发展的车联网技术，国内相关领域的研究及智能交通系统的产业化相对比较落后。国内尚未有专门的机构或者公司，比如类似欧洲的 ERITCO 等专门从事国际上车联网技术相关协议的标准定义工作的组织。但伴随着国内经济的日益发展，人们物质生活水平的提高，加上国内城镇化的推进，智能交通系统的研究已经逐步被列入了重点。在近十年来，我国已经把智能交通领域的研究列入了“十五”、“十一五”、“十二五”的国家科技攻关计划中。各种国家研究所、高校甚至是企业也开始投入大量的物力和人力到智能交通系统的领域来，并也取得相应的成就，如公路的电子 ETC 收费系统、城市交通的智能指挥控制系统等。总之，我国虽在智能交通系统的研究中然相对落后，但发展迅速，且后劲十足。

1.2.2. 车联网的信息分发系统研究现状

伴随着智能交通系统的高速发展，为高速行驶的移动车载终端提供实时可靠的信息服务是当今 ITS 中一个十分重要的研究热点。但是以目前的方法和手段收集与发布信息的成本很高，并且针对不同需求开展信息服务的效果与可用性受到特定技术的限制。而

能为智能交通服务、功能完善的可用于车联网的信息发布系统乏善可陈[24]。

现有的车联网信息发布系统，按照通信方式，主要分成以下四种：

- 基于射频识别（RFID）技术的信息发布系统

这类系统的典型代表就是电子 ETC 收费系统（Electronic Toll Collection）。ETC 系统是目前最流行并且逐渐普及的一种用于高速公路等收费路段的电子自助收费系统。这类系统利用射频识别技术，通过非触碰通信的方式进行身份鉴权和数据传输服务。此类系统一般依赖于专用的收费通道进行信息分发，不能完整的部署在所有的道路上，服务覆盖范围较小，并且信息交互的种类有限，一般只能交互身份信息、收费信息等特定的消息类型，无法满足大数据时代高信息容量传输服务要求。并且此系统中的移动设备只能被动的接收路侧单元的信息并作出处理，无法进行信息交换，也即移动车载设备无法向路侧单元主动请求信息。因此，其消息发布是单向传递的，无法满足现今智能交通系统所需要的可靠稳定的交互式信息发布[25]。

- 基于移动基站提供的 2G/3G 通信方式的信息发布系统

这种功能系统的典型代表，是使用移动通信运营商的 2G/3G 网络的信息分发系统。但该系统对运营商基站的依赖性较强，在没有基站信号的偏远道路或者基站信号盲点处，信息发布系统作用将大打折扣。加之，对于高速移动的车载终端传输速率较低，实时性较差，无法为车辆提供实时可靠的消息服务。更重要的是，使用基于运营商的 2G/3G 信号信息发布系统的通信成本较高，商用性能较差[26]。

- 基于 ZigBee、蓝牙等低速率传输设备的信息发布系统

使用 ZigBee、蓝牙网络作为交通信息发布系统，其最大的特点就是低功耗、低成本。但低功率带来的代价就是传输速率过低和服务覆盖范围过小。使用 ZigBee 网络进行信息分发，传输的数据量较低，只能勉强满足基本的交通信息收集和发布功能，无法满足现今大数据时代高信息容量传输服务要求。对于数据量要求较大的视频图像广告等业务无法满足[27]。

- 基于 Wifi 通信的客户端-服务器模式的信息发布系统

这类系统使用无线 Wifi 作为网络覆盖和通信的媒介，但其终端之间通信依赖于网络接入点，在没有网络接入点覆盖区域，网络通信将会中断，并且这类系统仍然使用普通的 802.11a、802.11b、802.11g、802.11n 等通信协议，身份验证和鉴权耗时复杂，通信效率较低。其次，该系统中所有车载设备的网络接入服务均需要依靠在路侧部署的网络接

入点（AP）。但该接入设备一般只起到网关路由器的作用，该系统所发布和存储的信息均需要经过网络中心，网络接入点没有任何信息处理过滤的能力，网络中心服务负荷巨大。因此，这类系统信息冗余较多，效率较低，不利于在高速移动的车载环境中进行信息分发[28]。

从上面四类系统的描述中，可看出，在高速移动的车载环境中，面向车联网业务的信息发布系统乏善可陈。并且每个系统均有其严重的局限性，如何使行驶在公路上通信环境快速变化的车辆有效并且稳定的获得网络资源，与周围通信系统进行稳定的消息传递是智能交通系统中需要迫切解决的问题。

1.3. 系统功能

从前一小节，可以看出，在智能交通领域中，能使在快速变化的通信环境中的车辆有效并且稳定的获得网络资源的车联网的信息发布系统少之又少。

本文主要研究重点是在基于车联网缺乏完善的信息系统现状的前提下，利用 WAVE 协议栈和 LAMP 动态网络应用服务平台架构的技术，设计出一种新型的面向车联网应用服务的信息发布系统，以满足智能交通系统中交通道路安全服务、交通数据采集分发服务以及道路娱乐商业服务等服务需求。

本信息发布系统研究内容及主要核心功能包括以下几点：

1. 根据车联网信息发布特点，设计三层架构的信息发布系统。

针对车联网信息发布系统，系统数据信息的分发对象主要为移动中的车载单元 OBU。车载单元 OBU 的消息一般具有数据量大、有效通信时间短等特点，而老式的服务器客户端模型（C/S 架构）的信息发布系统并不能十分切合车联网通信需求。因此，本文提出了新型的三层架构的面向车联网运用的信息发布系统。

2. 根据车联网通信特点，实现系统的底层 WAVE 通信功能。

车联网信息发布系统通信环境快速多变，老式的无线 802.11 协议拥有复杂繁琐的身份验证机制和鉴权机制，车辆往往尚未身份验证完毕便离开路侧单元服务区。基于 WAVE 协议栈的通信方式，在功能上可以很好的解决该问题，本文将在后面详细描述该协议栈。

3. 根据车联网信息数据量、访问并发量巨大的特点，设计 LAMP 架构的服务器。

由于车联网中 OBU 的数量庞大，造成对网络中心的并发访问能力要求很高。因此，作为网络中心需要性能强大、并发能力强、系统稳定的服务器。因此，本文将详细叙述

使用 LAMP 架构部署动态网络服务平台实现网络中心功能的原因及其相应的特点。

4. 根据车联网 RSU、OBU 服务的特点，设计实现 RSU、OBU 客户端软件系统。

作为车联网信息发布系统最低层的数量庞大的路侧单元 RSU 和车载单元 OBU，它们一方面需要具有两两之间进行通信的能力，另一方面它们的服务一般都具有基于位置的特征，并且其位置是不断变化的。因此，本文将在后面根据 RSU 和 OBU 的特点设计 RSU 和 OBU 的软件客户端系统。

1.4. 本文研究目标及组织架构

本论文研究目标是将 WAVE 通信技术和 LAMP 动态网络应用服务平台架构技术应用到车联网通信系统中，从而根据本文提出的三层结构部署系统的设计方案搭建出能够应付快速变化的车联网通信服务环境的车联网信息发布系统。

本论文各章节的内容安排如下：

- 第一章 绪论：

本章简要介绍本论文研究背景，并针对车联网技术国内外发展现状以及目前使用的作车联网信息发布系统做了一定的研究对比。

- 第二章 信息发布系统技术研究：

本章根据车联网研究现状，阐述了 WAVE 通信技术以及 LAMP 技术体系。第一节详细的介绍了 WAVE 协议栈的结构和 WAVE 通信特点，第二节则概述了 LAMP 架构体系，LAMP 架构模型特性以及 LAMP 每一层的技术特征。

- 第三章 车联网信息发布系统的整体设计：

本章分析面向车联网应用的信息发布系统的服务需求，并根据该服务需求，提出了三层架构的车联网设计方案，并利用第二章研究的 WAVE 协议栈、LAMP 架构实现 RSU、OBU 底层通信功能和网络中心服务器。

- 第四章 网络中心的设计与实现：

本章根据第三章的设计方案，描述了网络中心的设计与实现。其中重点描述了网络中心 Web 服务器的部署实现概要，网络中心数据库的设计（包括数据库表，数据库接口设计），Python CGI 网关的设计。

- 第五章 RSU 与 OBU 的设计与实现：

本章根据第三章的设计方案，分析路侧单元 RSU 和车载单元 OBU 的功能需求，并根据其功能特点，介绍了其软硬件平台，以及软件系统三大模块（信息分发功能子模块、

Web 客户端子模块和系统配置更新子模块）的详细实现方案。

- 第六章 系统测试：

本章从系统的功能和性能出发，分别对所设计的雏形系统进行测试。并根据测试结果与其他同类信息发布系统进行比较，得出本雏形系统的性能优势。

- 总结与展望：

本章对本论文进行详细总结，论述论文所做的研究和工作，以及系统后续需要进一步研究和解决的问题。

第二章 信息发布系统技术研究

2.1. 本章内容概要

本章将介绍/研究信息发布系统的关键技术，主要包括车联网通信标准 WAVE 协议栈以及 LAMP 服务器平台。其中 WAVE 协议栈将基于硕士阶段对 WAVE 框架的自主研究成果进行介绍，LAMP 架构技术则是根据现有的 LAMP 部署技术进行拓展探讨。

2.2. 车联网通信技术

随着车联网通信技术研究的深入，IEEE 已经在旧式的无线通信协议（802.11）的基础上制订了车载通信的专属规范 WAVE，专门用于车联网中 V2V 以及 V2I 的无线通信应用[29][30]。WAVE 是指能够实现的宽带车车之间、车路之间无线传输通信，完成实时精确和可靠的图像、语音和数据双向传输的新兴技术。WAVE 作为下一代智能交通系统(ITS)的发展方向，它承担着车联网络技术发展的核心构件的作用。作为为车载单元与路侧单元提供网路通信服务的 WAVE 技术，它使得交管中心能收集车辆的行车数据的同时，也使得车辆能拉取车联网中的各种消息资源。WAVE 技术能提供高速的数据传输，并在此基础上同时保证系统的可靠性与稳定性，以及安全性[31]。就目前的研究状况而言，基于 WAVE 的运用范围非常广。在本节中，将重点对适用于车联网络通信的 WAVE 协议栈标准进行研究和分析。

2.2.1. WAVE 技术简介

WAVE 作为下一代智能交通系统的关键技术，其通信特点有以下几点：

1）基于 WAVE 的移动设备间可以进行信息交互，通信理论距离大概 1000 米内，实际运用的有效距离约为 300 米 [32][33]；

2）高传输速率：一般可达到 3-27Mbps；

3）与现有的互联网兼容：其网络层采用了 IPv6 协议；信道的具体分布见图 2-1。

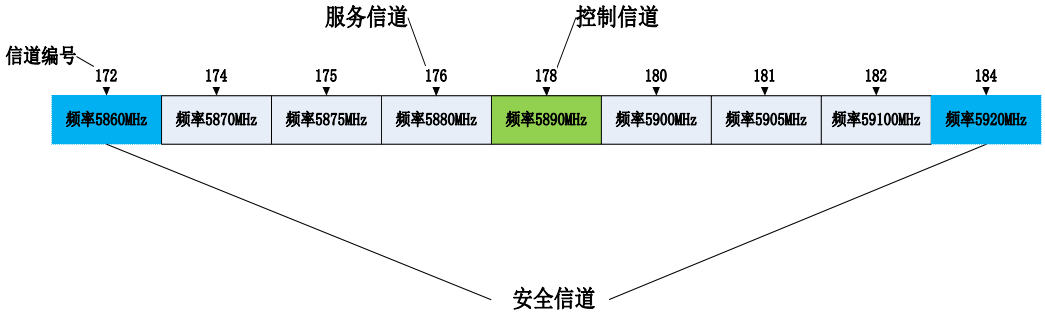

图 2-1 WAVE 信道分布及应用

以 WAVE 为代表的下一代车路专用高速通信技术为各种 ITS 应用提供了有力支撑，其典型的服务包括：

1）交通安全的服务，如典型的车辆免碰撞提示、行车道路通告、行车天气实时通知等；

2）公共服务相关的服务，如典型的道路交通监控、智能导航地图服务、区域道路状况采集，停车车位搜索等；

3）商业娱乐性质的服务，包括互联网接入、商业广告服务、酒店住宿餐饮等商业服务的搜索。

2.2.2. WAVE 协议栈研究与分析

2.2.2.1. 协议栈分析

为了确保在快速移动的车联网通信环境中，车载单元 OBU 之间、车载单元 OBU 与路侧单元 RSU 之间，可以进行稳定的高质量通信，IEEE 协会开始在旧式无线传输的 802.11 技术（一种为无线传输服务的通信协议）的基础上定制 ASTM DSRC 标准[34]。并对该新协议标准重新命名规定为 802.11p 协议，其主要涵盖物理层和数据链路层管理规范。车联网的通信基础 WAVE 协议栈就是在 802.11p 协议的基础上，包含了对通信协议中的数据链路层，网络层，传输层以及资源的安全管理的各种规范协议的集合。WAVE 车联网技术能提高车辆在高速移动中的数据传输能力，并且能保证通信系统的稳定性。基于 WAVE 协议栈的通信设备主要分成两类[35]：

- 路侧单元 RSU

RSU 承担着网络控制接入中心的作用，即 AP（Access Point）的角色。它一般通过有线或者无线的形式与网络中心进行互联通信，通常设置在道路两旁。

- 车载单元 OBU

OBU 一般为车载设备单元，它通过无线通信的模式与 RSU 或者其他 OBU 进行互联互通。

RSU 和 OBU 有两种基本的通信方式，在有 RSU 的情况下，RSU 可以和周围覆盖范围的多个 OBU 以基础通信服务集 WBSS（WAVE Basic Service Set）的形式提供服务。当周围没有 RSU 的情况下，OBU 之间通过自组网的方式，构建 WIBSS (WAVE Independent Basic Service Set)，也即独立结构式的基础服务集，从而实现车-车通信并共

享信息。RSU 和 OBU 均是利用 WAVE 协议栈实现在快速移动的通信环境中完成数据信息的传递[36]。

WAVE 协议栈从其诞生历史可以知道，其底层的协议（PHY 层和 MAC 子层）便是基于 IEEE 802.11 协议的。它借鉴了 802.11 协议簇中 11a，11b，11g 等通信规范的特点。因而也可以说 WAVE（或者 WAVE 模式）就是定义在 IEEE 802.11 协议之上的一种新的车载通信架构，因而它同时具备了与旧式无线通信设备兼容的特性。

从之前叙述的 WAVE 设备可以看出，新的 WAVE 协议栈和当前的无线网络结构类似，它也有自己的 BSS 和 IBSS 结构，即 WBSS 和 WIBSS。对于 WAVE 设备而言，其主要控制中心节点 AP（Access Point）为路侧单元 RSU ，移动工作站（STA）为车载单元 OBU。

WAVE 协议栈借鉴了传统七层网络结构，并且将信息传输区分为数据层和管理层进行管理。WAVE 的数据层担负着数据传输的责任，WAVE 的管理层则担负着系统传输行为的配置管理职责。从功能上 WAVE 协议栈包含了以下内容：

1) 802.11p 服务：包含了 802.11 数据链路层和物理层，针对快速移动通信环境的特点进行了相应的优化，其频段为 5.8G Hz，频段较为干净，调制模式则为 OFDM[37]；

2)多信道协作功能服务：它隶属于 WAVE 协议栈的数据链路层。WAVE 协议栈规定了 WAVE 设备支持多物理信道共同工作，即包括控制 CCH 信道和服务 SCH 信道。它们在系统时钟的控制下根据车载环境协同工作或自主切换，从而实现增强 WAVE 数据链路层传输质量的功能。

3)信息传输服务：WAVE 标准规定了本层有 IPV6 协议和 WSMP 协议两个标准可选。为了提高数据传输的效率，WAVE 将网络层和传输层合二为一，也即路由功能以及数据传输功能均由此层负责。并且为了增加 WAVE 协议的扩展性能，使其能与具体应用相切合，WAVE 并没有具体规定使用的路由协议。此外，由于 WAVE 协议栈规定可以实现 IP 协议，因而基于 WAVE 的网络可以好现有网络无缝融合。与此同时，WAVE 还设计了 WSMP 协议，它简化了传统网络协议的流程，大大的优化了通信效率。

4) 其他：WAVE 协议栈为了保证车辆通信中的私有信息的安全，在上层还规定了如 Privacy 等的各类安全算法。但由于车联网技术还并不成熟，WAVE 并没有实现一个成熟统一的的安全算法方案。该部分方案，IEEE 组织目前仍在协商中，有望在不久的将来达成一致。

2.2.2.2. 协议栈结构

WAVE 协议栈主要包括 IEEE 802.11p 以及上层的 IEEE P1609. 1、P1609. 2、P 1609. 3 以及 P 1609. 4 规范。其中 802.11p 为其底层的协议，它主要制定了物理层和介质访问控制层的相关规范。而 IEEE P1609 的协议簇则是在 802.11p 底层规范的基础上定制的，主要涵盖了链路层、网络层、传输层以及相应层中的资源配置管理标准以及安全规范。如下图 2-2 所示：

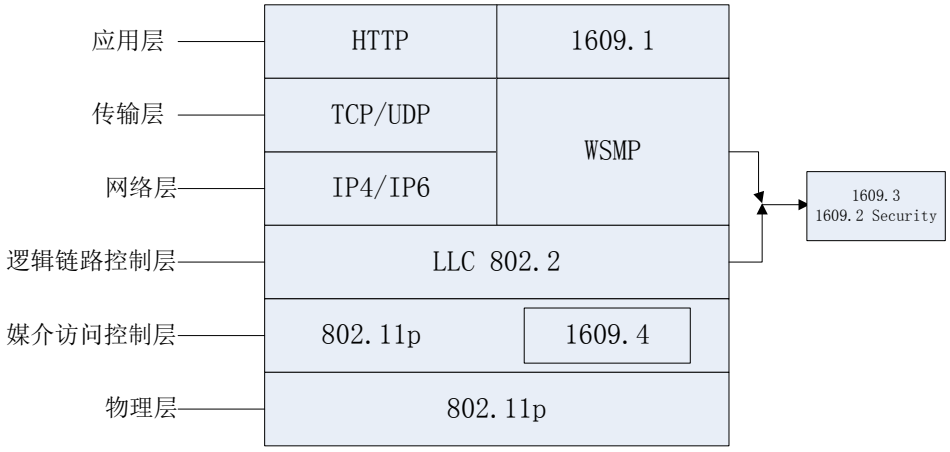

图 2-2 WAVE 协议栈结构

802.11p 协议的 MAC 层和 PHY 层使用的是 5.8GHz 的频带。这样，就保证了 WAVE 数据的可靠性，并且大幅度的优化 MAC 层，同时还能简化握手流程，提高了通信的效率。与此同时，它还同步使用 1609.4 协议，这样一来使得 WAVE 支持多信道操作，包括服务 SCH 信道和控制 CCH 信道 CCH 的操作、信道跳转和路由控制以及多信道选取。同时，它作为数据链路层的一部分，能为数据帧的传输提供高效的 QoS 保证[38][39]。

WAVE 协议栈的网络层由 1609.3 规范来负责实现，划分为数据层及管理层，数据层则主要是通信的协议以及发送数据的硬件结构，它由 LLC、IPv6 的 UDP 和 TCP 以及 WSM 和它对应的 WSMP 协议组成；管理层主要是系统的配置和维护的功能，它包括了应用程序的注册、WBSS 的管理、信道使用情况的监督、IPv6 的配置以及 RCPI 的监测和 MIB 的维护[40]。

WAVE 协议栈的安全服务保证是由 1609.2 规范负责实现。在本规范中，它定义了安全报文格式，以及如何处理这些安全报文的方法，并在此基础上提供对 WAVE 的应用程式及管理消息的安全加密功能[41]。

WAVE 协议栈的上层是由 1609.1 规范来实现，是一个可选层，它规定了如何将 OBU 的系统资源规范化地存储。利用该规范，外界应用便可以根据规范约束的接口来访问

OBU 资源，大大的提高了 WAVE 设备应用的兼容性能[42]。

2.2.3. WAVE 信息传输服务

WAVE 的信息传输服务主要分为两种，维护系统安全性稳定性的管理数据传输服务以及传输实际数据的数据传输服务。

- WAVE 管理信息传输服务

WAVE 协议栈通过层与层之间的传递信息管理数据进行系统的管理服务，主要包括：管理数据的传输功能、信道的接入和分配服务功能、WAVE 服务广播监听功能、IPv6 的定制配置功能以及管理信息库 MIB 的维护功能。

WAVE 管理信息传输服务的职责是由 WAVE 管理实体 WME 履行的。WME 将根据上层应用服务的不同请求，配置系统的服务参数以及提供跨层 API，以满足上层应用接入服务 SCH 信道的要求。同时，WAVE 管理实体 WME 还会定期的广播 WMA 数据包，告知其他 WAVE 设备自己的管理信息。例如，要访问系统特定的服务 SCH 信道，WME 通过收发器向数据链路层的管理模块发送访问请求，并在信道规定的时钟内，调整通信设备工作到指定的频段中。

在车联网通信中，WAVE 设备的路侧单元和车载单元的职责是固定的。作为服务提供者的 RSU（或者在自组网中的某一 OBU）将发送管理信息 WSA 帧，以此向其他设备声明其在某一个或者多个服务 SCH 信道上具备数据交换能力。其他的 OBU 所承担的职责则是监听和接收广播中的 WSA 信息报文，并根据其应用层服务的要求配置到某服务信道 SCH 上进行信息交互。按照 WAVE 协议栈规定，对任一特定的 WAVE 设备，它可以根据自身服务需求选择成为服务提供者或者服务使用者，甚至两者功能兼有。

根据上述描述可知，所有的 WAVE 设备均是由 WAVE 管理实体配置管理信道分配的。其基本的接入信道服务的例子可以由下图 2-3 清晰看出。

从图 2-3 可以看出，信息传输服务根据 WAVE 的接入分配信道的种类分为以下四种：只运行在服务信道或控制信道；来回运行不同信道，即在服务信道系统时钟内使用服务信道 S，在控制信道系统时钟内使用控制信道 CCH；优先使用服务信道，即在服务信道或者控制信道系统时钟内，可以立即对服务信道进行接入，并可适当增加服务信道系统时钟时间；在服务信道或控制信道系统时钟内，对服务信道进行接入控制（不能像上一种一样接收到服务请求后立即控制服务信道）[43]。

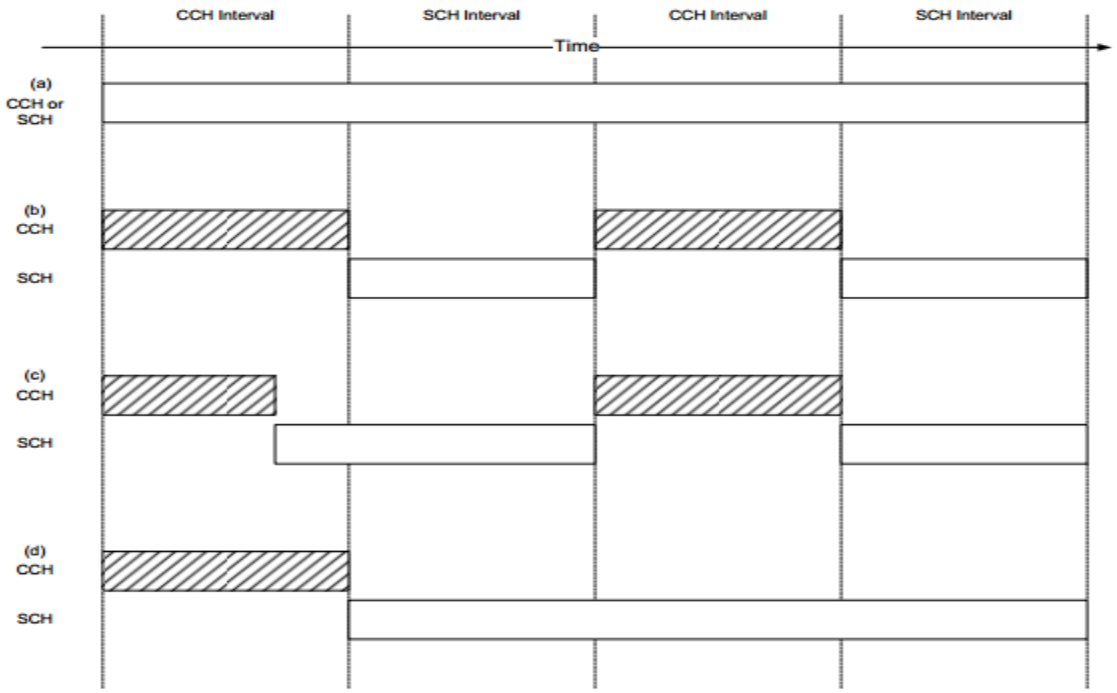

图 2-3 信道时钟分配

- WAVE 数据传输服务

从图 2-4 WAVE 协议栈结构图看以看出，WAVE 数据传输从底层往上区分可以划分为三大功能部分：由数据链路层和物理层共同组成的物理信道接收和发送数据功能部分；由 IPV6 以及上层的 TCP/UDP 协议组成的兼容现有网络的数据传输功能部分；以及 WAVE 协议栈设计的替代网络层和传输层的 WSMP 数据包传输协议功能部分。

