**Computer Science and Application** 计算机科学与应用, 2015, 5, 204-211
Published Online June 2015 in Hans. http://www.hanspub.org/journal/csa
http://dx.doi.org/10.12677/csa.2015.56026

Hans汉斯

# Remote Monitoring System for Smart Domestic Floor Heating System Based on Node.js

**Ziran Wu[1], Duo Li[1], Zhenghui Yang[1], Hua Ye[1,2]**

[1]School of Automation, Southeast University, Nanjing Jiangsu
[2]Key Laboratory of Measurement and Control of CSE of Ministry of Education, Southeast University, Nanjing Jiangsu
Email: zhineng@seu.edu.cn

Received: Jun. 5th, 2015; accepted: Jun. 22nd, 2015; published: Jun. 25th, 2015



## Abstract

**A remote control system is a very common subsystem of a smart homer system. The architecture of the system is usually based on C/S mode or B/S mode, which sets limits on the functionality and performance and has to ignore some user demands. This article is going to design a remote control system based on C/S and B/S mixed mode using Node.js, which is based on a smart domestic floor heating program. The method of building a Node.js server will be described in details in this article, and the tools and methods to design a web page and a client software will be introduced briefly.**

## Keywords

**Node.js, C/S + B/S Mixed Mode**

# 基于Node.js的家庭智能地暖远程监控系统

**吴子然[1]，李　多[1]，杨争辉[1]，叶　桦[1,2]**

[1]东南大学自动化学院，江苏　南京
[2]东南大学复杂工程系统测量与控制教育部重点实验室，江苏　南京
Email: zhineng@seu.edu.cn

收稿日期：2015年6月5日；录用日期：2015年6月22日；发布日期：2015年6月25日

## 摘　要

**智能家居系统中往往会有远程控制部分，这些远程监控系统通常是基于单一C/S或B/S模式实现的，其功能和性能受到单一模式的限制，无法最大化满足用户需求。本文以家庭智能地暖控制系统为项目背景，基于现在流行的Node.js设计了一个C/S和B/S混合模式的远程监控系统，重点描述基于Node.js构架服务器的方法，并介绍其中网页与客户端部分的设计工具与方案。**

## 关键词

**Node.js，C/S与B/S混合模式**

## 1. 引言

一般的智能家居远程监控系统是以 C/S 模式(客户机/服务器模式)或 B/S 模式(浏览器/服务器模式)实现的，两种模式各有优缺点。

C/S 模式运行速度快，能在客户机上完成更多运算，减轻服务器负担。在家庭智能地暖系统中，客户端在移动平台上运行。问题在于，当今流行的移动操作系统很多，要适应不同用户的需要，就需开发不同版本的客户端，这大大增加了软件的开发和维护成本。

B/S 模式适用性更高，客户机只要运行浏览器，即可通过网络访问服务器来获取 web 应用。Web 页面的开发维护更简单，只需在服务器端完成，软件成本低。但相对于 B/S 模式，B/S 运行过需要传输更多数据，其运行效率较低，功能也有所限制[1]。

不同的用户在不同的情形下，可能有不同的使用需要。比如一个用户在使用电脑时，他可能更希望直接打开浏览器对系统进行远程监控；而在回家路上时，他更希望使用自己手机上的软件实现远程监控。本文将结合上述两种模式，基于 Node.js 开发 C/S 和 B/S 混合模式的远程监控系统，此系统服务于家庭智能地暖系统。其中的客户端软件将使用 QML 进行开发，web 页面则用 HTML 实现。

## 2. 系统设计

家庭智能地暖系统有一个嵌入式控制终端，负责管理本地每个区域的温控器，终端与温控器间的通信使用 Zigbee 技术。嵌入式终端通过自身的网络接口与局域网中的远程监控服务器建立网络连接，与服务器进行数据交互。远程监控服务器接入互联网中，互联网中的客户机可通过客户端或浏览器登录服务器，并通过服务器对本地的嵌入式控制终端实现监控。家庭智能地暖系统的结构如图 1 所示。

## 3. 服务器设计

Node.js(简称 Node)是一个建立在 Chrome 的 V8 JavaScript 引擎的服务器平台。Node 使用单线程、事件驱动、非阻塞 I/O 来处理请求。相对于 ASP.NET、PHP 等常见的服务器实现工具，Node 更加轻量，数据处理效率更高[2]。Node 不需要依赖额外的服务器软件或运行库，对于独立且小型的智能地暖系统来说，Node 无疑是更好的选择。