从上述三部分可以看出，支持 IPv6 和 TCP/UDP 协议的特性使得 WAVE 协议栈能无缝兼容现有的无线传输网络。标准中新增加的 WSMP 传输协议，融合了网络层和传输层，并加强了专用的数据传输接口性能，以应对车载通信中的高速率低延时特性。

WAVE 协议栈的一个创新之处就是使用了替代传输层和网络层协议的 WSMP 传输协议。基于 WSMP 协议的数据传输服务，传输的每一个数据包称为 WSM 数据包，并由服务的应用层根据需求设置其传输信道、发送功率和传输速率。由于 WAVE 协议栈同时支持 IPv6 网络传输服务和 WSMP 传输服务，因而符合 WAVE 通信标准的移动设备，可以根据自身服务的需求，选择兼容 IPv6 网络传输服务或者兼容 WSMP 传输服务，甚至两种协议兼之。

WAVE 协议栈在设计之初，便是为了尽可能减少在高速移动的车载通信环境中，WAVE 设备传输数据的延时时间，保证数据传输的通信质量，以满足各种基于车联网的应用服务需求。故使用单层的 WAMP 协议取代传统的网络层和传输层服务协议，以简

化数据传输时的处理时间。

2.2.4. 车联网通信技术对比

目前，现有的车联网信息发布系统，主要有四种类型，它们各自使用的车联网通信技术各不相同，分别是基于射频识别（RFID）技术、基于移动基站提供的 2G/3G 通信方式、基于 ZigBee 等低速率传输设备以及基于旧式的 802.11 的 WiFi 方式。

使用射频识别技术的车联网应用系统，其信息的传递一般是单向流动，无法进行消息交互；使用 ZigBee、蓝牙等低速率低功率的车联网应用系统，其速率低，服务覆盖范围狭小，无法满足现今车联网高信息容量的服务要求。使用运营商基站提供的 2G/3G 通信方式的车联网应用系统，对运营商基站的依赖性较强，在没有基站信号的偏远道路或者基站信号盲点处，信息发布系统作用将大大折扣。并且，该类通信系统通信成本较高，无法大规模应用。使用 WiFi 通信方式的客户端-服务器模式的信息发布系统，其使用传统的 802.11 的无线通信方式，身份验证和鉴权繁琐，无法在高速移动的车联网中进行可靠的传输服务。

从上述的分析可以看出，在高速移动的车载环境中，面向车联网业务的信息发布系统乏善可陈。并且每个系统均有其严重的局限性，如何使行驶在公路上通信环境快速变化的车辆有效并且稳定的获得网络资源，并与周围网络进行消息传递是智能交通系统中需要迫切解决的瓶颈[44]。

本文研究的以 WAVE 协议栈为基础的车联网通信技术，能够很好的解决上述的问题。

众所周知，旧式的 802.11 无线通信规范中，对新用户加入网络均会使用比较繁琐的身份验证和鉴权机制。这种繁琐的身份验证和鉴权机制将会占用用户与接入点大量的通信时间，其可能的后果是，用户尚未身份验证完毕便离开原来接入点的服务覆盖范围。在某些情况下，它甚至会让服务用户陷入了无限制的身份验证死循环中，大大浪费了宝贵的通信资源。WAVE 协议栈的 802.11p 协议就是基于这种情况下，对旧式的 802.11 身份验证和鉴权机制做了修正，简化了身份验证流程，提高了通信效率。

WAVE 通信标准中设定了两种用户角色，即服务提供者（Provider）和用户（User）。当两台支持 WAVE 的设备建立通信连接时，将由通信发起的一方承担服务提供者的职责，另一方承担用户的职责。首先，Provider 开启广播 WAVE 服务信息，宣告一个 WAVE 基础服务集（WBSS）的存在。WAVE 服务信息中包含了信道使用信息、应用程序支持类型以及各服务的配置参数等信息。当用户（User）进入服务提供者（Provider）提供 WAVE 基础服务集的通信覆盖区域后，收到服务提供者广播的服务配置帧。此时，

用户将利用预先保存的密钥解密该服务配置帧，以获取传输的管理信息（主要为获得基本集标识符 BSSID）。接着用户根据 BSSID 判断是否和自身预设的 BSSID 匹配。如果 BSSID 符合，则用户被允许加入该 WAVE 基础服务集中，并根据服务配置帧中的信息配置通信参数（如使用的频段等信息）。一旦服务提供者和用户之间建立起了一条通信信道，服务提供者和用户就转为对等服务，它们之间开始平等交换数据信息。当 WAVE 基础服务集有用户需要结束通信时，由该用户向 WAVE 相关的管理实体传输结束通信的消息，并离开该服务提供者的服务区；在某些特殊情况下，当用户监测到已经离开当前的 WAVE 基础服务集时，用户同样会自下往上的异步传输结束通信的消息命令，进而结束当前的通信。

目前，IEEE 对 802.11p 规范并没有最终的确定，根据最新的草案，WAVE 协议栈对两个 WAVE 设备建立通信的方式在原来的基础上再次优化了。它规定，WAVE 用户并不需要预设属于任何的 WAVE 基础服务集。相反，任何 WAVE 设备只需要在传输的数据链路帧的头上加入加入身份鉴权标识符 BSSID 即可。在信道设置相同的情况下，任何两个 WAVE 设备只要拥有相同的身份鉴权标识符 BSSID 便可以进行免验证的通信。在这样的方案中，在同一信道通信的车辆只需要拥有同样的身份鉴权标识符 BSSID，它们就可以完全避免了额外的开销便能直接通信。这一改进与上一版本相比，节约了服务提供者向用户广播 WAVE BSS 等一系列流程，更为简洁，使得 WAVE 协议栈相比旧式的 802.11 无线通信协议更适合于快速移动的车载通信环境。

因此，本文设计的面向车联网应用的信息发布系统，由于其底层设计使用了 WAVE 协议栈，使得本系统能更适合车联网快速移动的通信环境，为上层各种基于车联网通信的应用服务提供高速率可靠稳定的通信质量保证。

2.3. LAMP 架构技术体系

2.3.1. LAMP 架构概述

目前，国际上最流行的开源 Web 架构当属于 LAMP（Linux+Apache+MySQL+Python/PHP/Perl）架构[45][46][47]。LAMP 架构一般由类 Linux 操作系统、Apache Web 服务器、MySQL 管理数据库以及 Python 等脚本语言四部分组成。LAMP 架构包含的这四部分结构，均是开源的产品，是国际上目前比较成熟的架构，如新浪，搜狐，百度，腾讯等国内知名企业均使用这个架构。和传统的其他著名的 Java/J2EE 或者 Microsoft 的.Net 架构相比较，LAMP 模型是轻量级架构，其 Web 资源相对丰富，具有高快发，高性能以及跨

平台等优势。因此，LAMP 架构成为了大多数企业搭建 Web 服务器的首选，典型的 LAMP 网站如下表 2-1 所示：

表 2-1 经典 LAMP 架构网站

	L: System	A:WebServer	M: Storage	P: Script
Yahoo	FreeBSD+Linux	Apache	MySQl	PHP
Facebook	FreeBSD	Apache	MySQl+Memcached	PHP
Wikimedia	Linux	Apache+Lighttpd	MySQl+Memcached	PHP
Fickr	RedHat Linux	Apache	MySQl+Memcached	PHP+Perl
Sina	FreeBSD+Solaris	Apache+Nginx	MySQl+Memcached	PHP
Audiogalaxy	Linux	Apache	MySQl	PHP
Firendster	Linux	Apache	MySQL	PHP+Perl
YouTube	Suse Linux	Apache+Lighttpd	MySQl	Python
Mixi.jp	Linux	Apache	MySQl+Memcached	Perl
TypePad	Linux	Apache	MySQl+Memcached	Perl

2.3.2. LAMP 架构模型特性

紧随着互联网技术的急速发展，基于 HTTP 协议的的 WEB 结构在互联网技术中起着举足轻重的作用。目前，市场上的 WEB 架构的开发模型主要有三种，即 ASP.NET 架构（Windows Server + IIS + SQL Server + ASP 组合），J2EE 架构(UNIX + Tomcat + Orecle + JSP 组合)以及本文提到的 LAMP 架构（Linux + Apache + MySQL + PYTHON 的组合）开发模型。

三种架构模型的性能对比，如下表 2-2 所示：

表 2-3 Web 架构性能比较

性能比较	LAMP	J2EE	ASP.NET
运行速度	较快	快	快
开发速度	快	慢	中等
扩展性能	好	好	较差
安全性能	好	好	较差
建设成本	较低	高	中等
运行平台	Linux/UINX/Windows	绝大多数平台均可	Windows 平台

从上表看可以清晰看出，与 ASP.NET 架构，J2EE 架构相对比，LAMP 架构模型一般具有以下的这些特性：

- 灵活的开发特性

作为目前商业软件的两大阵营 ASP.NET 和 J2EE，都是由单一公司控制，由顶层控制的。系统开发人员可以控制的范围非常有限，例如，J2EE 是源于 Java 程序的框架，.NET 框架则是 ASP.NET 开发的唯一选择， Rails 则必须基于 Ruby 框架。但是，LAMP 架构的开发模型，系统开发人员可以根据自身系统的需要选择不同的 Web 框架。此外，开发者因为可以获取全部的源代码。从而可以根据自身需要，在 LAMP 模型中增加或者删除特殊的模块（如身份验证模块）。因此，LAMP 架构具有极其灵活的开发特性。

- 高模块化特性

LAMP 模型的灵活性和稳定性是兼容的，除了拥有优异的灵活特性外，还一向以稳定性而著称。LAMP 模型的各个开源软件，均能稳定的向后兼容。即便是系统更换了或者升级了新版本，开发人员一般也不需要重写代码。LAMP 模型的稳定性源于其具有高度模块化的特点。LAMP 架构的模块化是松散耦合的，并且模块间是可组合的并且自治，它依赖系统间的配置文件组合而成，因此 LAMP 模型的维护成本也比较低廉。

- 开发成本低，系统可扩展性能优异

面向车联网应用的信息发布系统，在系统部署阶段，开发部署成本肯定是最首要因素。与 ASP.NET 和 J2EE 等商业平台不同，LAMP 架构模型中，系统从高往下架设的所有产品均是开源免费的。并且，LAMP 对运行的硬件条件要求也更为低廉。在相同条件下，对于 Web 服务的托管领域，相比 ASP.NET 架构以及 J2EE 架构的托管服务，LAMP 模型的要低很多。与此同时，由于 LAMP 模型的开源特性，开发者可以根据自身的需求，对 LAMP 模型做特殊的定制，使得 LAMP 模型具有优良的可扩展特性。

- 安全性能高

作为目前最流行的 Web 框架开发模型，LAMP 架构的部署量是巨大的，国内许多大的门户网站均使用 LAMP 模型架构服务器。但由于 LAMP 模型的开源特性， LAMP 模型的安全问题也在不断地被发现和被优化中。就目前而言，相比于 ASP.NET 和 J2EE 开发模型，LAMP 模型单位访问故障次数是最小的。

2.3.3. LAMP 架构技术详析

2.3.3.1. LAMP 架构基础:Linux

LAMP 架构的“L”一般泛指 Linux 操作系统，但一般认为它包括但不仅限于 Linux，即我们一般是认为“L”可以代表各种 Liunx/Unix 系统，包括熟知的各种 Debian、FreeBSD、

Redhat 以及 Suse 等系统。Liunx/Unix 系统，在 LAMP 模型处于基础作用，所有的其他开源软件均运行在 Liunx/Unix 系统上，它的高效开放性等特点切实影响着 LAMP 架构服务器的运行效率。

Linux 作为类 UNIX 的操作系统，并且服从可移植操作系统接口 POSIX 标准的开源系统。它属于自由的开源软件，通常通过各种包含了完整的安装包管理系统进行发布。作为 LAMP 架构的操作系统，截止至 2013 年 10 月 1 日止，Debian 和 Ubuntu 之间共享市场中 58.5%的 web 服务器市场份额,而 RHEL,Fedora 和 CentOS 则占据了剩余的 37.3%。

作为 UNIX 体系中最重要的分支 Linux，选择它作为 LAMP 模型的操作系统，一般是因为它具备以下四大的特点：

- 系统的高效开放性
- 多用户，多任务
- 系统的可移植性

2.3.3.2. LAMP 架构核心:Apache

LAMP 架构的“A”一般泛指 Apache 服务器，Apache Web 服务器在整套 LAMP 架构中起着承上启下的作用，它是连接着 Linux 系统和 Python 等脚本语言的桥梁。

Apache 服务器是一种高性能的开源服务器，占了全球 72%的市场使用份额。Apache 服务器在整个 LAMP 架构中起着极其重要的作用，它是整个 Web 系统的核心处理单元。它的性能直接决定着网络中心 LAMP 框架的性能高低。LAMP 架构之所以选择 Apache 服务器作为其 Web 服务器，是因为它具有以下的一些特性：

- 跨平台性

Apache 服务器是一款跨平台的 Web 服务器，它不仅能运行于 UNIX 系统中，还能广泛的部署于 Windows 系统中。

- 稳定性:

Apache 服务器使用的是多进程方式提供服务，每个进程均不能访问其他进程的资源，因此 Apache 服务器非常稳定，不容易出现死锁或者假死的情况。目前互联网中过半的服务器均使用 Apache 服务器。LAMP 架构的高效稳定特性很大程度是因为它使用了 Apache 服务器。

- 高性能的处理动态请求

Apache 在处理动态请求具有先天的优势，他比其他的 Web 服务器在准确性和速度

方面有着优势。并且 Apache 服务器功能组件丰富，它比其他的 IIS 或者 Nginx 提供更多的功能组件，方便用户扩展功能。

2.3.3.3. LAMP 架构数据中心:MySQL

LAMP 架构的“M”一般泛指关系型数据库 MySQL，MySQL 是目前应用最广泛的数据库，其市场占有量正逐步上升，为大多数企业网络系统的企业级数据库的首选。

不过 LAMP 架构的“M”的概念，从广义上可以扩展为 Memcached，即内存缓存系统，从这种观点出发，MySQL 只是 Memcached 范例中最典型的应用而已。因此，”M”可以扩展到一系列的产品，包括 MySQL 数据库的 memcache_engine/memcachedb_engine、MemcacheQ 以及 Sharedance 等产品。

从 LAMP 模型出发，其“M”在整个架构中起着数据中心的作用。车载单元 OBU 访问的资源，OBU 的各种监测信息都存储在数据库中。从服务里的角度出发，它起着一个数据服务器的作用，它辅助 Apache 服务器为用户进行数据存储和数据处理的作用。

本文主要讨论的”M”则主要代指关系数据库 MySQL，作为开源免费的数据库，MySQL 和商业数据库如 Oracle, Db2, Sybase 等，一样具备数据库的通用特点。其一，作为数据库管理系统，它们本质上都是一些结构化的数据的结合体，都需要向用户提供对数据库中的一般化处理接口，如 CRUD 操作（创建 Create，读取 Read，更新 Update，删除 Delete）等。其二，它们均是关系化数据库管理系统，均支持国际上标准的数据库语言 SQL。众所周知，在数据库发展的历程中，曾经出现过各种不同类型的数据库系统，但因为关系型数据库有着其他数据库系统没有的优越性而被广泛的使用。

除了拥有一般数据库所拥有的特性外，MySQL 还具备则以下特性：

- 开源数据库
- C/S 架构的数据库管理系统
- 高性能的内存和磁盘管理技术

2.3.3.4. .LAMP 架构粘结剂: PYTHON/PHP/PERL

LAMP 架构的“P”一般泛指 Python, PHP, Perl 等解释性脚本语言。这些”P”类型的解释性语言，能很好的与 Apache 服务器相结合，使得传统的静态 Web 网页服务得到动态交互的能力。并且，这些“P”类脚本语言，一般都自带了丰富的并且功能强大的数据库操作库函数。利用这些库函数，Apache 服务器能够简便的操作 MySQL 等数据库，进行相关的数据存储处理工作。”P”类型的解释性语言，在整个 LAMP 架构中起着粘合剂的作用，它的存在使得 LAMP 模型的服务器各个部分能够无缝的对接起来，是系统运行

的更为流畅。

本系统将采用 Python 作为服务器的 CGI 的脚本语言。这里选择 Python 最大的原因是，和前面叙述的三类系统一样，Python 不仅功能强大，而且开源免费，是一种通用型的解释性编程语言。作为”“P”的代表，Python 语言可以跨平台运行，包括：以 Linux 为内核的（Debian, Rehat, Ubuntu）,微软的 Windows 系统系列（xp/win7/NT 等），Marcintosh (包括 OSX)，以及几乎所有的类 Unix 系统。跟 Java 或者 Perl 语言相比，Python 拥有许多显著的优势，利用 Python 语言可以实现许多强大的功能并完成服务器的各种任务。

2.4. 本章小结

本章主要介绍了信息发布系统中的关键技术，即车联网通信技术以及 LAMP 架构技术。首先介绍了系统底层的通信协议 WAVE 协议栈，包括 WAVE 协议栈的结构、WAVE 通信特点、WAVE 信息传输服务以及车联网通信技术对比。其次，介绍了 LAMP 架构模型体系，包括 LAMP 架构的结构特点/系统特性以及优缺点等，为下文面向车联网应用的信息发布系统的设计奠定了基础。

第三章 车联网信息发布系统的整体设计

3.1 本章内容概要

在第一章我们提到了车联网中信息分发系统的现状，其主要的问题是现有的信息发布系统在快速移动的通信环境下，通信质量差、传输速率低、系统并发访问能力不足、业务单一，并不能满足现今大数据高信息容量传输服务要求。

在本章，我们将以此为主要解决目标，并跟据的车联网信息发布系统的实地应用环境，设计出一种具有实用价值的新型面向车联网应用的信息发布系统。该系统主要以上章介绍的 WAVE 协议栈和 LAMP 架构为技术核心，实现车-路互通，车-车互通等形式车辆网络的互联通信，并以此作为车联网通信基础，实现了各种基于车联网信息发布服务的交通应用服务功能。

3.2 系统功能需求分析

面向车联网应用的信息发布系统，其典型的运用场景便是车-车通信（V2V），车-路通信（V2I）。支持 WAVE 的车载设备 OBU，将通过 V2V 或者 V2I 的方式在快速移动的车载通信环境下，依靠路侧设备 RSU 或者不依靠 RSU 的自组网的形式，组成车联网络。其典型的运用场景图如下图 3-1 所示：

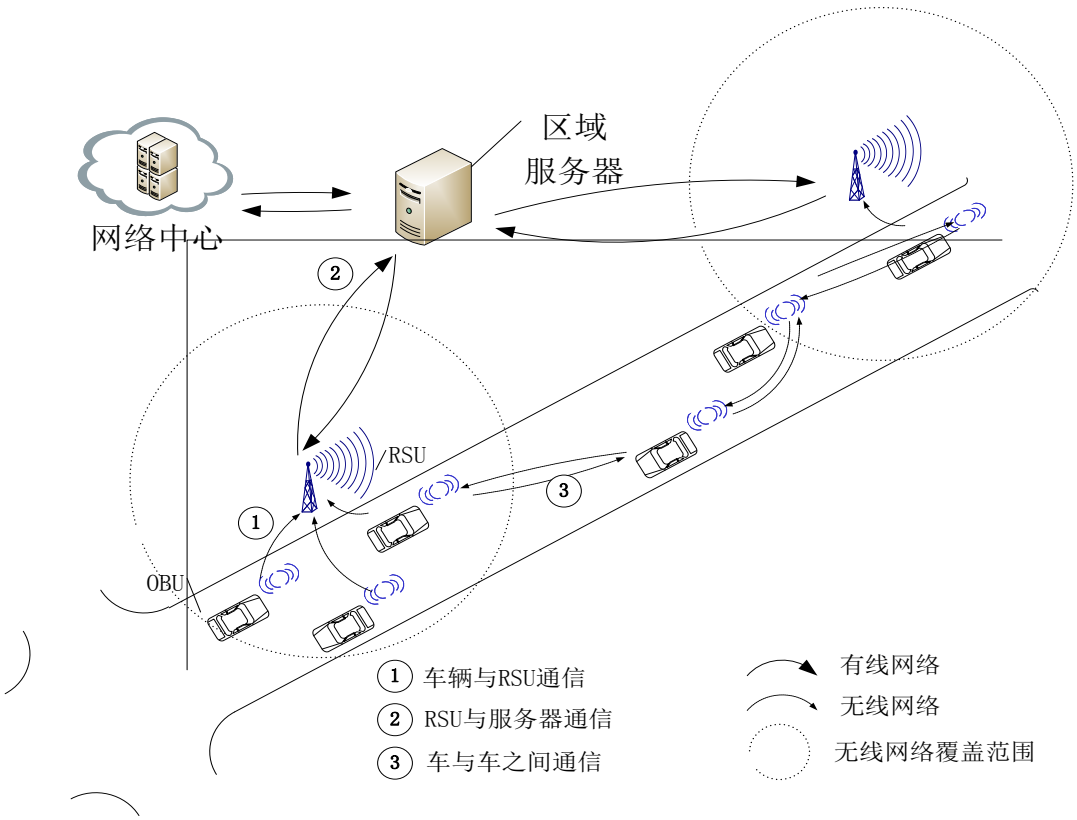

图 3-2 车联网信息发布系统运行场景图

从车联网信息发布系统的运行场景图可以清晰得出，面向车联网应用的信息发布系统，其潜在的应用服务大致可以分成三种，即交通道路安全服务、交通数据采集分发服务以及娱乐信息服务[48]。

3.2.1 交通道路安全服务需求

根据车联网信息发布系统的特点，每个车载单元 OBU 的用户，都会成为整个车联网信息发布系统数据采集者和接收者。OBU 在定时的接收来自其他 OBU 或者 RSU 的信息的同时，也会不断把自己的信息整理上传。基于这个基础，关于道路交通安全服务大致可以分为以下二种：

- 道路交通安全广播信息服务。

利用车联网信息发布系统，当系统覆盖范围中的某一位置发生如交通事故，物理道路灾害或者人为道路封闭等道路安全状况时，系统将根据车辆的位置，有针对性的实时发布道路安全广播信息。

- 车辆行驶自身安全预警服务。

路侧单元 RSU 可以与其覆盖范围的 OBU 或者其他 RSU 进行信息交互。RSU 通过采集到附近的 OBU 的传感器信息后，便可以知道其服务范围内行驶车辆的行车状况。利用 RSU 强大的计算处理能力，便可以针对每个 OBU 的特定状况发布行车指引，如变道，减速等。

3.2.2 交通数据采集服务需求

从图 3-3 的应用场景图可以得出，系统的网络中心，将不断和分布在各个位置的路侧单元 RSU 进行数据交互，每个路侧 RSU 又实时性地采集该服务覆盖范围中的 OBU 中的传感器数据，典型的应该包括车辆 GPS 位置、车速、油箱存储量、车辆类型、行驶方向等等。

交通管理部门，通过网络中心的特定 API 接口，可以方便实时获得车联网系统中的所有车辆信息，并对这些信息进行存储处理或者下行分发，这就是车联网信息发布系统的交通数据采集分发服务。其典型的运用需求可以有如下的两种：

- 道路状态动态地图监测和绘制服务

交管部门可以通过网络中心的数据库，获得所有道路 OBU 的传感器信息，例如 GPS 位置信息、车辆类型、车速、行车方向等信息，然后通过系统建模，利用地理信息系统或者交通地理信息，动态地绘制出基于车联网系统的交通状况动态地图。利用绘制的动

态地图，交通管理部门可以根据数据模型，监测道路拥堵状态，并可以根据系统的数据进行交通优化，大大的减少城市拥堵问题。

- 道路状况监控收费服务。

对于高速路段等收费路段，常常由于停车收费的原因而在高速路段的出入口发生交通拥堵。利用车联网信息发布系统，交管部门可以实时准确的测算 OBU 的行车距离，并可以通过网络实时扣费。车辆可以快速通过高速路段的出入口，而无需在收费站中停车等待。