Node 有良好的开发生态和丰富的功能模块，给开发带来很多便利。Express.js (简称 Express)就是一个被广泛使用的 HTTP 服务器框架。利用 Express 可快速构建 HTTP 网站，且更好地实现 URL 路由管理，

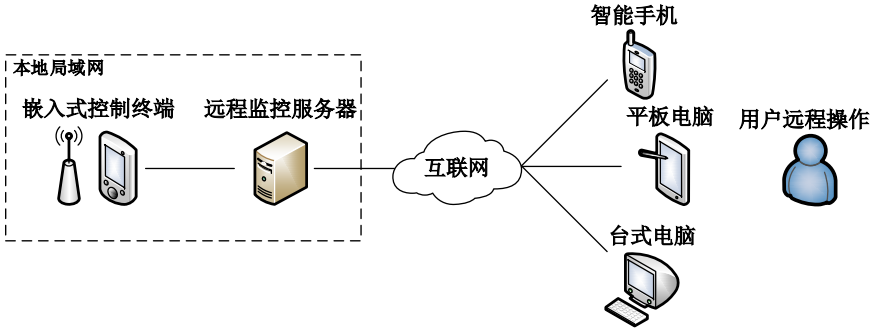


**Figure 1.** The structure of the smart domestic floor heating system
**图 1.** 家庭智能地暖系统结构

提高代码重用率和开发效率[3]。

### 3.1. 服务器结构

本地局域网中，服务器与嵌入式控制终端(下面简称终端)建立 TCP 网络连接，向终端采集温控器的数据信息。服务器同时接入互联网中，处在远端的客户机可与之进行连接。客户机成功登录服务器后，可查询服务器上保存的温控器数据；并可发出控制指令，由服务器转达给终端以实现控制操作。服务器结构如图 2 所示。

服务器主要由五个部分组成：

1) 基于 Express 框架的 HTTP 服务器。客户机上的客户端或 web 页面将从 HTTP 服务器获取相关的应用服务。HTTP 服务器负责处理客户机请求，并作出响应。客户端和 web 页面的请求 URL 路径由 APP 路由和 WEB 路由分别进行管理；

2) 应用服务接口。它为 HTTP 服务器提供应用服务的接口调用，负责处理由 HTTP 服务器获取的客户机请求指令，实现用户管理、信息监控等业务逻辑；

3) 嵌入式接口。该接口负责与局域网中的终端建立网络连接，与之进行数据交互。它负责把从终端采集的数据直接记录到数据库中，并能封装应用服务接口发出的控制指令进而发送给终端；

4) 本地数据管理模块。它对用户、设备进行记录和管理，并对外提供数据管理接口；

5) 文件系统。文件系统保存数据库文件、web 页面所需要的 HTML 页面及 JavaScript 脚本等文件。

### 3.2. 与客户机通信

服务器与客户机间使用 HTTP 协议(超文本传输协议)进行通信。HTTP 是互联网中使用最广泛的网络通信协议，一般是被用来传输 HTML 页面或实现页面与服务器间的通信。然而在本文中，客户端与服务器的通信也应用 HTTP 协议进行通信。

现在，无论是客户端还是 web 应用，客户机与服务器之间的通信都是基于 HTTP 协议的，在服务器看来二者几乎没有任何区别。开发过程中，两个模式下的通信实现有大量可重用的代码；运行时，服务器能以几乎相同的方式处理客户端和 web 页面的请求。

HTTP 通信中主要使用 GET 和 POST 方法。GET 用于从服务器获取数据，POST 则是向服务器发送数据。无论使用任何方法，HTTP 报文中都会含有一个唯一的 cookie，每个 cookie 对应于一个 session (会话)。服务器使用 session 保存与客户机通信的相关信息，并利用会话技术，对以不同模式下的客户机通信进行统一管理。

举个例子，当一个用户成功登录后，服务器从数据库中将该用户的权限等信息记录到 session 中。接

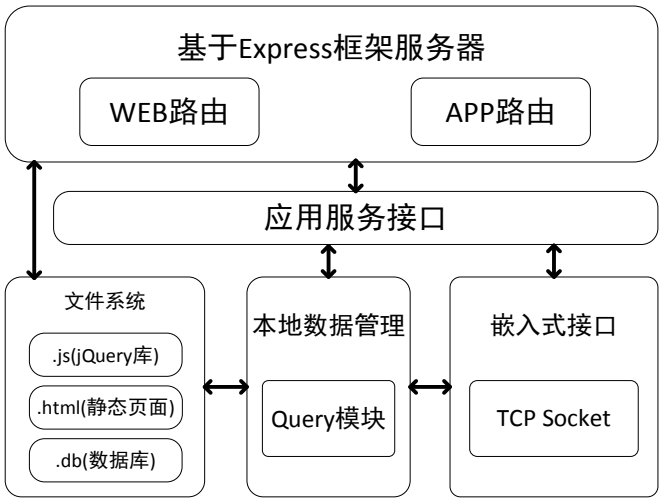


**Figure 2.** The structure of the server
**图 2.** 服务器软件结构图

着，服务器每接受到用户请求，都会从 session 数据中获取该用户的权限，判断请求合法性并进行处理。如果不使用 session 技术，每次信息匹配都要进行数据库读写，极大降低处理效率。除此以外，利用 session 技术还可实现用户登录管理(避免帐号重复登录)、会话超时、控制权分配(避免操作冲突，详见 3.6)等功能。

虽然，在 C/S 模式与 B/S 模式下都使用 HTTP 通信，意图将二者的通信实现的差异减到最小，但二者还是有差别的，具体体现在 web 应用拥有更多的管理配置功能(在 web 页面容易进行较繁琐的配置操作，移动客户端更适合实现简易监控功能，详见 3.6)。为此二者利用 URL(统一资源定位器)进行区分。B/S 模式的 URL 路径是以“/”为前缀，而 C/S 模式则是以“/app”为前缀。基本 URL 结构如图 3 所示。

### 3.3. 与嵌入式控制终端通信

服务器与局域网中的终端建立 TCP 网络通信。利用 Node 中的 net 模块，可以轻松 TCP 实现 TCP 网络通信。

实际上，终端上运行 TCP Server，而服务器则使用 TCP Socket 与终端上的 TCP Server 进行通信连接。嵌入式控制终端是整个家庭智能地暖系统的核心部分，即便远程监控服务器崩溃了，嵌入式控制终端也要确保能够正常运行。况且，地暖系统运行信息数据都需要经过终端处理。所以从逻辑上看，嵌入式控制终端在整个智家庭能地暖系统的局域网中应当以主服务器的形式存在。

由于篇幅有限，服务器与嵌入式控制终端通信的协议不在这里进行详细说明。要说明的是，协议的解析与封装全部由嵌入式接口实现。嵌入式接口所有的控制终端指令封装成函数，对外提供应用接口调用。而如何使用这些应用调用，则是有应用服务接口决定。应用服务器接口属于业务逻辑层，嵌入式接口与本地数据库模块则同属数据访问层，服务于业务逻辑层。

### 3.4. 数据库设计

服务器数据库使用的是 MongoDB，它是一种非关系型的开源数据库，拥有快速的数据库读写速度[4]。并使用基于 Node 的 Mongoose 模块作为 MongoDB 数据库管理接口，以提高开发效率。

MongoDB 的数据模型可以 JSON 格式表示，数据中的每条记录以文档形式保存，文档中可以嵌套其他文档或数组，每个文档则基于特定数据模型实现。为了方便说明，简单列出两个主要的数据模型。