3.2.3 娱乐信息服务需求

从图 3-4 的应用场景图可以知道，每个车载单元 OBU 均通过 RSU 或者 OBU 自组网的形式连接到互联网中。所以，面向车联网应用的信息发布系统，从需求出发，可以实现以下的娱乐信息服务：

- 基于位置的商业服务

利用面向车联网应用的信息发布系统，可以知道用户 OBU 的地理位置信息。因此，系统便可以引入各种基于用户位置的商业服务，如基于用户位置的精确投放式广告服务，基于用户位置的语音向导服务等。

- 基于互联网的用户上网服务

路侧单元 RSU 可以通过有线网络接入到外部网络中，这样车联网信息发布系统便可以利用 WAVE 协议栈在快速的移动车载环境中稳定高速的传输速率特性，提供的可靠的，远比 3G/4G 通信代价低的互联网接入服务，如在线视频，语音聊天等。

- 基于自组网的用户自娱乐服务

利用 WAVE 协议栈的通信功能，OBU 与 OBU 之间可以实现自组网式的互联互通。OBU 之间可以利用 Adhoc 模式建立连接通信，可以自己组成独立的集群网组。基于区域的组网方式，车载用户们可以实现自组网的各种服务，如多人游戏，多人视频相片共享浏览等自娱乐服务。

3.3 三层架构的体系设计

面向车联网应用的信息发布系统，其设计的目的是实现一套基于快速移动通信环境中车载单元之间进行信息交互的车联网信息发布系统[49]。由于车联网通信系统时刻处在快速移动的通信环境中，为了增加车载单元 OBU 的消息交互的实时性和可靠性，减少

网络时延等造成的信息延误，因此本系统将设计成三层架构的模式。即在传统的客户端-服务器（C/S）结构的通信系统基础上增加了一个路侧单元（在本文设计的雏形系统中将区域服务器的功能融合进 RSU 中，后续进一步商业化系统时，可以再根据需要拆分路侧单元和区域服务器）的中间层。此时，路侧单元 RSU 并不再是一个单纯的网关作用，RSU 还肩负着处理和分发消息的作用。

因此，本系统的架构整体上分为三层，从上到下为：网络中心、路侧单元 RSU 以及车载单元 OBU。同时，为了针对车联网网络快速变化的特点以及车载单元获得信息具有位置性的特性，本文设计的车联网信息发布系统各个部分所需要处理的消息总量是不一致的，系统将根据系统各部分的功能权限进行任务切割。

最顶层的网络中心可以根据外设管理信息源（如交通管理局，广告发布商等）传入的消息，获取或分发面向整个车联网网络的管理配置消息。并且网络中心内设的 MySQL 数据库将存储整个网络所有 OBU 上传的 OBUData 数据。

车联网系统处于承上启下作用的路侧单元 RSU，将根据网络中心传入的管理配置消息，采集或者发布路况安全消息以及收集其服务覆盖范围内所有 OBU 的传感器信息。与此同时，RSU 还担负着 OBU 的互联网无线接入功能，OBU 将利用 RSU 的 AP 接入点功能进行网络访问。此外，RSU 服务覆盖范围内的 OBU，并不能感知网络中心的存在，OBU 的消息拉取和推送工作都是通过 RSU 完成的。

处于系统底层并且广泛分布的车载单元 OBU，将定时的从 RSU 的广播中获取各种交通通告（如交通事故、交通拥堵等），并根据自身 OBU 的需求上传自身传感器的信息。由于车联网网络环境快速多变，OBU 并不能时刻处在 RSU 的服务覆盖范围。因此，OBU 还可以通过与其他 OBU 进行多跳的形式，接入到远端的 RSU，进而实现网络资源的接入访问。

与其他车联网信息系统不同，本文所设计的面向车联网应用的信息发布系统所有信息的交互都是双向的，即系统每一部分之间均可以通过消息的拉取和推送的方式进行消息传递。并且由于所设计的系统，在传统的客户端/服务器的结构中，增设了路侧管理单元 RSU，并让 RSU 行使网络中心的部分管理职责。这将大大减少网络中心的负荷，减少 OBU 传递处理消息的等待时间，有助于提高系统的信息发布效率。此外，由于 RSU 同样有消息的处理功能，这使得整个信息发布系统可以通过很小的代价高效的实现各种基于 OBU 位置的车联网业务，如基于 OBU 的广告精确投放服务等。

3.4 系统 WAVE 底层通信设计

车联网通信环境是一个十分恶劣而且变化迅速的移动环境，由于车辆行驶速度较快，多普勒效应特别严重。传统的基于 802.11 协议（包括 11a、11b 以及 11n 等）的无线通信方式，均绑定有十分复杂而且耗时的身份验证和鉴权的过程，这务必将造成车载单元 OBU 处在不停身份验证和切换网络的死循环中。其典型的场景是：

当某 OBU 驶入某 RSU 服务覆盖范围时 OBU 需要和 RSU 进行通信，OBU 将请求 RSU 发起身份验证。由于旧的 802.11 协议繁琐的身份验证和鉴权流程，当 RSU 确认该 OBU 身份，并允许 OBU 进行通信访问时，OBU 可能由于高速移动的特性，已经驶出该 RSU 的服务范围，进入其他 RSU 的覆盖范围。此时，由于 OBU 通信的需求没有得到解决，它将重新发起身份验证请求。至此，OBU 通信资源将耗尽在不断的身份验证中和 RSU 网络服务切换中。

为了解决这类情况，面向车联网应用的信息发布系统，拟采用以 WAVE 协议为基础的 MadWifi 驱动作为底层通信驱动 [52]。该驱动采用的是双网卡模拟 WAVE 的服务信道 SCH 和控制信道 CCH，并且其通信流程符合 802.11p 协议。

利用修改后的 MadWifi 驱动协议，用户将先通信后身份验证。即每次 RSU 底层接收到 OBU 的数据包，先检查数据链路层的数据包包头是否携带允许通行的 BSSID，如果携带则允许通信，如果没有则丢弃该数据包。此时用户的身份验证和鉴权将由上层服务实现。

本系统的底层通信驱动将均采用基于 WAVE 定制的 MadWifi 驱动，它能大大减少车载单元 OBU 的身份验证及鉴权时间，大大增加车联网信息发布系统的通信效率。MadWifi 驱动是 Linux 下基于 802.11 协议簇的（包括 11a，11b，11n）无线通信的网卡驱动。由于 WAVE 驱动协议簇使用的 802.11p 底层通信协议，所以需要定制 MadWifi 驱动，使其符合 WAVE 通信标准。

由于 MadWifi 驱动中的 802.11n 协议与最新的 802.11p 协议类似，均是使用正交频分复用技术 OFDM 作为其调制解调技术。利用正交频分复用技术将系统的通信信道分成若干个正交的子信道，并在此基础上将系统高传输速率的数据流转变成若干个低速率的数据流。这些正交的子信号可以通过相关的接收机在接收端利用正交特性来区分。使用 OFDM 作为调制解调技术，将大大的提高车联网通信的传输速率和抗干扰性能。

由于 MadWifi 驱动是支持 802.11n 协议的，因此定制满足于 WAVE 通信标准的系统底层通信协议标准，只需要在 802.11n 的 MadWifi 驱动中做少量的定制修改。根据标准，

WAVE 在传输中是支持 IP 和 WSMP 两套协议的，由于本文设计实现的只是雏形系统，故这里我们保留 MadWifi 驱动原有的 IP 模块，使用 IP 作为网络层的协议标准。在数据链路层上，802.11n 有着繁杂的身份验证和鉴权的模块[50][51][52]，在本系统将直接删除该功能模块，并相应的增加数据链路层中监测报文头部的 BSSID 域的功能模块，以替代 MadWifi 驱动的数据链路层的身份验证功能。这将大大减少 OBU 和 RSU 之间通信身份验证和鉴权消耗的通信时间，符合 WAVE 协议栈规定。

3.5 系统框架图

根据车联网信息发布系统的需求分析，得出其对应的系统框架图，如下图 3-5 所示：

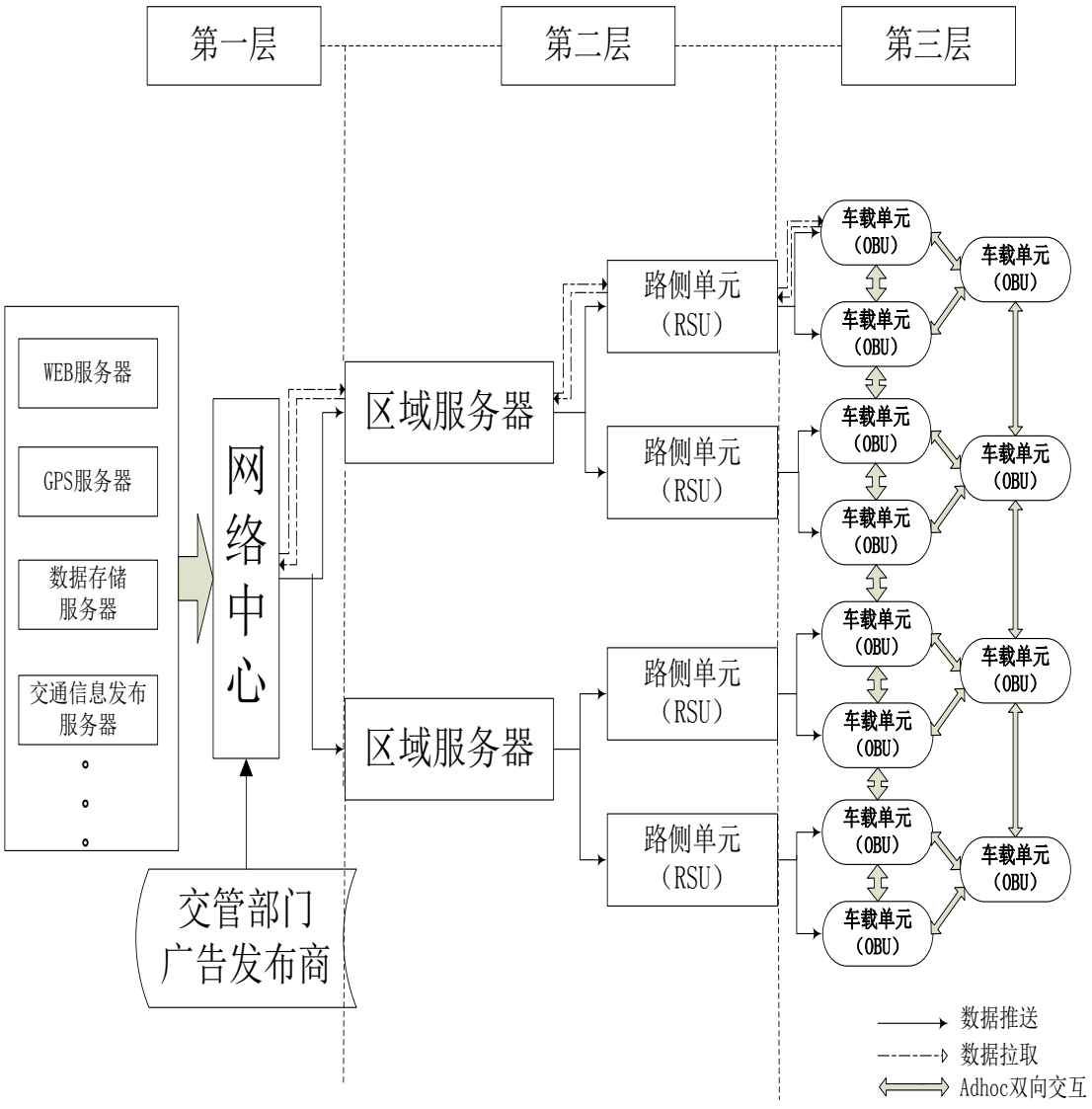

图 3-6 系统框图

3.6 本章小结

本章首先进行了面向车联网应用的信息发布系统的功能需求分析，详细的阐述了交通道路安全服务、交通数据采集分发服务以及娱乐信息服务等三大车联网服务需求。其次，在根据车联网通信环境变化快、质量差、可靠通信时间短、车辆时刻处在移动中的通信特点，创造性的设计出一种底层通信以 WAVE 协议栈为基础，三层架构的新型车联网信息发布系统，使其能满足智能交通系统中对车联网信息发布系统高速率、高可靠性以及高实时性的要求。

第四章 网络中心的设计与实现

4.1 本章内容概要

本章将根据第三章提出的车联网三层结构以及网络中心的功能需求，阐述网络中心的具体设计与实现方案。其中，网络中心的 LAMP 架构平台的搭建将主要依据于现有的技术，而数据库表的设计、数据库访问接口的设计以及 Python CGI 网关的设计(源于自主创新及研究)则是本论文的重点研究内容之一，并根据车联网网络中心的特点以及车联网业务的需求，对这三个模块进行详尽的论述。

4.2 网络中心分析与设计

根据第三章所设计的车联网信息发布系统的框架图，可以看出系统网络中心的功能与地位。位于系统第一层的网络中心对外可以获得外部信息源（如交通管理局、广告商等）输入的信息，对内可以获得所有的区域服务器的管理信息以及大部分车载单元 OBU 的数据索引信息。由于网络中心需要实时获得交管中心或者广告商输入的数据（如广告视频，图片等）以及处理车载单元 OBU 的资源访问请求，因此网络中心可以部署一个高性能高并发量的 web 服务器，本系统拟使用用的是目前运用最广泛的并且十分稳定的 Apache 服务器。

此外，由于网络中心还需要存储所有区域服务器的管理信息以及 OBU 的数据信息，因此，网络中心还必须部署专用的数据库，进行数据的并发存储管理。在这里我们采用开源的基于多线程的关系型数据库 MySQL。

鉴于以上两点原因，网络中心需要拥有高并发量、高性能以及高稳定性的处理能力，因此利用 LAMP 架构技术部署网络中心非常切合 [18]。

4.3 基于 LAMP 架构的服务器实现概要

第二章相关技术中重点提到了 LAMP 架构，本节将基于 LAMP 架构着重描述网络中心的服务器的设计与实现；并将主要通过 LAMP 框架图，以及 LAMP 架构服务器的实现过程两部分详细讲解基于 LAMP 架构的 Web 服务器的实现。

4.3.1. 网络中心框架图

面向车联网应用的信息发布系统，其网络中心将使用基于 LAMP 架构的服务器，也即为部署了 Linux（Debian）操作系统+Apache 高性能并发服务器+关系型数据库 MySQL+脚本解释性语言 Python 的物理机器。

如下图4-1所示，网络中心架构图中最低层的是Debian操作系统，在其上端为Debian系统运行的 Apache 服务器，Apache 通过 Python 脚本和 MySQL 数据库以及外部的 Web 应用进行交互。

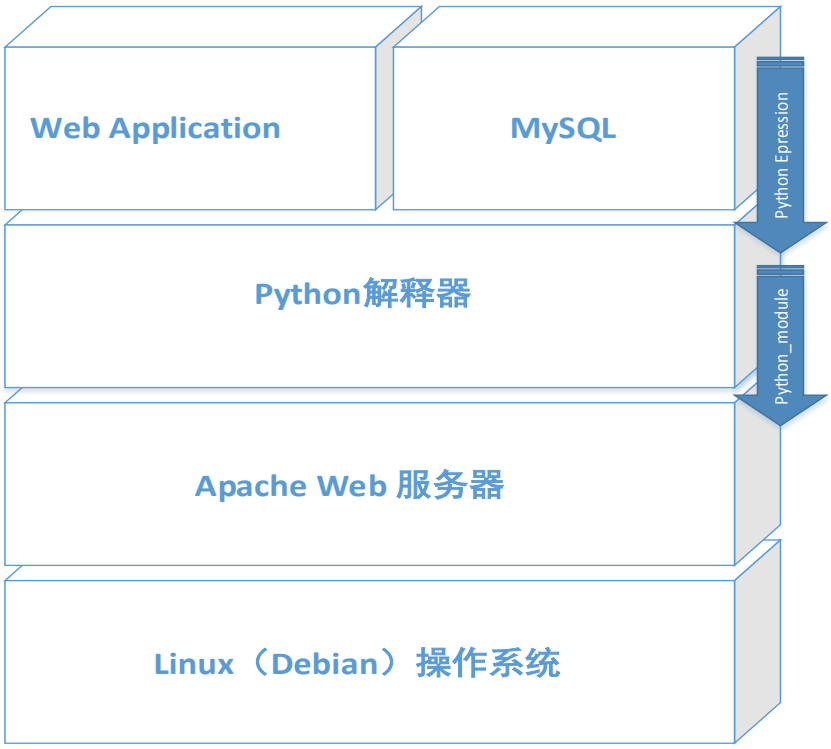

图 4-2 网络中心 LAMP 框架图

当外部的用户，如交通管理中心或者广告发布商需要访问网络中心，获取网络中心收集的数据，或者内部的 OBU 用户，需要上传或者拉取网络中心的数据时，它们仅需要使用包含了浏览器（Browser）的客户端即可实现数据的获取。当用户通过 web 浏览器访问服务器时，其本质经历了下图 4-3 所示的流程：

当客户端（Client）通过浏览器向 Apache 发送 HTTP 请求时，运行在服务器（Server）中的 Apache 服务器，通过监听 80 端口获得该请求消息。此时，Apache 服务器通过预设的 Python 网关程序，利用 Python 脚本解释器，运行 SQL 命令。与此同时，同样部署在服务器（Server）中的关系型数据库 MySQL 将会根据 Python 网关给出的 SQL 语句，处理并返回数据给 Python 解释器。Python 解释器根据返回的 Data 数据，生成浏览器可以预览的 HTML 格式数据，返回给 Apache 服务器。最终，由 Apache 服务器将用户的 HTTP 请求生成 HTML 文件，通过响应推送到客户端（Client）的浏览器中显示。服务器（Server）通过 Apache 服务器的多线程运行模式，能够高效稳定的同时处理来自不同客户端成千上万的 HTTP 服务请求。

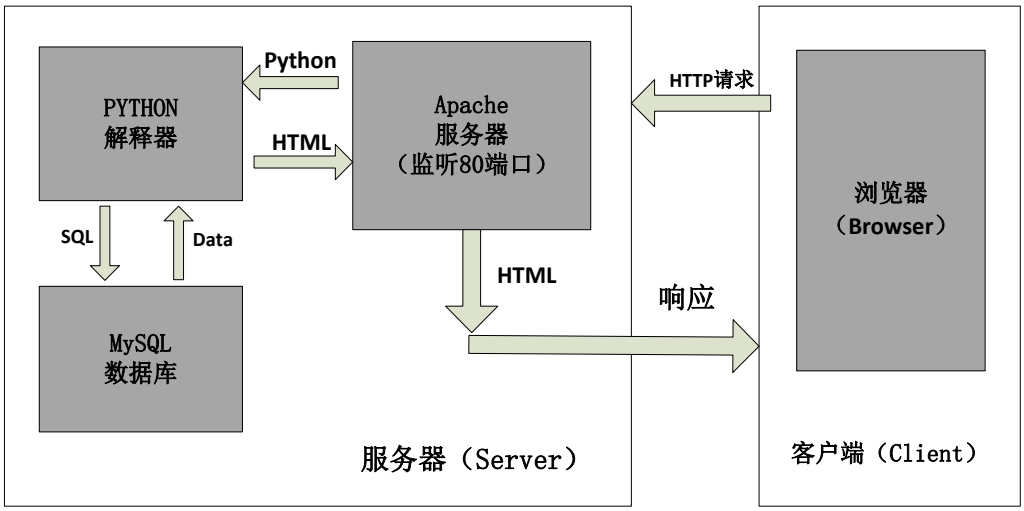

图 4-4 Web 服务器工作流程图

4.3.2. LAMP 架构服务器的部署

4.3.2.1. Debian 安装部署

Debian 与其他的 linux 发行版本，如 Readhat、 Foredo、 Ubuntu 一样，均属于 Linux 内核的操作系统。本系统所设计的网络中心将采用 Debian 的稳定版（Debian Wheezy）作为操作系统 [46]。

Debian Wheezy 的安装很简单，只需要登入 Debian 的官网下载 Debian 系统光盘镜像便可以安装使用了。安装完成后，便可以看到下图 4-5 所示的登录界面：

图 4-6 Debian 系统登录界面

Debian 系统安装完成后还需要进行环境配置，以便为后续的 LAMP 架构服务器的运行做必要的准备。Debian 系统的环境配置包括以下几点：

- 更改源：

通过命令：vi /etc/apt/sources.list 修改 Debian 系统的源，修改成国内的源。

```
deb http://mirrors.163.com/debian/ wheezy main non-free contrib
deb http://mirrors.163.com/debian/ wheezy-proposed-updates main non-free contrib
deb-src http://mirrors.163.com/debian/ wheezy main non-free contrib
deb-src http://mirrors.163.com/debian/ wheezy-proposed-updates main non-free contrib
```

- 配置编译开发环境

新安装的系统必须通过 apt-get update、apt-get upgrade 以及 apt-get install build_essential 命令更新和升级配置开发环境，这样才能让 Apache 以及 MySQL 完美运行。

4.3.2.2. Apache 安装部署

Apache 在 Debian 上的部署十分简便，只需要的终端中输入命令：sudo apt-get install apache2 即可。通过命令:/etc/init.d/apache2 start 便可以启动 Apache 服务器。Apache 服务器安装完成后，打开任意浏览器，在浏览器中输入：http://127.0.0.1 或者 http://localhost, 看到“It work!”的页面，则表示安装成功。

Apache 服务器静态网页默认的根目录：/var/www/，因此只要把静态的网页文件放在该目录下，便可以通过 Apache 服务器直接访问。

为了帮助区分 Apache 服务器不同的 http 业务，我们在/var/www/目录下，新建目录：/var/www/cgi-bin/存在车联网信息发布系统相关的 python 网关程序。然后并配置相关的 Apache 配置文件 apache.conf，使得服务器的的 URL 地址/cgi-bin/能直接映射到/var/www/cgi-bin/目录下。

在该配置文件上增加如下内容：

```
ScriptAlias /cgi-bin/ /var/www/cgi-bin/        #浏览器路径映射关系
<Directory /var/www/cgi-bin/>
    Options +ExecCGI
    AddHandler cgi-script .cgi .py
    AddHandler default-handler .jpg .png
</Dirctory>
```

该配置内容的意思是，在路径/var/www/cgi-bin/下 Apache 服务器支持 cgi 网关脚本为 cgi 文件或者 py 文件（python 文件），支持直接访问该目录的后缀为 jpg 或者 png 的文件。

4.3.2.3. MySQL 安装部署

在 Debian 系统中安装 MySQL 数据库很简单，只需要通过命令：

```
apt-get install mysql-server
```

MySQL 在安装过程中输入用户登录命令，这里请注意，MySQL 的登录密码不能和管理员登录密码相同，否则出错。MySQL 数据库的一些启动方法如下所示：

```
service mysql start #启动
chkconfig mysql on #开机启动 mysql
service mysql restart #重启
```

通过命令：mysql –uroot –ppassword（-u 后面表示用户名，-p 表示用户密码）便可以登录 mysql 数据库。

4.3.2.4. Python 安装部署

Debian 系统默认是安装了 Python 解释器的，用户只需要输入：python -V 便可以查看自己安装的 python 解释器的版本，如图 4-7 所示：

```
root@debian:/home/ora/Yagra# python -V
Python 2.7.3
```

图 4-8 Python 解释器版本

如果 python –V 没有输出，则系统没有安装 python 解释器，只需要通过 sudo apt-get install python-dev 命令安装即可。

4.4 数据库设计

4.4.1. 数据库表设计

面向车联网应用的信息发布系统，其数据均存储在网络中心的数据库中。为了实现信息发布系统的三类服务需求，即交通道路安全信息服务、交通数据采集分发服务以及娱乐信息服务。网络中心的 MySQL 数据库需要建立下列表：

1) 系统用户信息表

网络中心通过 Web 服务器为交管中心或者其他商业用户提供数据访问服务，因此需要建立系统用户信息表，以记录外部调用的用户的基本信息及其访问权限。当系统用户在系统注册时，把信息添加到该表中，如新的广告发布商入驻系统时更新表单。系统用户信息表的具体内容见表 4-1 所示：

表 4-2 SystemUserInfo 表

名称	类型	空值	说明
SystemID	varchar(50)	否	系统用户唯一 ID
Password	varchar(50)	否	登录密码
UserName	varchar(50)	否	用户名称
ServiceName	varchar(50)	否	提供商提供的服务名称
ServiceAbstract	varchar(50)	否	服务摘要
AccessLevel	Int	否	用户访问权限
Tel	varchar(50)	否	联系电话
Email	varchar(50)	否	联系邮箱
RegisterTime	Datetime	否	注册时间
LoginTime	Datatime	否	用户登入时间