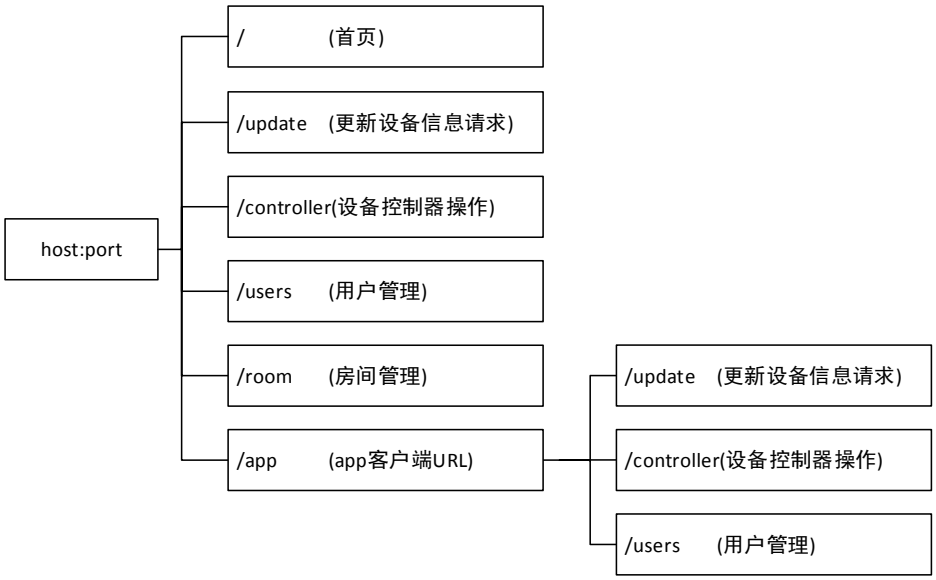


**Figure 3.** URL routing
**图 3.** URL 路由结构

设备模型：

```
{   deviceID: Number,                    // 设备 ID
    type: String,                        // 设备类型
    status: {                            // 状态
        parameter:[],                    // 运行参数
    …                  }    }
```

房间模型：

```
{   roomID: Number,                      // 房间 ID
    name: String,                        // 房间名
    devices: [deviceSchema]      }       // 设备列表，保存设备模型文档
```

每个房间模型中有一个“devices”的键，它的键值是一个数组，数组中保存设备模型的对象。这在逻辑上很好理解，系统中有多个房间，房间在数据库中以文档形式表现；而每个房间里又有多个设备，这些设备以设备模型的形式存在，其对象被保存在每个房间文档的 devices 数组中。

相比关系型数据库，MongoDB 中的数据结构更为紧凑灵活，逻辑上更为简单直接，与 Javascript 的对象非常相似，在 Node 服务器中使用显得非常合适。

设备模型保存设备的型号、运行信息等内容。房间模型保存房间号、设备组等内容。此外还有用户模型，用来保存用户信息。

## 3.5. 服务器安全

服务器将接入互联网，如果服务器遭入侵，系统就会被非法控制，可能造成不良后果，因此服务器安全问题不可忽视。针对以下安全隐患采取相应的防范措施。

1) 用户信息安全。在登录页面和用户创建页面采用 md5 对用户密码进行编码，防止用户密码明文在传输途中被截取，服务器端用户密码保存的是 md5 的编码结果而明文，以防止服务器端的密码泄露；

2) CSRF (Cross-site request forgery，跨站请求伪造)。被攻击者盗用身份(如管理员权限)，对服务器数据非法修改。要防御这类攻击，首先要将能够修改数据请求都改为使用 POST (因为伪造 GET 请求只需要简单的 URL 路径连接，而 POST 则需要借助 JavaScript 实现)方式。服务器给每个成功登录的客户机分配一个数字令牌，当客户机发出指令请求时，需要将这个数字令牌回发，服务器对其进行验证。

除此以外，还有对 DoS 攻击、XSS 攻击等多种安全隐患的预防，在此不一一列举。

### 3.6. 管理策略

服务器需要按照一个良好有效的管理策略对用户和设备进行管理，以正确处理运行过程中可能发生的冲突。下面是本文所遵循管理策略：

1) 系统中有且只有一个管理员账号，该帐号拥有最高权限，可以登记房间设备、增减用户、更改用户权限等；

2) 同一帐号不允许在同一时刻重复登录；

3) 用户权限分为管理员权限(只有一个)、监控权限、查询权限(只能查询不能控制)；

4) 首先登录系统的拥有监控权限的用户会获取系统控制权(类似于互斥锁)，在该用户登出前，其他用户无法获取系统控制权，只能够查询系统运行信息；

根据所制定的管理策略，服务器可以限定用户权限，让服务器的房间设备配置稳定而不会被随意更改；系统控制权可以很好的规避冲突操作，并尽可能降低对用户访问的影响。

## 4. 网页设计

网页使用 HTML 语言进行设计，使用 jQuery 协助开发。jQuery 是一个优秀的前端 JavaScript 库，利用 jQuery 可以很方便地实现前端的事件管理、页面元素操作。

本文使用单页应用的方式，即将所有的功能操作都集中在一个 HTML 页面中，以减少浏览器与服务器之间的数据传输，提高网页应用的响应速度。网页的设计如图 4 所示。

## 5. 客户端设计

经过调研发现，目前市场额占有率最高的两个移动平台操作系统是苹果公司的 Apple iOS 以及谷歌公司的 Android。如果要基于原生开发环境开发两个平台的应用，则需要两套基于同语言的代码，不仅开发成本高，还为维护带来不便。因此，本文选用了 Qt 框架下的 QML (Qt Markup Language)进行跨平台的客户端软件开发。