2) 车载单元信息表

网络中心数据库中，除了要保存系统用户信息外，还需要保存每个车载单元 OBU 的用户信息，与系统用户信息表类似，我们需要建立如表 4-3 所示的表单：

表 4-4 OBUInfo 表

名称	类型	空值	说明
OBUID	varchar(50)	否	车载单元唯一 ID
UserName	varchar(50)	否	车载单元用户名
CarNumber	varchar(50)	否	车载单元所在车牌号
RegisterTime	Datetime	否	注册时间
LoginTime	Datatime	否	用户登入时间

3) 系统服务信息表

系统服务信息表通过外键 SystemID 与系统用户信息表关联，同一个系统用户可以允许多个服务内容。系统服务信息表主要记录了服务内容，服务提交文档，服务的有效范围以及服务的提供商（SystemID）等信息。系统服务信息表的具体内容见表 4-5 所示：

表 4-6 ServiceContent 表

名称	类型	空值	说明
ContentID	int	否	服务内容内容唯一 ID
SystemID	varchar(50)	否	服务提供者 ID，与系统用户信息表 SystemUserInfo 的 SystemID 关联
ContentFileName	varchar(50)	否	提交的内容文件名
Abstract	varchar(500)	否	提交内容摘要
RelateRegion	int	否	内容所在区域
CreateTime	datetime	否	创建时间

4) 车载单元监测信息表

车载单元监测信息表主要记录了网络中心每次由区域服务器中收到的车载单元 OBU 传感器的监测信息，如 GPS 的位置信息、车辆速度、湿度等。每条记录均有唯一的 ID 标记。该表通过 OBUID 与车载单元信息表中的 OBUID 通过外键关联。车载单元监测信息表的具体内容见表 4-4。

表 4-7 OBUData 表

名称	类型	空值	说明
ID	Int	否	记录 ID
GPSV	Float	否	车载单元所在位置经度
GPSH	Float	否	车载单元所在位置纬度
Speed	Float	否	车辆速度
Temperature	Float	否	所在位置温度
Humidity	Float	否	所在位置湿度
Time	Datetime	否	记录获取时间
OBUID	varchar(50)	否	关联的车载单元 ID

5) 车载单元服务关联表

车载单元服务关联表 ServiceRelation，主要记录了服务类型，服务提供商以及服务 OBU 对象的多对多关系。网络中心，只需要根据服务内容 ID，即 ContentID 便可以快速的获得服务的 OBUID 集合。车载单元服务关联表的具体内容见表 4-5。

表 4-8 ServiceRelation 表

名称	类型	空值	说明
ContentID	int	否	服务内容内容唯一 ID
SystemID	varchar(50)	否	服务提供者 ID，与系统用户信息表 SystemUserInfo 的 SystemID 关联
OBUID	varchar(50)	否	关联的车载单元 ID
Abstract	varchar(500)	否	服务关联摘要
CreateTime	datetime	否	创建时间

面向车联网应用的信息发布系统的网络中心，主要用到了上述的五个表，表与表之间的关系如下图 4-9 所示：

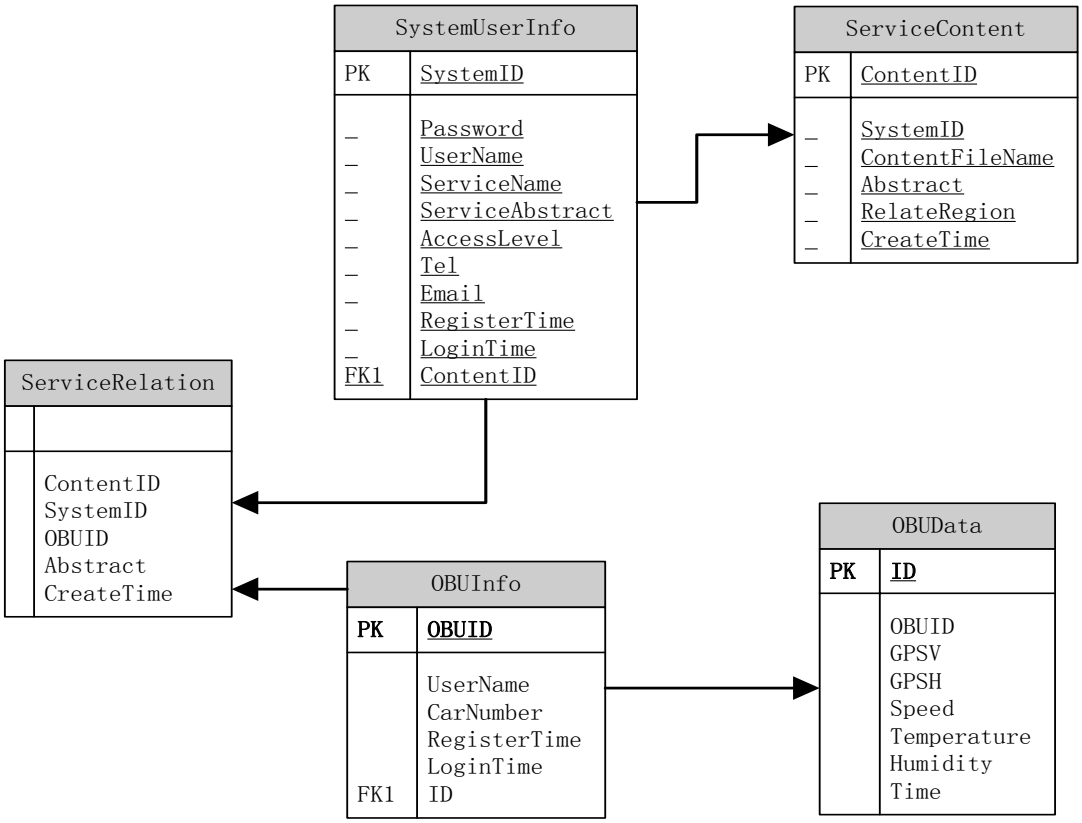

图 4-10 数据库表单关系示意图

4.4.2. 数据库接口设计

从 LAMP 架构框架图中我们可以清楚看到，网络中心的数据库是经过 Python 脚本调用的。要利用 Python 使用 MySQL 数据库，必须下载安装 Python 的 MySQL 库，即 MySQLdb。在使用 MySQLdb 提供的 API 之前，只需要 import MySQLdb 即可。

本信息发布系统利用 Python 脚本对数据库中五个表单的操作是通过定义五个数据

库访问类（5 个类对应 5 个表），五个类均封装了对相应的表进行 CRUD（Create，Read，Update, Delete）操作。即：

- 系统用户信息表 SystemUserInfo : SysUserDB 类
- 车载单元信息表 OBUInfo: OBUUserDB 类
- 系统服务信息表 ServiceContent： ServContentDB 类
- 车载单元监测信息表 OBUData: OBUDataDB 类
- 车载单元服务关联表 ServiceRelation：ServRelationDB 类

由于五个类的操作大同小异，本文只通过描述对系统用户信息表 SystemUserInfo 的接口来介绍，网络中心数据库的接口设计。

数据库表 SystemUserInfo 对应的操作类为 class SysUserDB,其定义如下图 4- 11 所示：

```
1  #!/usr/bin/env python
2
3  import os
4
5  import MySQLdb
6
7  from md5encry import MyMd5
8
9
10 class SysUserDB(object):
11     'class defination for using MySql to change table SystemUserInfo'
```

图 4-12 数据库操作类 SysUserDb

当网络中心需要访问系统用户信息表 SystemUserInfo 时，只需要 import SysUserDB, 并初始化一个 SysUserDB 实例便可。Class SysUserDB 类提供了以下一些接口，方便系统对数据库表 SystemUserInfo 进行访问：

- __init__（）

当类 SysUserDB 实例化的时候，将会默认执行该方法。在该方法中，系统在会从 MySQL.txt 文件中读取数据库的用户名，密码，默认数据库表等配置信息，并初始化数据库（获取 MySQL 数据的 cursor，self.cur = self.conn.cursor()），为用户的进一步操作做准备。

- insert_user(self, SystemID, Password, UserName, ServiceName, ServiceAbstract,AccessLevel,Tel,Email,RegisterTime,LoginTime,)

插入操作，将系统需要在 SystemUserInfo 表中增加新的用户信息时，可以直接调用本方法，实现插入操作。

- delete_user(self, user, passwd)

与插入操作相反，当系统需要在 SytemUserInfo 表中删除用户信息时，将可以直接调用本方法，实现删除操作。

- update_status(self, column, status):

更新操作接口，当系统需要在 SystemUserInfo 更新用户某列数据状态时，将列名以及对应的值传入本接口，从而实现状态更新操作。如，用户需要更改访问级别状态 AccessLevel 时，只需要调用 updata_status(“AccessLevel”,1)，即可以将用户的访问权限更改为 1。

- select_user(self, user, passwd):

信息查询接口，当系统用户需要查询系统用户信息时，只需通过传入用户名字和密码，便可以访问系统用户信息。

- check_name(self, user):

用户名字检测接口，利用本接口，用户可以快速的检测到用户名是否已经被使用，一般在注册时使用本接口。

此外，为了保护用户个人隐私，对用户输入的密码，均使用 MD5 加密算法加密后再保存进数据库中，系统无须知道用户真正的密码值，只需知道用户密码加密后的 MD5 值即可。

4.5 Python CGI 网关模块的设计与实现

在本论文的第二章我们详细介绍了 LAMP 结构，其中“P”在本系统我们采用的是 Python 脚本语言。Python 解释器，在 LAMP 结构体系中，其本质是作为一个 CGI 网关（通用网关接口，即定义在 Web 服务器和自定义脚本之间用户交换信息的一组定义标准。）存在。CGI 网关在整个 Web 浏览服务的作用可以通过下图 4-13 清晰看出：

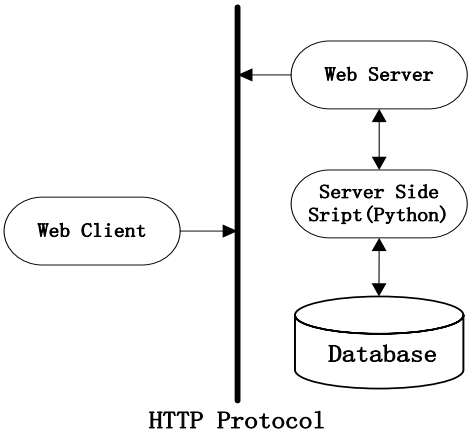

图 4-14 CGI 工作原理

CGI 脚本通常被 Web 服务器调用，其功能用于处理用户通过 HTML <FORM>或者 <ISINDEX>元素提交的数据。一般而言，CGI 脚本设置在服务器的特殊目录下（如本系统设置在/var/www/cgi-bin）。Web 服务器在该目录下，处理各式各样的客户端访问请求（如客户机的主机名,请求的 URL,查询字符串等等）以脚本的 shell 环境，执行脚本，并将脚本的输出返回给客户端。

Python CGI 脚本的输入一般通过直接连接的客户端获得，在某些情况下客户端表单的数据也是通过这种方式读取的，当然更一般的情况下是通过 URL 的请求字符串传递表单数据。Python CGI 网关模块，其功能本质上就是为了处理不同的数据请求功能，并为我们的 Web 服务器提供一个简单的访问接口。

Python CGI 网关在本系统中，主要承担以下的作用：

1）接收 Web 浏览器系统用户（交管中心，广告发布商等）登录请求，并按照预设规则读取或者存储系统用户信息，并返回相关的 HTML 数据格式。

2）处理 Web 客户端（主要是 OBU）系统发送的请求，处理 OBU 端的数据，并按要求返回规定的响应信息。

3）处理 HTML 页面跳转的逻辑功能。

4）处理 MySQL 数据库的业务逻辑功能。

本系统最典型的 Python CGI 脚本是 logintest.py。它主要负责系统用户身份鉴权并显示鉴权结果的功能，其大致的源码如下所示（只保留关键部分）：

- logintest.py 源码

```
#!/usr/bin/python
import cgi
import os
from md5encry import MyMd5
import db

header = 'Content-Type: text/html\n\n'
url = '/cgi-bin/login.py'
loginout_url = '/cgi-bin/loginout.py'
logintest_url = '/cgi-bin/logintest.py'

reshtml = '''<HTML><HEAD><TITLE>
（省略）
</BODY></HTML>'''

scripthtml = '''
<script type="text/javascript">
function Upload(){
```

```
        var url = getFileUrl("idChange");
        var dataSend = "image=" + url;
        alert(url)
        $.ajax(
        {
            type: "POST",
            url: "/cgi-bin/upload.py",
            data:dataSend,
            success:function(){
alert("send success!");
            }
        });
}
</script>'"

errhtml = '"<HTML><HEAD><TITLE>
（省略）
</BODY></HTML>'"

def process():
    form =cgi.FieldStorage()
    if form.has_key('personName'):
        name = form['personName'].value
    else:
        name = ''
（省略）
    python_db = db.SysUserDB()
    result = python_db.select_user(name, passwd)
（省略）
#show HTML

if __name__ == '__main__':
    process()
```

从 logintest.py 的源码我们可以看出，一般的 Python 网关模块，主要由两部分构成：

第一，包含了 HTML 格式输出部分。即 Python 脚本中包含了输出 HTML 格式的字符串常量如：reshtml，errhtml，headhtml，scripthtml 等。其中 scripthtml 主要负责生成 HTML 中的 JavaScript 源码。JavaScript 源码能让静态网页具有动态执行的功能，其典型功能便是能够调用异步数据传输的 Ajax 接口函数。

第二，包含了 Web 服务器业务逻辑功能部分。在 logintest.py 源码中，业务逻辑功能主要通过调用 process()函数实现。process()接口的典型作用就是获取用户的表单输入，再根据用户传入的数据，利用 Ajax 的异步执行功能，从 MySQL 数据库中查询获取用户数据，并在页面上返回显示。它先通过：

```
form =cgi.FieldStorage() #python 的 cgi 模块用户获得用户输入的表单数据
```

获得用户输入，经过输入有效性判断后，实例化 SysUserDB 数据库操作类，然后利用 SysUserDB 对系统用户进行身份验证，并获取相应的用户数据，再通过服务器回调给用户的客户端显示。

4.6 本章小结

本章主要了根据第三章提出的设计方案，从具体实现出发，详细的阐述了网络中心的设计与实现。本章首先根据网络中心的功能需求出发，即高并发量、高性能以及高稳定性特点，阐述了使用 LAMP 架构部署网络中心的必要性。其次，再根据网络中心的架构，详细的描述了包括基于 LAMP 架构 Web 服务器的搭建步骤、网络中心数据库设计实现方案（包括数据库表单设计以及数据库接口设计）以及 Python CGI 网关模块的作用和具体接口实现方案。

第五章 RSU 与 OBU 的设计与实现

5.1 本章内容概要

本章将根据第三章提出的车联网三层结构以及 RSU 与 OBU 的功能需求，阐述 RSU 和 OBU 软件系统的具体设计与实现方案。其中，RSU 和 OBU 的硬件将使用的现有的移动电脑和开发板资源；RSU 和 OBU 软件系统则是利用 QT 开源库开发实现，并将着重分析 RSU、OBU 客户端模块以及 OBU 的底层通信驱动模块的具体实现方案。

5.2 RSU 与 OBU 需求分析与设计

根据第三章所设计的车联网信息发布系统的框架图，可以看出系统第二层路侧单元 RSU 以及系统第三层车载单元 OBU 的功能与地位。位于系统第二层的路侧单元 RSU 起承上启下的作用，它对上通过有线网络连接到服务器群中，对下负责与覆盖范围的车载单元 OBU 进行信息交互，承担着网络接入点（AP）的管理工作。因此，路侧单元 RSU，应该存在以下两大功能模块：

- 通信功能模块

该模块对上承接网络中心，负责和网络中心沟通交流，对下承接与车载单元 OBU 通信以及分发信息功能。

- Web 客户端模块

该模块负责通过 HTTP 协议链接到网络中心的 Web 服务器中，以获得各种广告发布商发布的广告消息。

位于系统第三层的车载单元 OBU，起着收集分发行驶车辆信息的职责。它同时也是整个信息发布系统的基础用户。车载单元 OBU 一方面负责记录车辆的各种传感信息，如车速、GPS 位置等，另一方面还承担着与路侧单元 RSU 的互联互通的职能。

与路侧单元 RSU 功能相似，在没有路侧单元 RSU 覆盖的情况下，车载单元 OBU 同样要承担网络接入中心 AP 的作用（Adhoc 自组网模式），因此车载单元也同样包含了通信功能模块与 Web 客户端模块。

从上面分析可以看出，由于 RSU 和 OBU 功能类似，均包含通信功能模块和 Web 客户端模块，故 RSU 和 OBU 的软件客户端可以设计为同一软件系统，用户只需要根据设备的类型配置客户端启动不同的功能即可。

5.3 平台介绍

5.3.1. 硬件平台介绍

对于路侧单元 RSU 和车载单元 OBU，本信息发布系统将采用一款基于 ARM 架构的开发板—Tiny6410 作为硬件平台，并匹配一款高性能的 NETSYS-9800000N 型号的 USB 网卡（RSU 的硬件平台可以为 Tiny6410 或者普通的 Linux 移动电脑）。其硬件实物如图 5-1、图 5-2 所示：

图 5-2 开发板实物图

图 5-3 硬件整体实物图

- Tiny6410 开发板

基于 ARM11 架构的 Tiny6410 开发板，其处理器为 S3C6410 的 ARM11 架构芯片。Tiny6410 开发板的资源特性如下表 5-1 所示：

表 5-1 Tiny6410 开发板资源

Tiny6410SDK开发板资源特性如下：**Item**	**Description**
CPU	S3C6410A
频率	533Mhz~ 667Mhz
RAM	128M ~256M
Nand Flash	128M/256M/512M/1GB, 缺省为256M
指示灯	4 x User LED(在核心板), 1 x Power LED
USB Slave	1 x mini USB(底板没有设计OTG功能)
SD卡	普通SD卡座
串口	4 x RS232 DB9串口，4 x TTL电平串口座
红外	1路红外接收头
温度传感器	1路DS18B02温度传感器

- NETSYS-9800000N 型 USB 网卡

NETSYS-9800000N 型号的 USB 网卡，外设 18DBI 的全向天线。该 USB 网卡工作功率为 4200mw，工作频段为 2.4GHz，具有网络覆盖范围大、传输速率高、连接稳定性好等特点。

5.3.2. 软件平台介绍

路侧单元 RSU 和车载单元 OBU 采用的软件开发平台均使用 Qt，开发工具为 Qt Creator，是一款跨平台的可移植性的 C++界面开发平台。从功能上讲，Qt 同微软 Windows 平台上著名的 MFC，ATL 以及 X Window 上的 GTK，Mofi 等图形界面库是相似的，但 Qt 具备以下三大有点：

- 跨平台特性:
- 优良的面向对象封装
- 丰富的 API

Qt 库提供了数量庞大的各种 C++功能类，其典型的代表有 TCP 服务的 QTcpServer ,QTcpSocket,,IO 操作的 I/O device，甚至还提供了包括复杂的正则表达式的处理功能，系统开发者可以利用这些 API 进行快速开发工作。

由于 Qt 库的这三大特点，特别是其强大的跨平台移植特性，故开发者可以直接在 PC 上开发测试程序，然后无缝的移植到 Tiny6410 等嵌入式开发板中。利用 Qt 的这种

特性，可以大大的减少了系统的开发时间和测试成本。

5.4 软件设计

从路侧单元 RSU 和车载单元 OBU 的功能模块可以看出，其软件系统功能上大同小异，只存在侧重点不同。因此，从系统设计实现出发，为了提高开发效率，增加软件客户端系统的可扩展性，故设计 RSU 和 OBU 的软件为同一个系统，只是启动时根据设备角色不同，配置不同的启动功能。

由于本系统软件部分除了实现信息发布功能外，还必须提供给系统用户实时交流使用的 UI 界面，故本系统采用观察者模式实现全系统 UI 功能。系统的程序结构关系图如下所示：

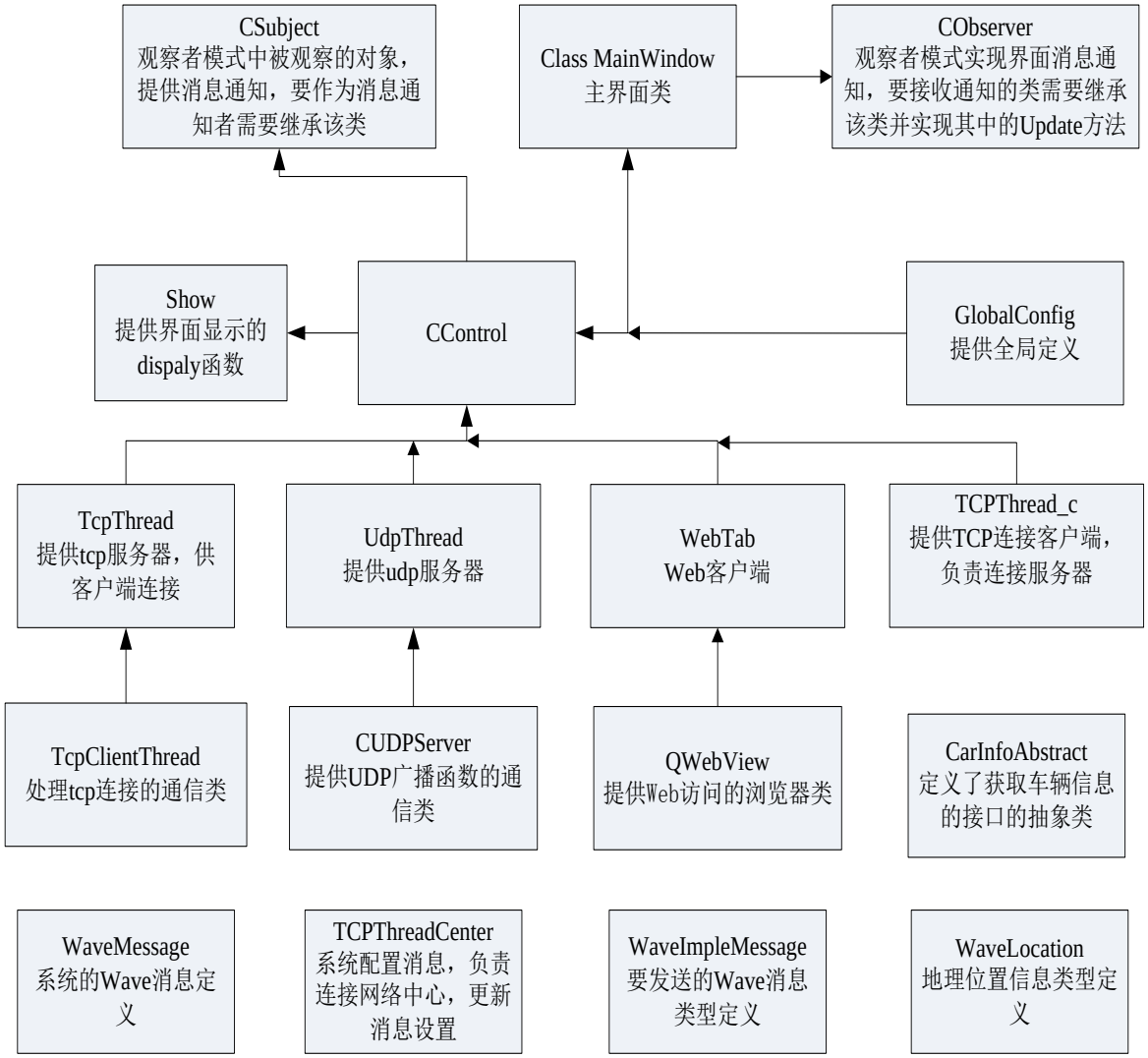

图 5-4 系统的程序结构关系图

从程序类结构关系图中，我们可以看出，系统的软件部分主要由以下的一些功能类组成，每个的功能类的典型作用如下所示：

- Class MainWindow

MainWindow 类，为整个系统 UI 界面的最高级父类。信息发布系统的所有功能都是包含在该类中。由该类生成四个主要 TabWidget 显示子类，每一子类负责一个主要功能的显示，分别为信息发布的 Server 部分，信息发布的 Client 部分，系统更新消息显示的 MesDef 部分，以及 Web 客户端部分。

- Class GlobalConfig

系统软件部分全局变量的定义处，定义了端口、IP 地址、最大缓存大小等常量。

- Class CControl

系统最重要的控制类，它包含了 Class TCPThread、ClassUDPThread、Class TCPThread_c、 Class WebTab 类。MainWindow 实例包含了一个 CControl 类，并利用该类开启 TCP、UDP 服务器线程以及和网络中心进行管理消息通信的 TCP 客户端线程。

- Class Show

负责系统消息的打印工作，本质上是调用 MainWindow 各个子控件的 show 接口打印显示消息。

- Class TCPThread

继承于 Qt 的 QThread 类，负责系统 TCP 服务器的监听和处理连接工作。TCP 服务器采用异步 IO 的方式进行通信（即采用轮询方式），利用 select 函数处理新连接以及已有连接的消息接收任务。并提供 m_client[]数组保存已有连接的套接字描述符，让 MainWindow 类可以直接调用并发送消息给 TCP 的客户端。

- Class TCPThread_c

继承于 Qt 的 QThread 类，负责系统作为客户端模式下，connect 到中心节点的 TCP 服务器的连接工作。连接建立完成后，则负责等待系统的消息推送和显示工作。提供了一个套接字描述符 m_client 的成员变量，MainWindow 实例能利用该 socket 描述符与网络中心实现异步通信。

- Class UDPThread

与 TCPThread 类似，继承于 Qt 的 QThread 类，负责系统的 UDP 服务器的建立和维护工作。UDPThread 包含了 CUDPServer 类，该类包含了一个发送广播消息的 messageList，CUDPServer 类维护该广播消息队列，并按照先进先出的方式进行消息发送。UDPThread 类提供了 ProduceMessage 接口给 CControl 类，利用该接口 CControl 类可以为该消息队

列添加发送的广播消息，并按照顺序把消息发送出去。

- Class TCPThreadCenter

继承于 Qt 的 QThread 类，是客户端软件负责管理消息更新的 TCP 客户端线程操作类。该软件启动的一开始，随着 TCP 服务器启动的时候，由 CControl 类控制开启。它首先连接到负责配置管理消息定义的网络中心，建立与网络中心的 TCP 连接。如果有消息重定义发生。TCPThreadCenter 类将收到网络中心的配置文件。该类的实例在获得新的配置文件后，将发送 update 信号给 MainWindow，更新显示新的消息定义。如果没有消息重定义，则关闭该线程。

- Class WebTab

该类是位于 MainWindow 界面类实例其中的一个控制类。该类主要包含了 Qt 的 QWebView 类，利用该 QWebView 类，该控件 WebTab 类可以简单的实现 Web 客户端功能，即从 RSU 中获得多媒体资源的 URL 地址，并循环显示。

- 其他

包括了 Class CarInfoAbstract，Class WaveMessage，Class WaveImpleMessage, Class WaveLocation 等其他消息或者资源类等。其中

CarInfoAbstract 负责获取车辆消息的接口定义的抽象类；WaveMessage 负责系统的 Wave 消息定义；WaveImpleMessage 负责要发送的消息类型的定义；WaveLocation 负责地理位置消息的类型定义。

此外，RSU 和 OBU 的客户端软件，如果从功能上划分，系统软件部分还可以分成三大模块:

- 信息分发功能模块
- Web 客户端模块
- 系统配置更新模块

5.4.1. 信息分发功能子模块

5.4.1.1. 作为服务器 Server 状态配置启动时的信息发布子模块

路侧单元 RSU 和车载单元 OBU 均需要信息发布功能模块。但两者的功能略有不同。当客户端软件作为服务器的 Server 状态时，信息发布功能模块具有以下的功能：

- UDP 广播通信功能

在用户启动系统的服务器版本时启动 UDP 服务器。UDP 服务器，其功能主要分为

两点：第一，广播告知发现功能。路侧单元 RSU 或者在没有 RSU 的通信环境下 OBU 打算通过自组网通信并且愿意成为临时的 AP 中心时，都会通过 UDP 广播的形式，发送消息给所有的 OBU 告知自身的存在。第二，交通安全信息广播功能。当路侧单元接到网络中心或者附近其他 RSU 传输的安全消息时，将通过广播的形式将消息传送给覆盖范围的所有车载单元 OBU。UDP 广播信息通信模块，其程序流程图如图 5-4 所示：

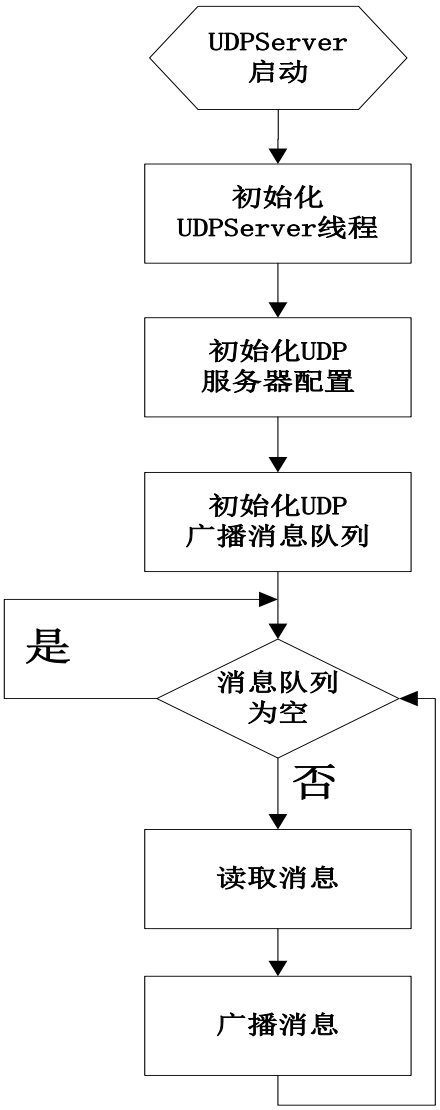

图 5-5 UDP 广播通信模块流程图

从 UDP 广播通信模块，可以看出，只要往 UDP 服务器的广播消息队列中增加消息，UDPServer 线程将会自动的广播出去。因此，我们在系统中还增加了一个 UDP 服务器的异步消息广播接口（线程安全）：

```
void CControl::ProduceMessage(BroadObj mes)
{
    this->m_udpthread->Produce(mes);
}
```

从代码可以看出，该接口本质上还是调用了 udpthead 线程的 Produce 接口函数，为 UDP 服务器消息队列增加广播消息。利用该接口，客户端在需要广播消息时，只需要简单的调用该线程安全的广播 API 接口，便可以随意使用 UDP 广播方式发送消息。

- TCP 服务器通信功能

在用户启动客户端软件的服务器版本时启动 TCP 服务器。和其他 TCP 服务器类似，该服务器有两大功能：一是绑定端口，然后监听来自其他客户端的 connect 连接请求，二是利用已有的套接字 socket，同时和其他客户端软件进行消息收发工作。不过和传统的 TCP 服务器不一样，本系统的 TCP 服务器通信模块并非采用多进程或者多线程模式，（即一个进程或者一个线程在不停的监听 TCP 服务器的端口等待连接，然后由该进程或者线程根据客户端的请求建立新的子进程或者线程进行通信），而是采用单线程 IO 异步的形式建立 TCP 服务器。

传统的 TCP 服务器是基于多线程的阻塞模式，也即阻塞式 IO 操作，它们两个一般有如表 5-2 所示的区别：

表 5-2 同步 IO 和异步 IO 对比

同步IO操作（特点）	异步IO操作（特点）
利用多线程提高吞吐量	单线程即可实现高吞吐量
通过时间片分割和线程调度利用多核CPU	通过功能划分利用多核CPU
需要由OS调度多线程使用多核CPU	可以将单进程绑定到单核CPU
难以充分利用CPU资源	可以充分利用CPU资源
内存轨迹大，数据局部性弱	内存轨迹小，数据局部性强

TCP 服务器通信模块的流程图如图 5-6 所示：

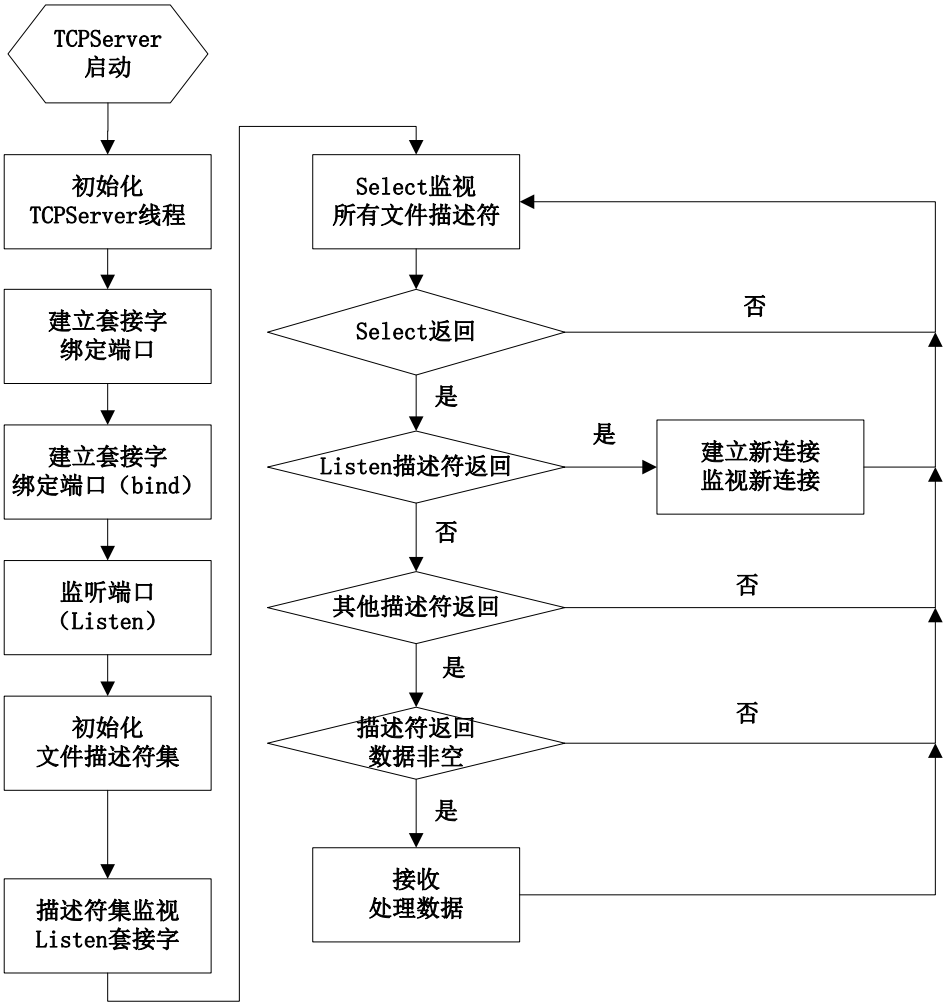

图 5-7 TCP 服务器通信功能流程图

由于 TCP 服务器监听端口和接收数据的 IO 操作等待时间较长，故在本系统中交由 select 异步 IO 操作线程进行管理操作。但由于服务器的消息分发具有不可意料性，故除了提供 select 异步 IO 操作线程之外，本系统还提供了一个 TCP 服务器端发送接口：

```
void MainWindow::on_bnt_send_message_clicked()
{
    QString mes,number;
    mes=ui->lineSend_s->text();
    number=ui->lineClientNumber->text();
    int number_client = number_client=number.toInt()-1;
    int sockfd=this->m_pControl->m_tcpthread->m_client[number_client];
    int n =write(sockfd,mes.toLatin1().data(),mes.length());
    if (n!=0)
    {
        number=QString("Send to Client %1:").arg(number_client,0,10);
        ui->textEdit->append(number);
        ui->textEdit->append(mes);
    }
    ui->lineSend_s->clear();
}
```

该接口本质上是利用了 CControl 类中包含了 select 子线程的套接字描述符的接口，利用该接口可以获得 select 子线程中所存储的和客户端建立连接的套接字描述符。然后再利用 socket 的 write 函数，将信息发送给 TCP 客户端（在本系统即为 OBU）。

5.4.1.2. 作为服务器 Client 状态配置启动时的信息发布子模块

当系统用户为车载单元 OBU 时，并且 OBU 不作为自组网的中心节点的情况下，将启动客户端 Client 状态的软件系统。在本状态下，软件系统只需启动 TCP 客户端，与中心节点的 RSU 或者 OBU 进行点对点的通信即可。其 TCPClient 线程的程序流程图如图 5-8 所示：

该线程主要负责消息的接收工作，由于客户端发送消息的异步性以及不可预料性，故额外提供了一个用户发送消息的接口函数：

```
void MainWindow::on_bnt_Send_clicked()
{
    QString mes;
    int sockfd=this->m_pControl->m_tcpthread_client->m_sockfd;
    if(sockfd==-1)
    {
ui->textEdit_2->append("Send error,Socket haven't open!:");
    }
  else
    {
        mes=ui->lineEdit_send->text();
        int i = write(sockfd,mes.toLatin1().data(),mes.length());
        if(i!=0)
        {
            ui->textEdit_2->append("Send to Server:");
            ui->textEdit_2->append(mes);
        }
        else
        {
ui->textEdit_2->append("Send Error!");
        }
            ui->lineEdit_send->clear();
    }
}
```

和服务器 Server 模式下的消息发送类似，on_bnt_Send_clicked 接口本质是用获得客户端线程所得到的套接字描述符 socketfd，然后利用该描述符进行异步消息的发送。

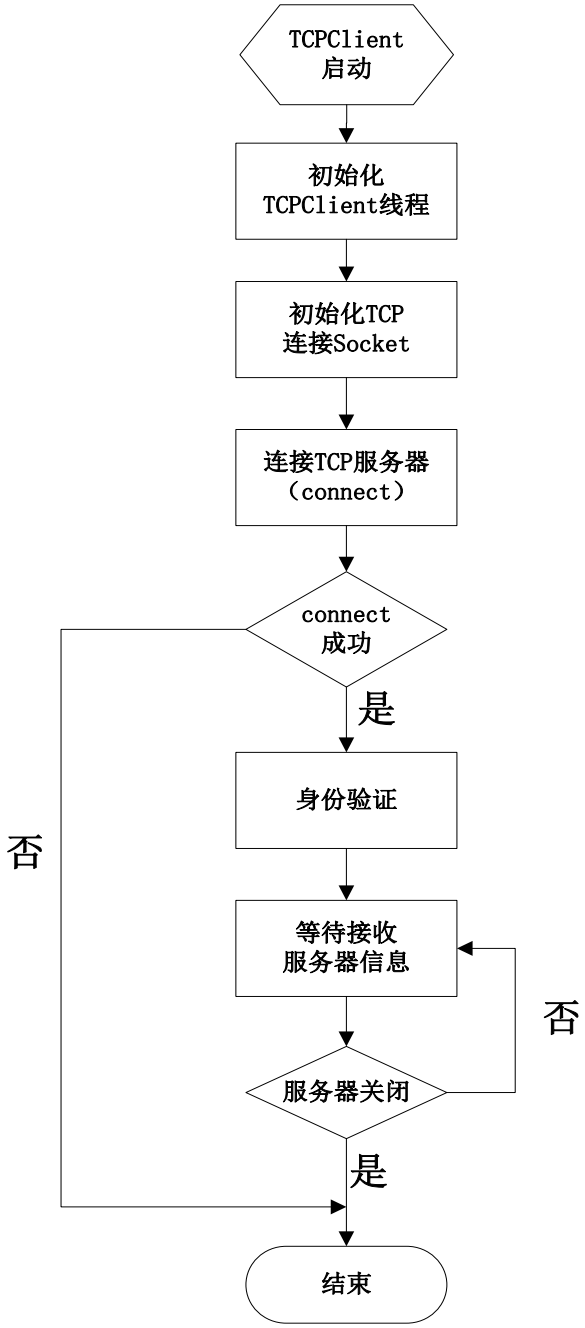

图 5-9 Client 模式下信息发布子模块流程图

5.4.2. Web 客户端子模块

Web 客户端模块，其本质是一个移动端的浏览器，其主要功能就是访问网络中心或者 RSU 的 Web 服务器，以获得广告发布商等发布的基于位置的多媒体资讯。Web 客户端模块本质上是通过 Qt 自带的封装功能完善的 QWebView 类实现。

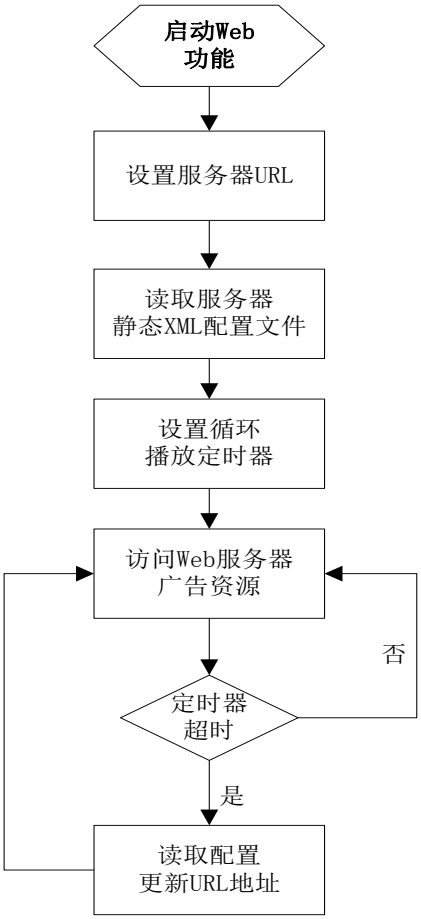

图 5-10 Web 客户端子模块流程图

从图 5-11 可以看出 Web 客户端模块的运作流程，它先从指定服务器获取静态 XML 配置文件（主要是保存了车载单元 OBU 需要访问的基于位置的网页 URL 地址）。获取到 XML 的配置信息后，Web 客户端模块便利用模块的 cycle 接口，循环访问 Web 服务器的网页资源。

```
void WebTab::cycle()
{
    QVector<QString>::iterator it=reader.urlvector.begin();
    for(it;it!=reader.urlvector.end();it++)
    {
        QString str=*it;
        QUrl url2(str);
        web1->load(url2);
        web1->show();
       QTest::qWait(5000);
     }
}
```

5.4.3. 系统配置更新子模块

本模块负责 RSU 和 OBU 配置更新功能。由于车联网信息发布系统内部管理的需要，

希望信息发布系统各部分之间，能够传递基于服务管理配置的消息。而消息的定义是由网络中心确定，并通过 RSU，一级级的传递给每个信息发布系统的用户。消息的定义、消息的类型、消息的增加和删除均处于时刻变化过程中。因此，路侧单元 RSU 以及车载单元 OBU 均需要增加系统配置更新模块。利用该模块，RSU 和 OBU 均能通过 TCP 通信直接或者间接的从网络中心处接收新的消息定义文件，并根据该文件更新消息定义。其获取更新的流程如图 5-12 所示。

系统配置更新模块本质上是一个 TCP 客户端的线程，它通过 TCP 连接直接连接到网络中心，获得新的消息定义配置文件，然后再更新系统的消息定义。该模块最主要的功能是通过 socket 连接获得的数据保存到配置文件 message.dat 中，该功能通过 mesSave 接口实现：

```
void    TcpThread_center:: mesSave ()
{
      ofstream outfile("message.dat",ios::app);//输出方式打开，写入数据文件末尾
      for(int i=0;i<6;i++)
      {
            memset(this->buf, 0, MAXDATASIZE);//每次调用，确保收到消息正确。
            if((nbytes =read(sockfd, buf, MAXDATASIZE-1))==-1)
            {
                  perror("read error");
                  exit(1);
            }
            if(nbytes == 0)
            {
                   printf("receive EOF...\n");
                   break;
             }
             else
             {
                    this->str_dispose=buf;
                    if((str_dispose.find("Stop")!=-1) || (str_dispose.find("File#%#") != -1))
                    {
                           exist=true;
                    }
                    else
                    {
                           exist=this->m_TabWidget->chbuff_fromctcp(buf);
                     }
         if(exist)
                     {
      cout<<"message exist."<<endl;
                      }
                     else
                     {
                            buf[nbytes]='\n';
                            outfile<<buf;
                      }
```

```
                }
            }
outfile.close();
}
```

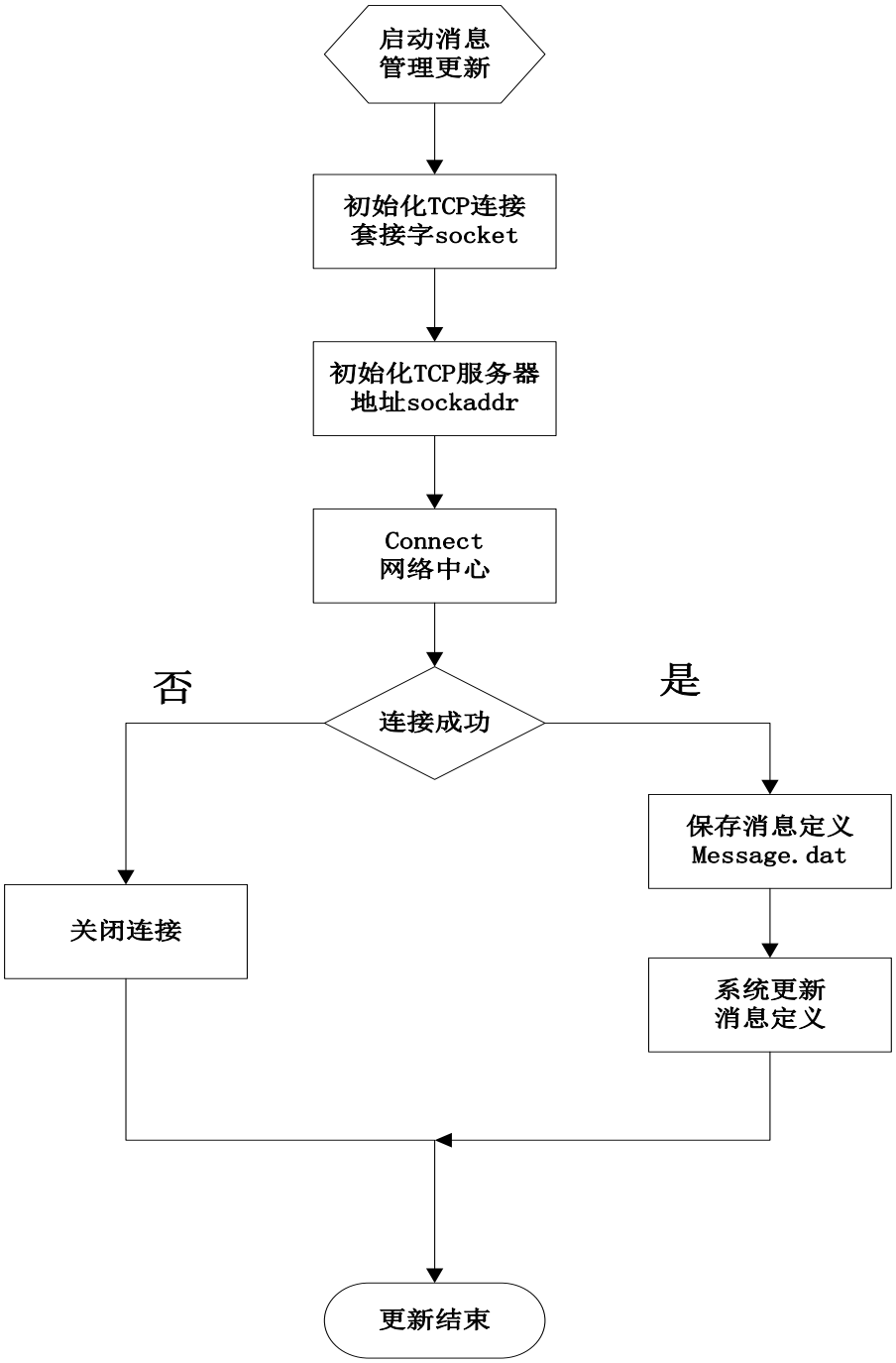

图 5-13 系统配置更新子模块流程图

5.4.4. WAVE 底层通信子模块

WAVE 底层通信子模块，本质上为 RSU 和 OBU 的通信驱动模块，在这里我们采用 MadWifi 协议为模板，定制其 WAVE 通信功能。WAVE 底层通信子模块主要工作如下：

1. 检测设备

当设备接入到操作系统，OBU 核心板会调用一个通信驱动程序中的探测函数来检测此驱动程序与新接入的设备是否匹配。探测函数根据 OBU 核心传入的 pci_dev 结构体和记录着本驱动程序所支持设备的 pci_device_id 结构体来判断。如果两者不匹配，那么 OBU 核心将会调用另一个驱动程序来进行相同的操作。