Qt 是一个跨平台的 C++图形用户界面应用程序开发框架，在此框架下可以使用一套代码编程出能够在多平台允许的软件程序。QML 则是 Qt 框架下的描述性语言，它结合了 HTML5 和 CSS3 的一些特性，简化了界面元素、动画的设计，同时支持内嵌 JavaScript 脚本，有极强的拓展性[5]。用 QML 可以方便快捷地开发出可运行在 Apple iOS 及 Android 上的客户端软件。

客户端软件结构如图 5 所示。软件主要由两个部分组成，上层是基于 QML 的用户界面，主要负责用户接收用户的输入操作、界面管理、利用 JavaScript 中 Ajax 技术与服务器进行基于 HTTP 协议的网络通信；下层是基于 C++的后台服务程序，进行与文件系统进行数据交互从而对用户设置参数进行保存。此部分的主要功能是在文件系统中创建配置文件，对用户配置参数进行管理，包括保存、修改、删除软件上下两层是基于不同的语言编写的，相互之间的通信是基于 Qt 的信号机制，利用信号作为载体传输数据。

经过不同的编译器编译后，就可以生成两本版本的程序，分别运行 Android 和 Apple iOS 平台上。客户端的运行效果如图 6 所示。


**Figure 4.** Design of the web page
**图 4.** 网页设计

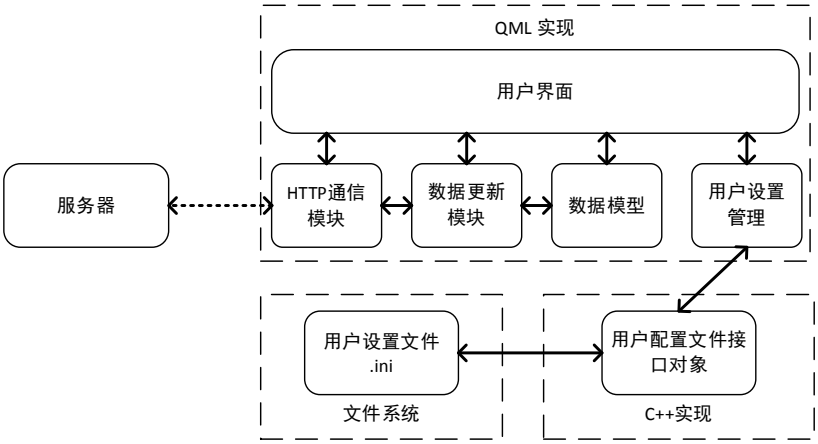


**Figure 5.** Structure of the client software
**图 5.** 客户端软件结构


**Figure 6.** User interface of the client
**图 6.** 客户端界面

## 6. 结束语

文中所述的远程监控系统的方案已经得到了实现，房间设备登记、用户管理、远程监控等功能可正常运作。C/S 与 B/S 混合模式让远程监控系统的访问方式更灵活，尽可能满足最多的用户需求。轻量高效的 Node 让服务器的开发变得更简单，令 C/S 模式与 B/S 模式的结合变得更自然流畅。而 QML 则令跨平台软件开发更简单，大大降低了开发与维护成本。

在系统开发期间，本文所使用的各项技术又有了新的发展与突破：QML 开发社区推出 Material Design 控件库，能让 QML 产品更接近于原生开发环境的软件产品；Node 也推出了新的版本，变得更稳定安全。在未来，系统将随着所使用技术的升级而得到更新，以保证稳定高效的运行。

## 参考文献 (References)

[1] 毛明毅, 蒋元恒, 陈志成 (2013) 智能家居远程 Web 管理控制平台的设计与实现. *微电子与计算机*, **5**, 121-124.

[2] 朴灵 (2013) 深入浅出 Node.js. 人民邮电出版社, 北京.

[3] 赵坤, 寸志, 吴中骅, 雷宗民 (2014) Node.js 实战. 电子工业出版社, 北京.

[4] Chodorow, K., 著 (2103) 邓强, 王明辉, 译. MongoDB 权威指南. 人民邮电出版社, 北京.

[5] Ryannel, J. and Thelin, J. (2015) Qt5 Cadaques. http://qmlbook.github.io