2. 初始化 MadWifi 网络处理实体 net_device

内核通过调用 alloc_netdev 接口实例化驱动的网络处理实体 net_device，其本质是通过驱动 API 接口 ether_setup 完成的。在这里 WAVE 底层通信子模块设置保留 MadWifi 驱动原有的 IP 模块，使用 IP 作为网络层的协议标准。在数据链路层上，802.11n 有着繁杂的身份验证和鉴权模块，WAVE 底层通信子模块将直接在该结构体中删除该功能，并相应的增加数据链路层中监测报文头部的 BSSID 域的功能函数，以替代 MadWifi 驱动的数据链路层的身份验证。

3. 注册中断处理程序

WAVE 底层通信子模块，对数据包的接收和发送是通过 CPU 中断模式实现对硬件的操作的。因此，定制后 MadWifi 驱动将会完成中断号以及中断处理程序的注册工作。此后，接收和发送数据本质将通过已在 CPU 注册后的中断程序实现。

5.5 本章小结

本章主要了根据第三章提出的设计方案，从具体实现出发，详细的阐述了路侧单元 RSU 和车载单元 OBU 的客户端软件设计与具体实现。本章首先从路侧单元 RSU 和车载单元 OBU 的需求出发，介绍了 RSU 和 OBU 的软硬件平台以及开发环境 Qt Creator。其次，再根据给出的系统的程序结构关系图，从系统功能类的角度介绍了客户端系统的构成。最后，再从具体功能出发，详细的介绍系统的信息分发功能子模块、Web 客户端子模块、系统配置更新子模块以及 WAVE 底层通信子模块的具体实现，包括各子模块的功能介绍、程序流程图以及关键接口的介绍。

第六章 系统测试

6.1 本章内容概述

本章主要根据第四、五章的信息发布系统的设计和实现，从功能以及性能的角度出发，对整个面向车联网应用的信息发布系统进行基本的测试(包括仿真测试和实物测试)。本章将重点测试以下三项内容：

- 网络中心和 RSU、OBU 客户端软件的服务可用性
- 车联网通信网络性能
- Web 服务器和数据库的并发访问性能

并通过这三项测试，与传统车联网系统一一对比，阐述本文所设计的车联网信息发布系统的优势所在。

6.2 测试目的和环境部署

面向车联网应用的信息发布系统，其设计的目的便是实现一个可以在道路上部署服务并用于车辆用户和路侧系统之间交流通信的信息发布系统。本节将通过仿真测试和实物测试对本文所研究的系统进行可行性验证。但由于场地以及硬件系统所限，本文所设计并且实现部署的仅为车联网信息发布系统的演示雏形系统。部署本演示雏形系统的最主要目的，是验证本论文所设计的车联网通信系统的可行性以及其服务的可用性。本系统采用如下硬件设施进行部署：

1. 安装了 Debian（Linux）系统的服务器一台（网络中心）

网络中心为一台 Intel 双核主频 2.20 GHz，内存为 1G 的服务器，其操作系统为 Debian 系统，并安装部署了 Apache2.2 服务器，MySQL5.5 数据库服务器，Python2.7 的脚本解释器。

2. 安装了 Ubuntu（Linux）系统的笔记本一台（路侧单元 RSU）

路侧单元 RSU 的操作系统为 Ubuntu 系统，安装部署了 Apache 2.2 服务器，并配备 NETSYS-网卡一块。

3. 安装了 Linux 系统的 Tiny6410 开发板四台（车载单元 OBU），

每个 Tiny6410 开发板操作系统为内核为 Linux 2.6 的嵌入式系统，通过串口外接了一个 GPS 传感器。

由于本系统只是简单的演示雏形系统，故车载单元外接的传感器只限于传输位置的 GPS 传感器，其他的如温度，车速传感器等均与 GPS 传感器的工作原理一致，待后续

开发升级时再加入到 OBU 中。

6.3 系统测试

在本节，将首先通过 iTETRIS 系统对车联网信息发布系统进行动态移动的功能验证。其次，再利用本文设计实物雏形系统于大学城外环公路上进行静态的功能验证和性能测试。

6.3.1. 仿真系统 iTETRIS 验证

仿真系统 iTETRIS 是一个无线通信的交通仿真平台，它支持手动配置通信协议栈。iTETRIS 系统可以根据开发人员需求进行基于 WAVE 协议栈通信的仿真场景测试，包括节点（node）设计、路段（edge）设计、连接（connection）设计、道路网络（network）设计以及交通工具与路线(vehicles and routes)设计。

本节仿真测试的场景如下：双十字路口，四个方向的路段都有 4 条车道，其中一条是驶离十字路口的车道，另外三条是进入十字路口的车道，按其车道上指定的方向标行驶，并受红绿灯的控制。路侧部署了 RSU 设备，车辆中设置了 OBU 设备。配置 iTETRIS 仿真系统网络协议栈为 WAVE 协议栈。如下图 6-1 所示：

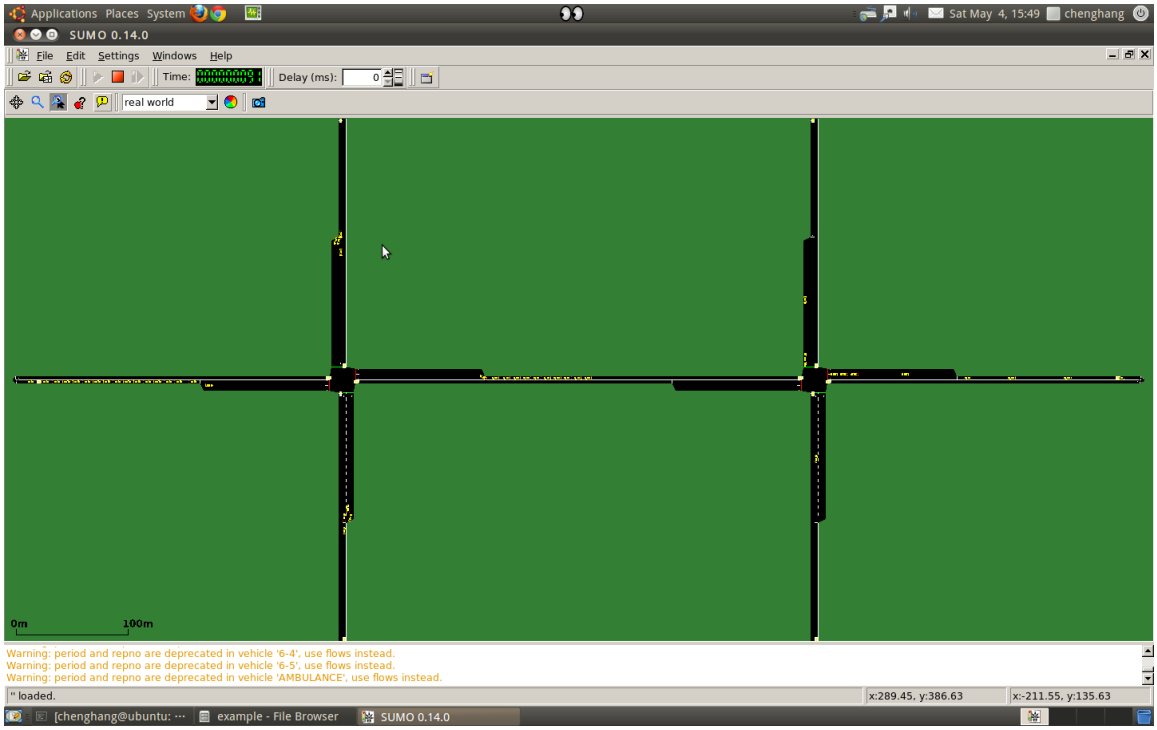

图 6-1 iTETRIS 仿真全局图

iTETRIS 的 Application 模块能定制一个或多个智能交通系统的应用逻辑功能。利用 Application 模块，我们可以把本文所设计交通道路安全服务、交通数据采集分发服务以

及娱乐信息服务通过 iTETRIS 仿真系统进行验证，其典型的仿真测试结果如下图 6-2、图 6-3 所示：

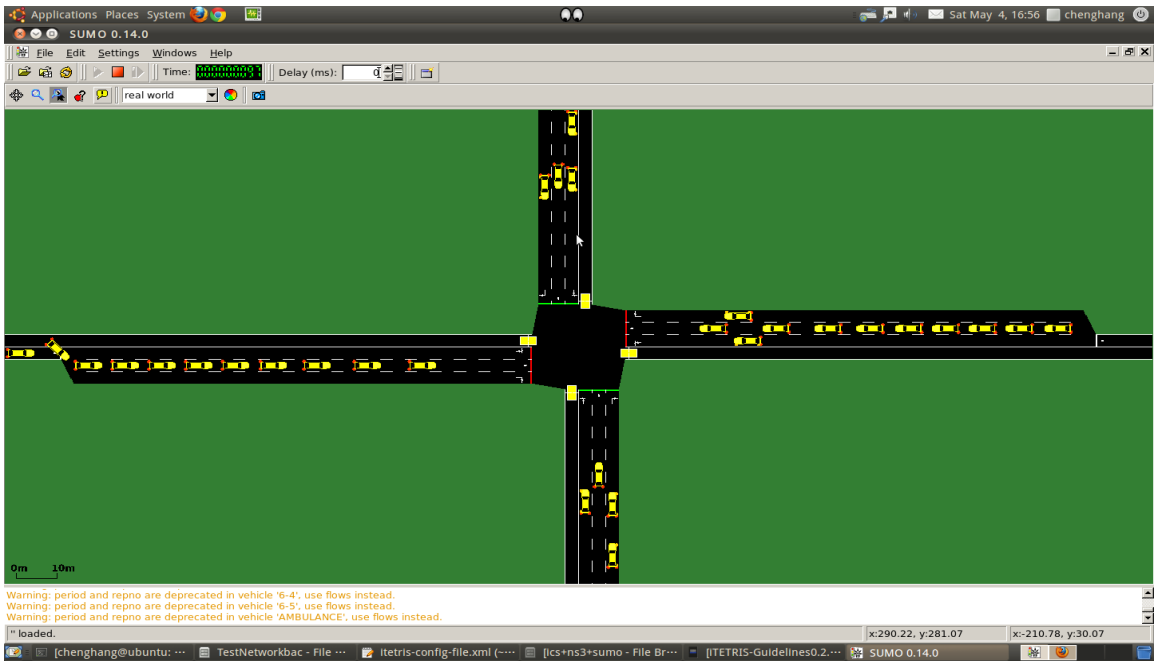

图 6-2 安全消息广播仿真演示

道路中的所有车载 OBU，均实时接收路侧单元 RSU 的安全广播消息，包括停车等待、减速换道、合并车道等。在 time step 到达 97s 时，RSU 广播了停车等待的 WSA 消息，收到此消息所有想要通过左侧十字路口的车辆都被迫停止，如图 6-2 所示。

图 6-3 切换车道仿真仿真

RSU 利用 WAVE 协议栈不停的和服务覆盖范围重点 OBU 进行信息交互，它通过实

时收集的 OBU 位置、车速、车道等传感器消息，经过处理器运算后，可以为 OBU 提供最佳运行线路规划建议。如图 6-3 所示，RSU 根据计算的最佳运行线路，对 OBU 发出切换车道的消息。

通过仿真系统 iTETRIS 容易得出，从理论上而言以 WAVE 协议栈作为车联网通信技术的信息发布系统，对实现交通道路安全服务、交通数据采集分发服务以及娱乐信息服务三大车联网业务是完全可行。

6.3.2. 系统功能测试

在本节中，将对面向车联网应用的信息发布系统的三层结构，即网络中心、路侧单元 RSU 以及车载单元 OBU 分别进行功能测试。

6.3.2.1. 网络中心功能测试

网络中心的功能测试分为两部分，即 Web 服务器功能测试以及 MySQL 数据库功能测试。

- Web 服务器功能测试

车联网信息发布系统的 Web 服务器主页地址为：http://192.168.0.110/cgi-bin/homepage.py。通过 Chrome 浏览器打开的界面如图 6-4 所示：

图 6-4 车联网信息发布系统主页

主页提供了车联网系统的简介以及各类功能导航，包括：

- 交管中心：用于交通管理服务系统配置的管理功能
- 广告发布：用于广告发布的资源更新的管理服务功能
- 监测管理：用于车载单元 OBU 监测的管理服务功能
- 服务配置：用于新服务类型配置的管理服务功能
- 系统用户登录：基于车联网 Web 服务器系统用户的登入管理功能

下图为用户点击了监测管理用的浏览器界面图，它能根据用户输入的车载单元 OBU 的 ID 或者路侧单元 RSU 的 ID 从数据库中调出该 ID 所代表的用户上传到网络中心的道路监控消息，并根据其上传的位置，在地图中标记出来。标记分成四类，分别代表四种传输消息的不同的优先级，从高往低分别是红色（A），橙色(B)，黄色(C)以及蓝色(D)。其典型功能验证案例如图 6-5 所示：

而“交管中心”，“广告发布”则是进入对应系统类型的用户，配置阅读并定制其特定专属服务的 Web 控制界面，“服务配置”则是系统用户配置管理服务类型的 Web 控制界面，“系统用户登录”为系统用户注册或者修改系统用户消息的控制界面。

图 6-5 车联网信息发布系统监测管理页面

- MySQL 数据库功能测试

位于网络中心部署的 MySQL 数据库，其功能的核心是存储路侧单元 RSU 以及车载单元 OBU 上传的所有道路消息，以及系统用户（目前只有交管中心和广告发布商）的管理配置消息，并配合 Apache 服务器提供基于 HTTP 协议的 Web 远程服务。

从 MySQL 服务器端可以看到，本系统建立了名为 IOV（Internet of Vehicles）的数据库，数据库中包含了如第四章所叙述的数据库表单：系统用户信息表 SystemUserInfo、车载单元信息表 OBUInfo、系统服务信息表 ServiceConten、车载单元监测信息表 OBUData 以及车载单元服务关联表 ServiceRelation。MySQL 数据库中的表如图 6-6 所示：

```
mysql> use IOV
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----------------+
| Tables_in_IOV   |
+-----------------+
| OBUData         |
| OBUInfo         |
| ServiceContent  |
| ServiceRelation |
| SystemUserInfo  |
+-----------------+
5 rows in set (0.00 sec)
```

图 6-6 MySQL 数据库表

6.3.2.2. 路侧单元 RSU 和车载单元 OBU 功能测试

位于信息发布系统第二层的路侧单元 RSU（在雏形系统中包含了区域服务器功能）和第三层车载单元 OBU 负责整个车联网信息发布系统的消息推送和拉取工作，其本质是系统的消息收集者以及消息的接收者。启动的客户端软件的 RSU 和 OBU 如下图 6-7 所示。目前，面向车联网应用的信息发布系统主要实现了以下四类功能，即路侧单元 UDP 广播功能，路侧单元 RSU 与车载单元 OBU 通信功能，车载单元 OBU 消息上传功能以及车载单元 OBU 基于位置的服务功能。在本小节里面将分别对着四种典型应用做功能性测试。

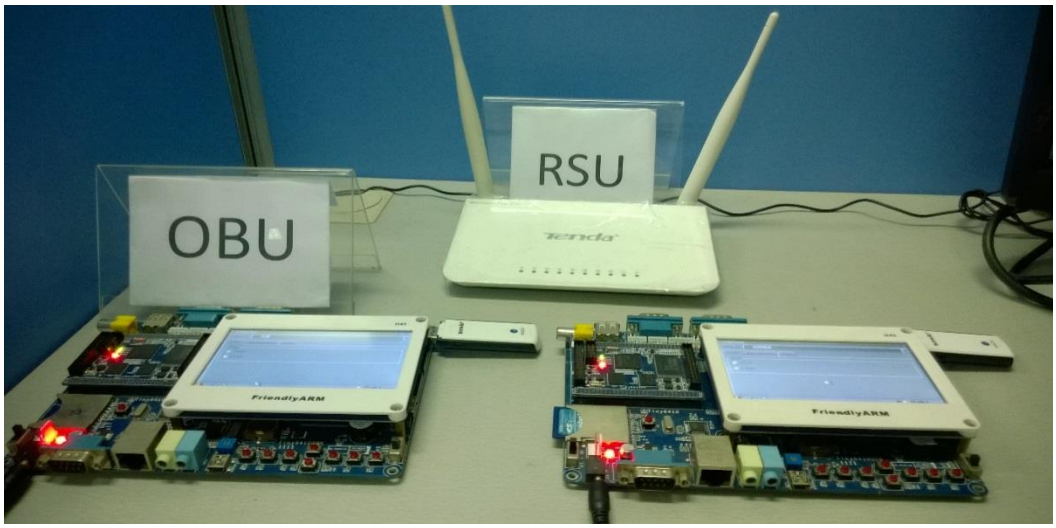

图 6-7 启动客户端软件的 RSU 和 OBU

- 路侧单元 UDP 广播功能测试

路侧单元 RSU 和车载单元 OBU 在作为网络接入点 AP 时，将会担任子网络的点协调中心的重任。这时候，系统用户只需要启动客户端软件的服务端模式即可。客户端程序将自动对所覆盖范围的所有其他 OBU 发送 UDP 广播包（如图 6-8、图 6-9 所示）。接收到 UDP 广播包的车载单元 OBU 便知道附近有 RSU 或者 OBU 愿意充当网络接入点的职能。此时，其他的车载单元 OBU 将根据用户自身的需要选择是否加入该网络中。

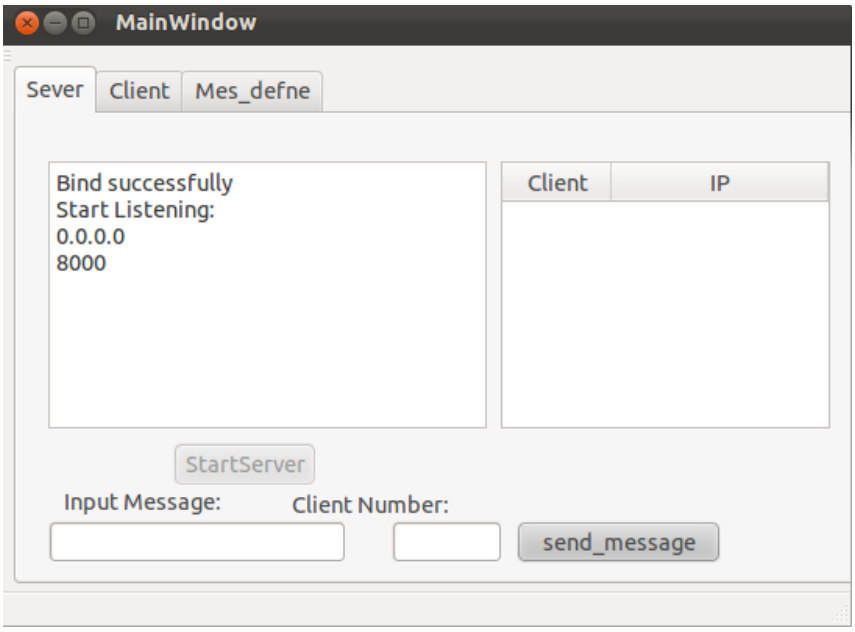

图 6-8 RSU 启动 Server 模式

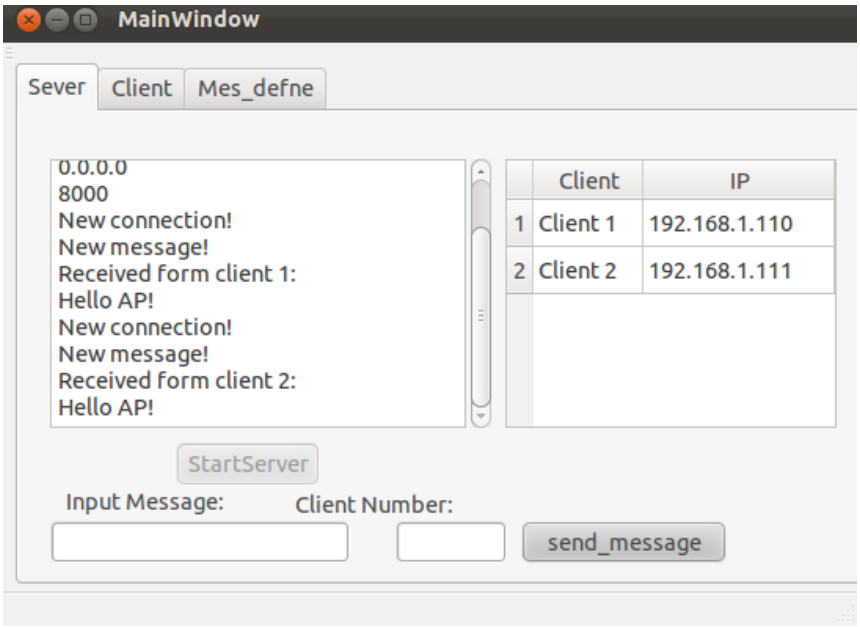

图 6-9 RSU 收到其他 OBU 的接入请求

- 路侧单元 RSU 与车载单元 OBU 通信功能测试

路侧单元 RSU 和车载单元 OBU 之间最基本的功能便是信息交互，当 RSU 通过 UDP 广播告诉周围的 OBU 自身存在的消息之后，周围的 OBU 便可以根据自身的需要和中心节点 RSU 建立 TCP 连接，并进行消息推送与拉取工作。

如图 6-10、图 6-11、图 6-12 所示：OBU1 为 ID 为 B20140001 的车载单元，其 IP 地址为 192.168.1.110；OBU2 为 ID 为 B20140002 的车载单元，其 IP 地址为 192.168.1.111；RSU1 为 ID 为 R20140001 的路侧单元，其 IP 地址为 192.168.1.109。首先，当 RSU 通过 UDP 广播自己的存在及相关信息之后，周围的存在 OBU1，OBU2 便与之建立 TCP 连接，并通过该连接同时与 RSU 进行信息交互。

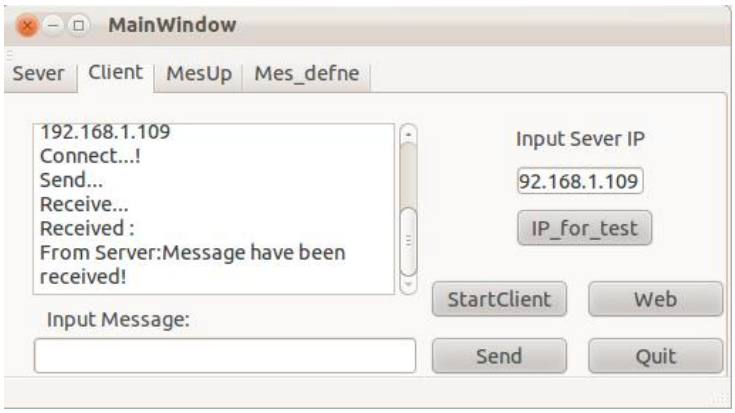

图 6-10 OBU1 运行图

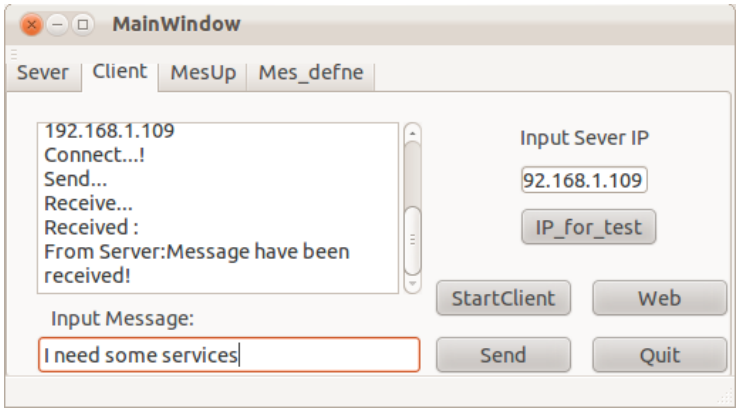

图 6-11 OBU2 运行图

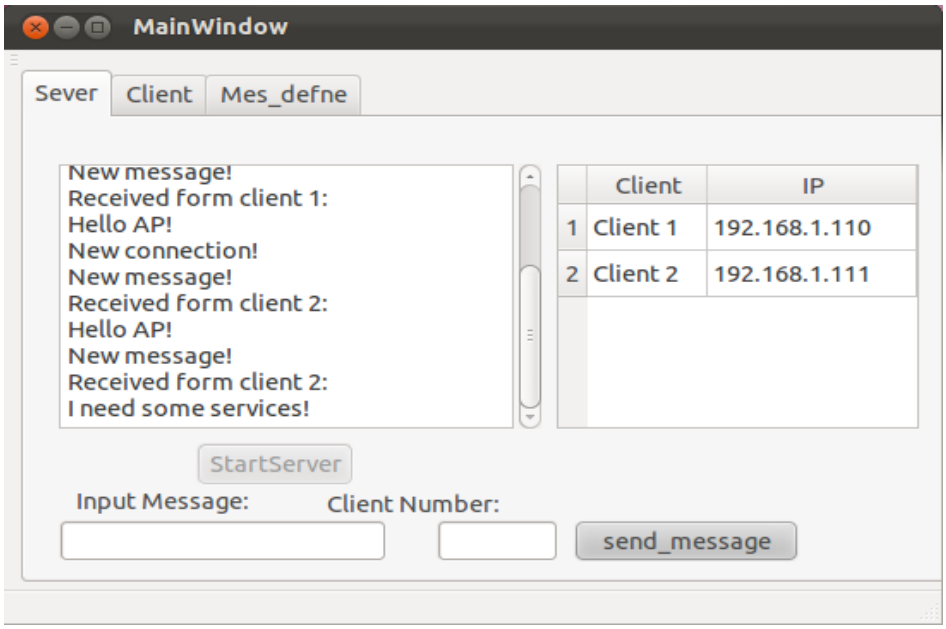

图 6-12 RSU1 运行图

- 车载单元 OBU 消息上传功能测试

我们知道在车联网信息发布系统中，每个 OBU 是信息发布系统的用户的同时，还是该系统的信息采集者。OBU 在运行中，除了定时定量的上传车辆的一些行驶消息外，还会上传如故障、事故等一些交通事件消息。此时，OBU 可以通过软件系统的消息上传功能进行消息的推送。网络中心可以迅速的获得该安全消息，并转发给周围的其他车载单元 OBU。

如图 6-13 所示，OBU 推送的安全消息分成三种事件类型，即交通事故、道路状态以及车辆故障，消息的安全等级分成四种，即 A 特大、B 严重、C 重要以及 D 普通。OBU 还可以选择是否上传自身的位置信息以及为该安全消息增加必要的摘要。

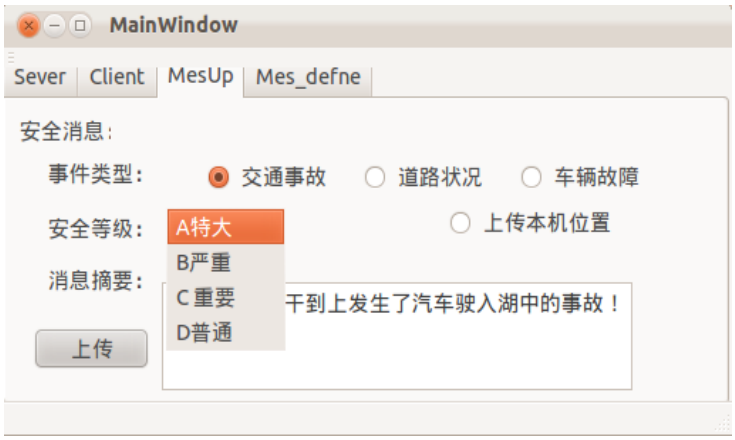

图 6-13 OBU 上传事故消息

- 车载单元 OBU 基于位置的服务功能测试

车载单元 OBU 可以通过客户端系统上的 Web 按钮，打开浏览器功能，如第五章提

到的：OBU 首先基于位置的从 RSU 中获取自身需要的资源的 URL 地址，然后通过定时器的方式循环访问基于位置分发的 Web 资源。

如下图所示：图 6-14 中 OBU1 获得了基于自身位置的安全消息，其中红色小车表示 OBU1 的位置，红色和黄色的标签为周围其他 OBU 上传的安全消息。图 6-15 中，OBU1 通过 RSU 的位置访问到该 RSU 附近的典型娱乐场所，如餐馆、电影院、酒店等信息。

图 6-14 OBU 获取周围的安全消息

图 6-15 OBU 获取周围的餐饮服务信息

6.3.3. 系统性能测试

在本节中，将对面向车联网应用的信息发布系统分别进行车联网网络性能测试、Web 服务性能测试以及数据库访问性能测试。

6.3.3.1. 车联网网络性能测试

本节将利用 Iperf 软件工具测试本文所设计的信息发布系统网络性能。

Iperf 是一个基于 TCP/IP 或者 UDP/IP 协议的网络性能测试工具，它能利用 C/S 通信模式，获取传输网络的吞吐容量、丢包率、传输时延等统计信息。通过 Iperf 获得的统计数据，可以清晰的反应出车联网信息发布系统的通信性能以及网络瓶颈。Iperf 网络性能

工具软件的工作方式和大多数网络性能测试工具的工作方式相同，都是以 Client/Server 方式工作，服务器端和客户端都使用同一程序“iperf”，服务器端使用“-s”选项，而客户端则使用“-c”选项。首先，Iperf 在服务器和客户端之间建立一个测试连接，并利用该连接进行测试参数配置；紧接着它会在服务器和客户端之间不停的来回传输测试数据包，以获得测试网络的性能参数[53]。

本测试的样机是两台 Tiny6410 开发板的 OBU，它们分别位于马路两侧，相距 50 米，将作为 Iperf 的客户端和服务器端启动。利用 Iperf 工具，使两开发板之间进行 UDP 报文通信，通过不断的增加报文数据量，测试他们的 UDP 延迟时间、传输带宽以及丢包率。测试的数据量分别为 1 Mb、5 Mb、10 Mb、20 Mb 和 30 Mb。重复测试两次。

此时，车载单元 OBU 底层驱动均是使用定制后的 MadWifi 驱动。定制后的 MadWifi 驱动符合 WAVE 通信标准。

下面是通过服务器收到的测试报告，得到以下的

第一轮测试数据：

```
[  3] local 192.168.6.14 port 5001 connected with 192.168.6.13 port 46612
[ ID] Interval       Transfer     Bandwidth       Jitter   Lost/Total Datagrams
[  3]  0.0- 9.9 sec  1.19 MBytes  1.00 Mbits/sec  0.283 ms    3/  852 (0.35%)
[  4] local 192.168.6.14 port 5001 connected with 192.168.6.13 port 60888
[ ID] Interval       Transfer     Bandwidth       Jitter   Lost/Total Datagrams
[  4]  0.0-10.0 sec  5.96 MBytes  5.00 Mbits/sec  0.409 ms    0/ 4253 (0%)
[  3] local 192.168.6.14 port 5001 connected with 192.168.6.13 port 37456
[ ID] Interval       Transfer     Bandwidth       Jitter   Lost/Total Datagrams
[  3]  0.0-10.0 sec  11.9 MBytes  10.0 Mbits/sec  0.415 ms    0/ 8505 (0%)
[  4] local 192.168.6.14 port 5001 connected with 192.168.6.13 port 51514
[ ID] Interval       Transfer     Bandwidth       Jitter   Lost/Total Datagrams
[  4]  0.0-10.2 sec  12.9 MBytes  10.7 Mbits/sec  2.158 ms 4709/13940 (34%)
[  3] local 192.168.6.14 port 5001 connected with 192.168.6.13 port 39281
[ ID] Interval       Transfer     Bandwidth       Jitter   Lost/Total Datagrams
   （省略）…
```

第二次测试数据：

```
[ ID] Interval       Transfer     Bandwidth       Jitter   Lost/Total Datagrams
[  4]  0.0-10.0 sec  1.19 MBytes  1.00 Mbits/sec  0.313 ms    0/  852 (0%)
[  3] local 192.168.6.14 port 5001 connected with 192.168.6.13 port 59349
[ ID] Interval       Transfer     Bandwidth       Jitter   Lost/Total Datagrams
[  3]  0.0-10.0 sec  5.96 MBytes  5.00 Mbits/sec  0.188 ms    0/ 4253 (0%)
[  4] local 192.168.6.14 port 5001 connected with 192.168.6.13 port 37254
[ ID] Interval       Transfer     Bandwidth       Jitter   Lost/Total Datagrams
[  4]  0.0-10.0 sec  11.9 MBytes  10.0 Mbits/sec  0.421 ms    0/ 8505 (0%)
[  3] local 192.168.6.14 port 5001 connected with 192.168.6.13 port 45139
   （省略）…
```

从上面的测试数据，我们绘出了 UDP 报文测试图表，UDP 延迟分析图，UDP 丢包速率分析图（见下图 6-15、图 6-16、图 6-17 所示）。

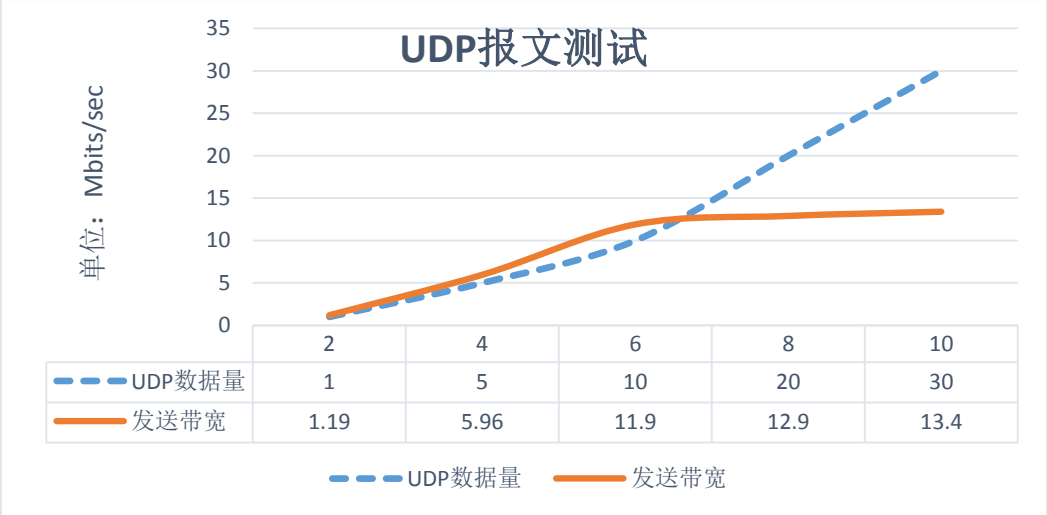

图 6-16 UDP 报文测试图

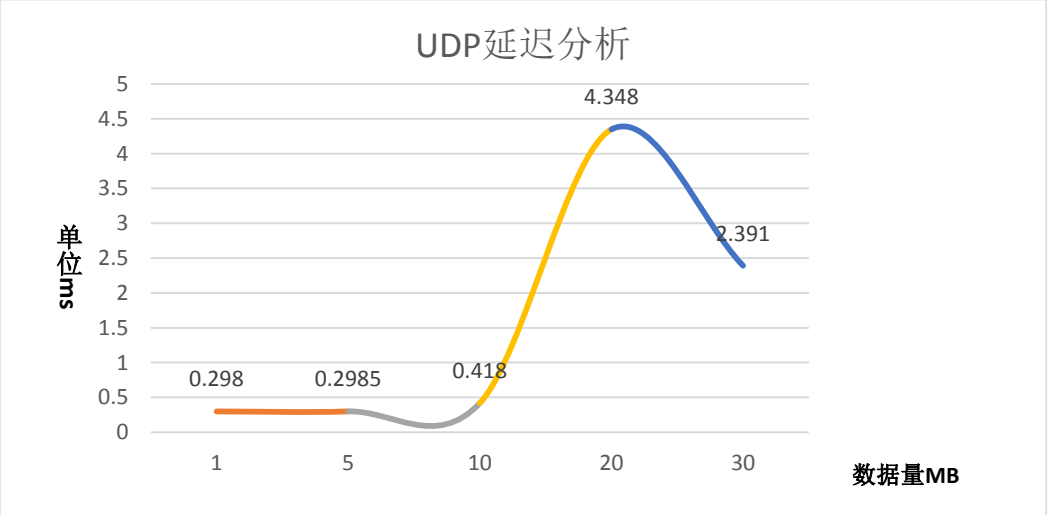

图 6-17 UDP 延迟分析图

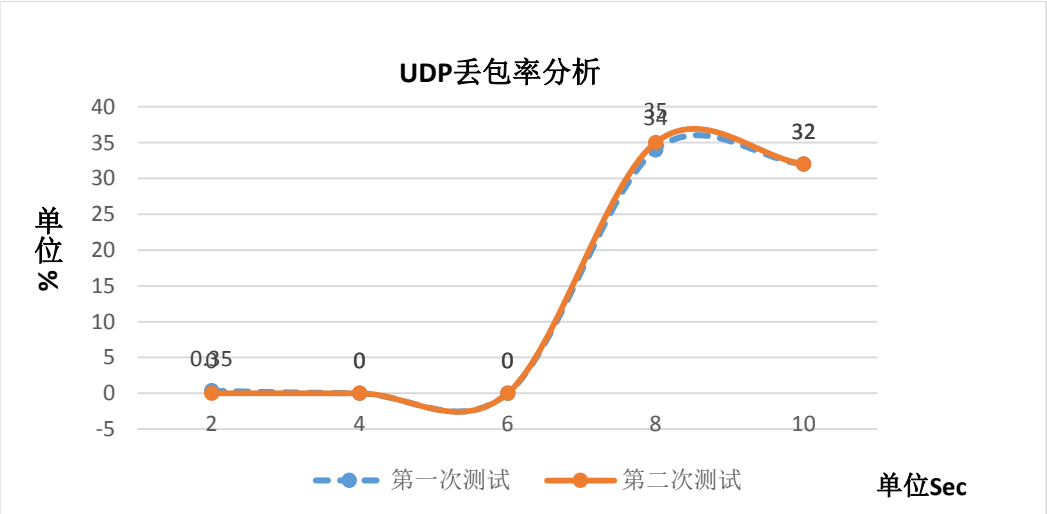

图 6-18 UDP 丢包速率分析图

从这三幅图可以看出，使用了定制的 MadWifi 驱动的车载单元 OBU 之间，点对点

的通信中，传输数据量大概为 13.1 MBytes 的时候有很好的传输质量保证，丢包率基本为 0%，数据传输延迟为 0.1ms 左右。但随着数据传输量的增加，在大于 13.1 MBytes 的数据时，延迟开始明显上升，其实际传输带宽保持在 12Mbits/s 左右不变，丢包率则在 30%左右，通信质量大大降低。

也就是说，本论文设计的面向车联网应用的信息发布系统，在使用了定制的 MadWifi 驱动，在达到符合 WAVE 通信标准后，系统的点对点通信带宽，在理想状态下约为 12Mbits/s（WAVE 协议标准为 3 MBits/s~27 MBits/s）。系统在低于该带宽下才能保证数据的有效传输，高于 12Mbits/s 后数据就会出现明显的丢包和延时现象。

6.3.3.2. Web 服务性能测试

利用 Apache 服务器附带的压力测试工具 ApacheBench 对系统进行 Web 服务器进行性能测试。ApacheBench 工具能根据用户设置的命令，模拟 HTTP 请求，向网络中心的服务器发起并发连接请求[54]。

下面是利用 ApacheBench 模拟 100 个用户向服务器并发 100 的 HTTP 请求（请求 URL 为：http://localhost/cgi-bin/login.py）后，ApacheBench 的输出结果为：

```
…
Concurrency Level:          100
Time taken for tests:       0.353seconds
Complete requests:          100
Total transferred:          72100 bytes
HTML transferred:           56600 bytes
Requests per second:        283.08 [#/sec]
Time per request:           353.329 [ms]
Time per request:           3.533[ms]
Transfer rate:              204.06 [Kbytes/sec] received
Percentage of the requests served within a certain time (ms)
  50%     252
  66%     304
  75%     322
  80%     335
  90%     346
  95%     350
  98%     351
  99%     352
 100%     352 (longest request)
```

从上面的输出结果可知，此次压力测试，共发送 72100 字节的数据，实际传输 HTML 字节 56600，服务器吞吐量为 283.08 请求/秒。用户平均请求等待时间 353.329ms，服务器平均处理时间为 3.533 毫秒，用户每秒接收的数据量为 204.06KB。

根据上面的结果，我们可以得出 HTTP 请求处理时间完成状态图，如图 6-19 所示。根据该图表，我们可以看出 100 个并发请求，50%都在 250ms 内完成，其平均的完成时

间在 300ms 左右。

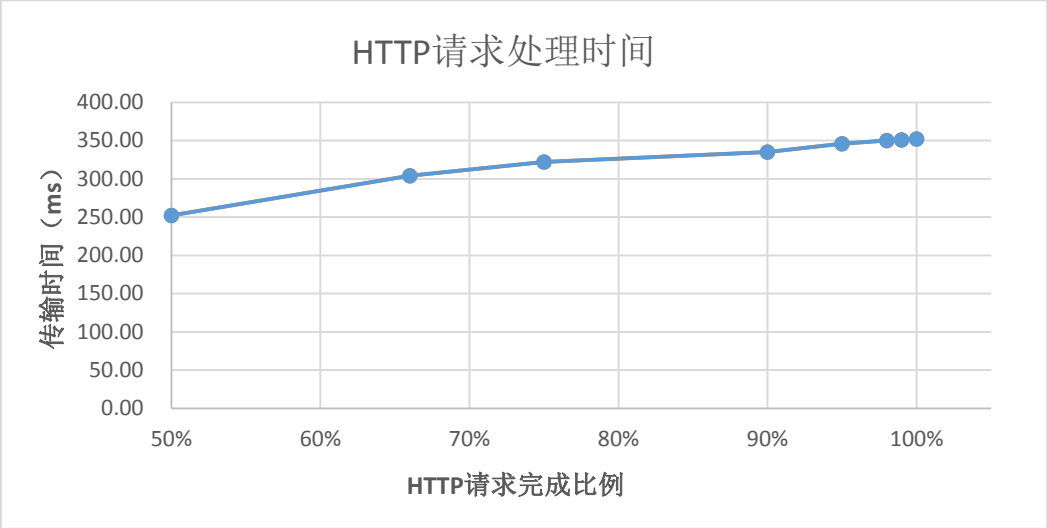

图 6-19 HTTP 请求处理完成时间图

从上述的压力测试可以看出，服务器的平均等待时间约在 300ms，服务器平均吞吐量为 280 请求/秒，能满足车联网信息发布系统 Web 服务的基本要求。按照单个 OBU 的正常访问频率，每台服务器能够支撑 200~500 台 OBU 同时在线访问。

6.3.3.3. 数据库访问性能测试

利用 MySQL 数据库自带的压力测试工具 mysqlslap，与 ApacheBench 工具类似，mysqlslap 可以模拟用户并发登入使用数据库，生成 scheme 以及查询更改数据[55]。为了精确的对 MySQL 数据库进行用户流量上限以及抗压性进行性能测试，本系统将利用 mysqlslap 分别模拟 25,50,75,100,125,150,175,200,225,250,275,300 的系统并发访问量对 MySQL 数据库进行压力测试，测试的命令如下所示：

```
[root@localhost ~]# mysqlslap -uroot -porange
--concurrency=25,50,75,100,125,150,175,200,225,250,275,300
--iterations=10
--auto-generate-sql
--auto-generate-sql-load-type=mixed
--auto-generate-sql-add-autoincrement
--engine=myisam
--number-of-queries=10
#该命令的含义是，利用 mysqlslap，模拟并发量用户为 25-300，系统自动生成 sql 命令对系统表数据进行读写测试，重复访问 10 次，每次执行 10 次查询命令，测试引擎为 myisam
```

重复压力测试十次，得到的 MySQL 性能测试数据如表 6-1 所示：

表 6-2 MySQL 并发测试响应时间表

并发量	平均时间（单位：s）	最短时间(单位：s)	最长时间(单位：s)
25	0.066	0.032	0.288
50	0.081	0.076	0.105
75	0.123	0.119	0.126
100	0.167	0.154	0.206
125	0.199	0.189	0.208
150	0.252	0.238	0.261
175	0.293	0.283	0.307
200	0.326	0.311	0.342
225	0.371	0.36	0.381
250	0.426	0.404	0.5
275	0.465	0.441	0.508
300	0.503	0.476	0.515

根据上表可以画出 MySQL 不同并发量的访问时间散点图，如图 6-20 所示：

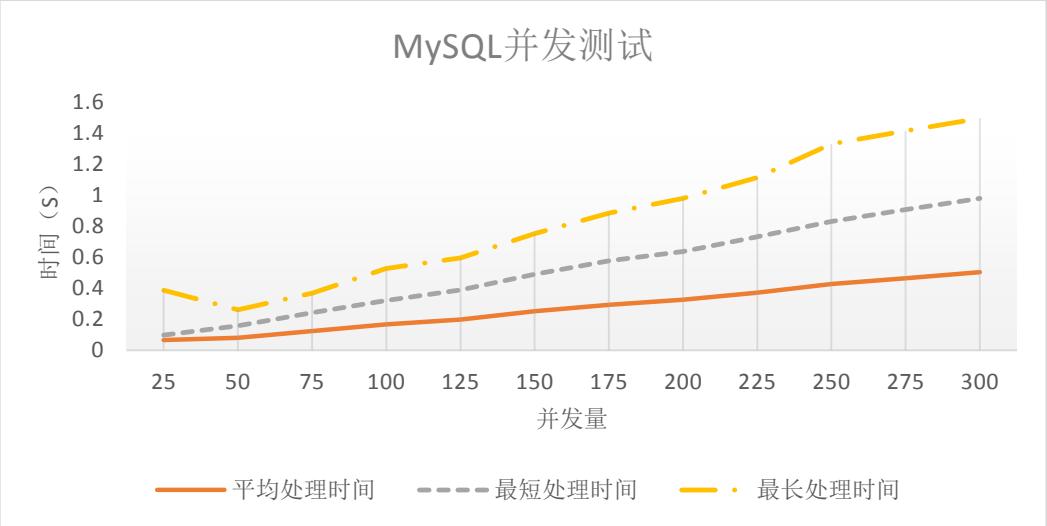

图 6-20 MySQL 并发测试处理时间图

位于网络中心部署的 MySQL 数据库，使用频率最高的莫过于 OBU 上传或存取车载单元监测信息的 OBUData 表。为了测试 MySQL 库中 OBU 频繁访问查询 MySQL 数据库的情况下，OBUData 表表单大小对系统性能的影响。系统在此将利用 mysqlslap 分别模拟 25,50,75,100,125,150,175,200,225,250,275,300 的系统并发访问量对 MySQL 数据库

OBUdata 表的进行压力查询测试，表单的大小分别为 10,100,1000,10000，测试的命令如下所示：

```
[root@localhost ~]# mysqlslap -uroot -porange
--concurrency=25,50,75,100,125,150,175,200,225,250,275,300
--iterations=10–create-schema=”IOV”
–query=”select * from OBUData where Humidity= “56C”
--number-of-queries=10
#该命令的含义是，利用 mysqlslap，模拟并发量用户为 25-300，对数据库 IOV 中的 OBUData 表执行 Select 查询命令，重复访问 10 次，每次执行 10 次查询命令
```

通过测试前，调整 OBUData 表单的大小，得到的 MySQL 性能测试数据如表 6-3 所示：

表 6-4 MySQL 并发测试响应时间表

并发量	查询时间 表单大小 10 （单位：s）	查询时间 表单大小 100 （单位：s）	查询时间 表单大小 1000 （单位：s）	查询时间 表单大小 1000 （单位：s）)
25	0.016	0.017	0.026	0.203
50	0.018	0.021	0.035	0.373
75	0.028	0.03	0.059	0.545
100	0.037	0.041	0.072	0.773
125	0.049	0.052	0.092	1
150	0.059	0.065	0.11	1.217
175	0.071	0.078	0.128	1.349
200	0.081	0.091	0.146	1.459
225	0.094	0.101	0.167	1.521
250	0.107	0.116	0.186	1.666
275	0.116	0.126	0.209	1.8
300	0.13	0.14	0.23	1.885

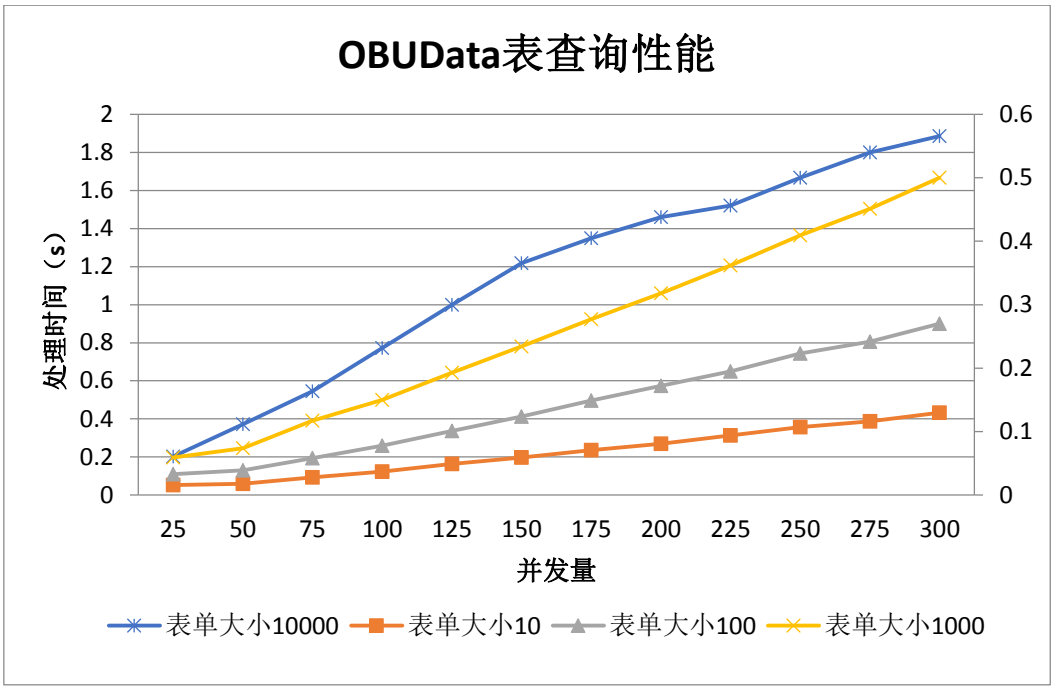

图 6-21 MySQL 查询 OBUdata 表单性能测试图

从 MySQL 并发测试处理时间图可以看到，数据库响应时间随着系统的并发量的增大而增大，其平均访问时间成线性增长模式。由于模拟的每个用户分别进行 10 次读写操作，当并发量为 100-200 时，用户的平均每次读写操作的耗时大概为 15ms-31ms，平均等待时间为 0.15s 到 0.31s。系统的平均等待时间，属于车载通信中对时延的容忍的合理范围。通过压力测试数据可以得出，每台服务器上的 MySQL 数据库能支持 100-200 个 OBU 并发持续的读写数据库操作。

当数据库并发量为 25 时，最长处理时间出现一极端值，从该值可以推测 MySQL 数据库对重复查询做了优化。因此，对某些相似的数据库查询命令可由上层软件系统进行优化，集中进行数据库查询访问，减少系统访问处理时间。

从图 6-21 OBUdata 表单性能测试图可以看出，随着 OBUData 表单的性能随着表单的大小逐步降低，MySQL 处理时间逐步升高。如在并发量 200 的情况下，遍历 OBUData 表查询 1 次消耗的时间，分别为 8.1 ms, 9.1 ms, 14.6 ms, 145.9 ms。因此，表单的数量在维持在 1000 的数量级以内，数据的遍历查询响应时间均非常短，特别是 100-1000 时，查询效率最高，但数量级超过万以后，表单的查询质量将急剧下降。因此，网络中心的数据库中的表单应尽可能保持在 100-1000 范围内，超过 1000 的表单可以根据 OBUID 的哈希值进行分表操作，以提高 MySQL 的并发处理速度，以满足车联网通信系统实时性的要求。

6.4 车联网系统对比分析

在上一节主要对本文所提出的面向车联网应用的信息发布系统进行了功能层面和性能层面的测试。在本节，将会根据上节的功能测试结果和性能测试结果，与传统的车联网系统进行对比，分析本文所设计的车联网信息发布系统的优点。

6.4.1. 系统功能对比

在功能层面上，上节先对系统的网络中心 Web 服务器的用户登入登出、交通服务配置、广告资源配置以及 OBU 监测管理等进行了验证。其次，再对系统的 RSU 和 OBU 之间的 TCP 数据传输、UDP 广播接收、安全信息上传查阅以及基于 OBU 位置的商业服务功能进行了验证。从功能层面看，传统的车联网信息发布系统主要有以下两点缺陷：

- 业务单一

每类系统均只注重单一的服务，如基于射频识别技术的信息发布系统，主要用于高速收费路段的不停车收费业务；基于移动基站提供的 2G/3G 通信方式的信息发布系统，主要用于交通信息查询业务；基于 ZigBee 等低速率传输设备的信息发布系统，主要用于道路交通状况实时监测业务；基于 Wifi 通信的客户端-服务器模式的信息发布系统，主要用于车载单元娱乐信息服务业务。

- 信息传输单向性

传统车联网信息发布系统，消息的流向非常单一，信息是单向传输的，如现在运用最广泛的 ETC 电子收费系统，信息单一的由车辆的用户信息流向收费站，收费站无法传输信息给用户；又如交通信息查询业务，只能由车载用户查询服务器的交通信息，车载用户无法上传自身的消息。

通过 6.3.1 小节的的车联网信息发布系统功能验证，可以清晰的看出，本文设计的基于三层架构的车联网信息发布系统，涵盖了交通道路安全服务、交通数据采集分发服务以及娱乐信息服务三大业务。它除了能提供与传统的信息发布系统相似的电子自助收费服务、道路状况监测服务等之外，还由于多了一层路侧单元和区域服务器（本雏形系统将区域服务器的功能合并到 RSU 中），使其可以方便的为 OBU 提供各种基于位置的服务。如本系统所验证的基于位置的交通安全提醒功能以及基于位置的广告服务功能。并且，本文设计的车联网信息发布系统，信息的流向是双向的，OBU 可以随时上传自己的信息给网络中心或者 RSU，OBU 也可以随时向网络中心或者 RSU 拉取自己需要的信息资源。

由此可见本文所设计的车联网信息发布系统，从功能角度出发，突破了旧式车联网系统功能单一以及信息传输单向性的局限，服务功能具有多样性，且信息为双向传递的。系统中的车载单元 OBU 既是系统信息的产生者也是系统信息的消费者。

6.4.2. 系统性能对比

在性能层面上，本文首先对系统 RSU 和 OBU 端 WAVE 底层通信子模块的网络性能进行了性能测试，其次再对网络中心的 Web 服务性能和 MySQL 访问性能进行了压力测试。从测试结果得出，本论文设计的面向车联网应用的信息发布系统，在使用了符合 WAVE 通信标准定制 MadWifi 驱动后，系统的点对点通信带宽，在理想状态下约为 12Mbits/s，传输时延约为 10 ms。网络中心配置的每台服务器（CPU2.20GHz，1G 内存）均能同时处理 200-500 个 OBU 并发访问，其 Web 服务器的平均等待时间在 100ms 级别，平均吞吐量为 280 请求/秒，MySQL 用户的平均每次读写操作的耗时大约处在 10 ms 级别，平均等待时间为 0.15s 到 0.31s。

下表 6-3 是传统的车联网信息发布系统与本论文设计的车联网信息系统的性能对比，通过该表可以清晰的看出，本文所设计的车联网信息发布系统，在保持低延时、高稳定性、高并发量的前提下，能通过 WAVE 通信标准大大增加的车联网的网络传输质量，为上层交通道路安全服务、交通数据采集分发服务以及娱乐信息服务三大业务提供了可靠的数据传输保证。与其他车联网系统相比，本文设计的系统，网络通信质量与网络中心的高并发处理能力是其最大的优势。

表 6-5 车联网信息发布系统性能对比

通信模式	传输速率	传输距离	系统稳定性	系统并发性能
基于 ZigBee	20～40 kbp	10～100 m	一般	中心节点模型（基于轮询，无需并发）
基于高频 RFID	100kbps	1 m 以内	高	并发能力差
基于 3G	144kbps~384kpbs	1000m 以上	低	并发性能高（基于基站）
基于 WAVE	3Mbps~12Mpbs	200m 以内	高	并发性能高

6.5 本章小结

本章主要根据第四、五章的信息发布系统的设计和实现，从功能以及性能出发，对整个面向车联网应用的信息发布系统进行基本的测试。

在动态仿真测试方面，首先，利用 ITETRIS 系统，对以 WAVE 作为通信标准的车联网业务的可行性进行了验证。

在功能测试方面，首先，从系统外部通过浏览器访问网络中心 Web 服务器，验证其功能的可用性。其次，再通过 RSU 和两个 OBU 进行通信访问，测试系统 RSU 和 OBU 客户端软件的功能可用性。

在性能测试方面，首先，利用 Iperf 工具测试 OBU 的网络性能。Iperf 通过 OBU 之间进行不断的增加报文数据量的 UDP 报文通信，来检测网络的 UDP 延迟时间，传输带宽以及丢包率；其次，利用 ApacheBench 和 mysqlslap 压力测试工具，模拟用户高并发量对系统进行压力访问。

最后，在通过与旧式的车联网系统进行对比，分析本文设计的面向车联网应用的信息发布系统的优势所在。通过测试数据，可以看出本文提出的面向车联网应用的信息发布系统，网络传输能力足以满足大数据时代高信息容量传输服务要求，并且单个服务器性能稳定高效，能同时满足并发量为 200-500 的用户持续的访问。

总结与展望

本文在基于目前车联网研究领域中，车联网信息系统普通存在通信质量差、传输速率低、业务单一、系统并发访问能力不足的现状下，立足于智能交通系统中车联网应用三大服务需求，即交通道路安全服务、交通数据采集分发服务以及娱乐信息服务，并根据车联网通信环境快速多变的特点，以及车联网信息分发的特征和需求，研究设计出一种新型的面向车联网应用——适用于车联网业务环境的信息发布系统。

本文的研究工作主要围绕着车联网应用的服务需求展开，其中包括 WAVE 协议栈研究、LAMP 架构模型研究、车联网信息发布系统的总体结构与功能研究这三大层面。并在此研究成果的基础上，结合车联网的服务需求，提出了网络中心-区域服务器路侧单元-车载单元的三层结构的车联网设计方案，创造性地融合 WAVE 技术和 LAMP 技术来搭建车联网信息发布系统。经测试实验证明，与传统的车联网系统相比，本文设计的系统在功能上和性能上具有服务多样、高稳定性、高并发量以及车联网通信环境的高适用性等优点。

本文主要针对车联网信息发布系统与 WAVE 协议栈、LAMP 架构等技术的结合做深入的研究探讨。所设计的系统作为车联网信息发布系统的雏形系统，对于实现车联网应用真正的商业化运营真正商业化运营具有一定的实践价值。本文所设计的车联网雏形系统主要取得如下的成果：

1）摒弃了传统车联网信息发布系统采用服务器-客户端（C/S）架构，采用了针对移动车载环境的网络中心-区域服务器路侧单元-车载单元的三层架构体系，使车联网信息发布系统更贴合实际的车载移动环境，增加了系统进行信息分发的效率和精确度，并使得系统能精确有效的实现各种基于 OBU 位置的服务业务。

2）利用 WAVE 协议栈，作为车联网信息发布系统移动设备的无线通信协议，解决了车联网信息发布系统移动设备在快速移动中通信质量差的问题，保证了信息发布系统信息分发的可靠性与稳定性。

3）利用 LAMP 架构部署网络中心的动态网络应用服务平台,大大增加了网络中心的并发性能与可维护可扩展性能，解决了后续车联网信息发布系统商业化后，大数据量的 OBU 对网络中心的负担。

但是本文所提出的信息发布系统，由于作者本人水平的限制，系统的实现并不是十分完善，系统的稳定性、可靠性仍然需要进一步修正。由于系统没有真正部署到高速公路等原因，系统的功能内容丰富度仍然未达到商用级别的要求。本文所提出的面向车联

网应用的信息发布系统，只是一个雏形系统，离真正能商用的应用级系统还有很大的差距，系统很多方面还需要不断的完善，主要有以下几点：

1）目前，并没有直接在硬件上实现 WAVE 协议栈，大多数均是从软件上模拟 WAVE 通信协议，因此后续需要从硬件上开发出基于 WAVE 协议栈的无线网卡，并把它运用到本系统中，进一步提高车联网无线通信质量。且由于本系统对动态移动的测试只通过仿真平台进行可行性验证，实物雏形系统的功能验证和性能测试均只在道路上进行静态的验证，后续可以进一步与城市交管中心配合，在道路上进行动态移动下的系统功能和性能测试。

2）由于本系统只是雏形系统，车载单元 OBU 功能相对单一，只有访问 Web 广告，接收和上传的安全警报等简单功能。因此，系统的后续需要基于本文提出的交通道路安全服务，交通数据采集分发服务以及娱乐信息服务需求进行功能升级优化。此外，本系统需要与现有的智能交通系统，如电子 ETC 收费系统等系统无缝对接，以最大程度的整合交通资源。

3）由于车联网信息发布系统，涉及了许多车联网用户的敏感信息，如车辆的牌照、车辆的位置、收费账户等。因此，本系统还可以在后续开发基于敏感信息的加密传输功能。

4）本系统真正商用后，将涉及海量交通数据的分发处理工作。基于海量数据的的分布式存储算法、相关的以海量数据为基础的数据挖掘算法和交通控制算法将会是下一步系统的研究重点，这将会给车联网的用户和管理者带来极大的收益。

总之，本文为车联网信息发布系统的发展和应用提供了一定的实用参考价值，未来可以从上述的四个方面对车联网信息发布系统进行完善和升级，使之能真正的应用到智能交通领域中，为智能交通系统的发展起到促进作用。

参考文献

[1] Gräfling S., Mahonen P., Riihijärvi, J.. Performance Evaluation of IEEE 1609 WAVE and IEEE 802.11p for Vehicular Communications[C]. Ubiquitous and Future Networks (ICUFN), 16-18, June 2010: 344-348

[2] 曹颖荣, 林小玲. IEEE 802.11p 无线车载自组网络协议的性能分析与模拟[J]. 仪表技术, 2011-2: 19-22

[3] Hikita T., Kasai T., YoshiokaA. Integrated simulator platform for evaluation of vehicular communication applications[C]. Vehicular Electronics and Safety, ICVES 2008. IEEE International Conference, 22-24 Sept. 2008: 323-327

[4] 宋俊德. 浅谈物联网的现状和未来[J]. 移动通信, 2010 年第 15 期: 8-10

[5] 广州市物联网交通应用示范建议[S]. 物联网信息技术与产业化省部院产学研联盟, 2010

[6] 杨瑞. 工程车辆联网系统及软件平台设计[D]. 浙江大学, 2012.05

[7] 常促宇, 向勇, 史美林. 车载自组网的现状与发展[J]. 通信学报,2007 第 28 卷第 11 期：116-124

[8] 刘富强, 单联海. 车载移动异构无线网络架构及关键技术研究[J]. 中兴通讯技术, 2010 Vol.16: 3-5

[9] 加斯特. 802.11 无线网络权威指南(译) [M]. 东南大学出版社,2006

[10] Mattbew S.Gast. 802.11Wirelss Networks：The Definitive Guide[M]. O'Reilly,2007.12

[11] IEEE Std 802.11p, IEEE Standard for Information technology—Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications-Amendment 6: Wireless Access in Vehicular Environments [S]. IEEE, 2010

[12] IEEE 1609-Family of Standards for Wireless Access in Vehicular Environments (WAVE) [EB/OL]. http://www.standards.its.dot.gov/fact_sheet.asp?f=80

[13] IEEE Std 1609.0-2012,IEEE Draft Guide for Wireless Access in Vehicular Environments (WAVE)-Architecture. IEEE P1609.0/D5[S]. IEEE, 2012

[14] 艾利锋, 刘春煌, 蒋荟. Web Services 在行车安全综合监控系统中的应用研究[J]. 铁路计算机应用, 2006, 15(4): 4-7

[15] Wikipedia. Web2.0[Z/OL]. http://zh.wikipedia.org/wiki/Web2.0

[16] Schroth C., Janner T.. Web 2.0 and SOA: Converging Concepts Enabling the Internet of Services[J]. IT Professional, vol.9, no.3, May-June 2007: 36-41

[17] Wikipedia. LAMP[Z/OL]. http://zh.wikipedia.org/wiki/LAMP

[18] 艾利锋. Web Services 技术在行车安全综合监控系统整合中应用的研究[D]. 铁道科学研究院, 2006

[19] 杨涛, 刘锦德. Web Services 技术综述一种面向服务的分布式计算模式[J]. 计算机应用, 2004, 24(8): 1-4.

[20] Wikipedia. Intelligent Transport System[EB/OL]. http://en.wikipedia.org/wiki/Intelligent_transportation_system

[21] 迟铁军, 高鹏. 国外智能交通系统发展状况分析及对我国的启示[J]. 黑龙江交通科技, 2009 年第 2 期: 111-114

[22] Jong-Moon Chung. Time Coordinated V2I Communications and Handover for WAVE Networks[J]. Selected Areas in Communications,March 2011 : 545-558

[23] 刘富强, 项雪琴, 邱冬. 车载通信 DSRC 技术和通信机制研究[J]. 上海汽车, 2007. 08: 35-38

[24] Minglu Li. Metropolitan VANET: Services on the Road[C]. High Performance Computing and Communications, 2008. HPCC '08. 10th IEEE International Conference, vol., 25-27 Sept. 2008: 17

[25] 杨洋, 黄征, 金涛. 基于 ETC 和 MTC 高速公路组合式联网收费系统设计[P]. 现代电子技术, 2007, 30(21): 144-147

[26] 彭文兵. 交通信息发布系统[P]. CN 102542822 A, 2012.07.04

[27] 姚笛, 刘佳. 基于 zigbee 无线通信的轨道交通线路杂散电流实验监控系统设计[J]. 科技创业家, 2013, (4): 2

[28] 周良文. 基于 WiFi 网络的集成信息发布系统[P]. CN: 102447653A, 2012.05.09

[29] Bilstrup K., Uhlemann E., Strom, E.G, et al. Evaluation of the IEEE 802.11p MAC method for Vehicle-to-Vehicle Communication[C]. Vehicular Technology Conference, 2008. VTC 2008-Fall. IEEE 68th : 1-5

[30] Yi Wang, Akram Ahmed, Bhaskar Krishnamachari. IEEE 802.11p Performance Evaluation and Protocol Enhancement[C]. 2008 IEEE International Conference on Vehicular Electronics and SafetyColumbus, OH, USA. September 22-24, 2008: 317-322

[31] 杜英田. 基于 IEEE802.11p/1609 协议的智能交通无线车载通信协议优化研究[D].

北京邮电大学, 2011

[32] 唐波. WAVE 架构及相关协议设计与实现[D]. 华东理工大学,2009

[33] L. Stibor, Y. Zang, H-J. Reumermann. Evaluation of communication distance of broadcast messages in a vehicular ad-hoc network using IEEE 802.11p[C]. Proc. IEEE Wireless Communications and Networking Conf., Hong Kong, China, Mar. 2007: 254-257.

[34] Campolo C., Molinaro A.. On vehicle-to-roadside communications in 802.11p/ WAVE VANETs[C]. Wireless Communications and Networking Conference (WCNC), 2011 IEEE, 28-31 March 2011: 1010-1015

[35] Karagiannis G.. Vehicular Networking: A Survey and Tutorial on Requirements, Architectures, Challenges, Standards and Solutions[J]. Communications Surveys & Tutorials, IEEE,Fourth Quarter 2011: 584-616

[36] Chien-Chun, Huang-Fu, Chi-Ling Chen. Yi-Bing Lin Location Tracking for WAVE Unicast Service[C]. Vehicular Technology Conference (VTC 2010-Spring), 2010 IEEE 71st, 16-19 May 2010: 1-5

[37] S. Eichler. Performance evaluation of the IEEE 802.11p WAVE communication standard[C]. Proc. IEEE Vehicular Technology Conf., Baltimore, MD, US, Oct. 2007: 2199-2203

[38] Ghosh S., Kundu A., Jana D. Implementation challenges of time synchronization in vehicular networks[C]. Recent Advances in Intelligent Computational Systems (RAICS), 2011 IEEE, 22-24 Sept. 2011: 575-580

[39] IEEE Std 1609. 4-2010, IEEE Standard for Wireless Access in Vehicular Environments (WAVE)-Multi-channel Operation[S]. IEEE, 2011

[40] IEEE Std 1609. 3-2010, IEEE Standard for Wireless Access in Vehicular Environments (WAVE)-Networking Services[S]. IEEE, 2010.

[41] IEEE Std 1609.2-2006, IEEE Trial- Use Standard for Wireless Access in Vehicular Environments—Security Services for Applications andManagement Messages[S]. IEEE,2006

[42] IEEE Std 1609.1-2006,IEEE Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE)—Resource Manager. IEEE Vehicular Technology Society (VTS) [S]. IEEE, October 2006

[43] 邱团准. 基于 SOA 的车联网应用系统设计与实现[D]. 华南理工大学, 2013

[44] Hartenstein H., Laberteaux K.P.. A tutorial survey on vehicular ad hoc networks [J]. Communications Magazine, IEEE, vol.46, no.6, June 2008: 164-171,

[45] Wikipedia. MySQL[Z/OL]. http://zh.wikipedia.org/wiki/MySQL

[46] Wikipedia. Debian[Z/OL]. http://zh.wikipedia.org/wiki/ DEBIAN

[47] Chrisian Benvenuti. Understanding Linux Network Internals[M], O'Reilly, 2005

[48] Daniel Q.Chen. 面向服务的行业解决方案[M]. 电子工业出版社, 2006

[49] 顾振飞. 车联网系统架构及其关键技术研究[D]. 南京邮电大学, 2012

[50] 魏永明, 耿岳, 钟书毅. LINUX 设备驱动程序(译)[M]. 中国电力出版社, 2006

[51] 韦东山. 嵌入式 Linux 应用开发[M]. 人民邮电出版社, 2003

[52] 百度文库. Madwifi 详解[EB/0L]. http://wenku.baidu.com/view/541d8307e87197

[53] Wikipedia. Iperf[Z/OL]. http://en.wikipedia.org/wiki/Iperf

[54] Wikipedia. ApacheBench[Z/OL]. http://en.wikipedia.org/wiki/ApacheBench

[55] MYSQL. Mysqlslap[Z/OL]. http://dev.mysql.com/doc/refman/5.1/en/mysqlslap.html

攻读硕士学位期间取得的研究成果

在硕士研究生期间，本人以第一学生作者身份申请国家发明专利一篇：

余荣，谢胜利，张洁柯等. 一种基于车联网的信息发布系统

（发明专利，已受理，申请号：201210588415.6）

致谢

蓦然回首，三年的硕士生涯转瞬即过。三年来，有过憧憬，也有过茫然。已经记不清经历过多少困难与挑战，记不清经历过多少次通宵达旦阅读文献、部署系统、调试程序。如果要用一句话来总结我的研究生生涯的话，我想“一分耕耘，一分收获”再贴切不过了。忘不了老师们的谆谆教诲，忘不了同学们的深厚友谊，更忘不了家人无微不至的关怀。是你们，让我从一个懵懵懂懂的少年蜕变成如今些许成熟些许稳重的我。三年的研究生生活即将结束，我衷心感谢所有帮助过和关心过我的人们。

首先感谢我的研究生导师傅予力教授，傅老师是一位德才兼备的老师，颇受学生的爱戴，我很幸运能够在傅老师的指导下进行研究生阶段的学习。傅老师渊博的知识、严谨的治学态度、低调的处事作风，都深深地影响了我，他不仅教给我专业知识，还教会我许多做人的道理，这些都将让我受用终生，在此，衷心地感谢傅老师。

其次，感谢实验室的余老师，他是我们项目组的指导老师，他认真负责的工作态度、专业的解决问题的能力，以及对学生的耐心指导，让我在做项目的过程中收获颇丰。余老师，感谢您让我参与到“车联网”的研究项目当中，让我第一次接触到了当时刚刚兴起的 WAVE 协议栈，踏入了智能交通系统这片广阔的天地中。

最后，感谢实验室的谢胜利教授和其他的各位老师，他们为我的学习和发展提供了一个很好的平台。感谢实验室的众多师兄师姐们，他们带领我入门，为我提供许多指导与帮助。

在论文写作期间，参考了许多相关书籍和文献，在此向这些作者表示谢意。此外，整个车联网系统雏形的搭建，还要感谢邱团准、周芳芳、陈钦波们一起共同努力。

感谢实验室的同学们（城少、钦波、老蒋、阿波、段哥、妍蓉、小强、文龙、启辰、奎文、芳芳），在学习上，大家营造了良好的学习环境和学术氛围，在生活上，大家相互关心相互帮助，正是你们，才使我的三年研究生生活收获丰富、充满精彩。感谢实验室的师弟师妹们，有你们的努力付出，才有我们的功成身退。

特别感谢我的家人，感谢你们的支持、理解与包容，正是这无微不至的关怀

和鼓舞，给予我无比的动力与力量，让我顺利的度过三年的研究生生涯！

“路漫漫其修远兮，吾将上下而求索”，研究生阶段的结束只是新生活的开始，最后，衷心祝愿大家身体健康，工作顺利，心想事成！