# 第一章 引言

## 1.1 项目背景

[国际电信联盟](http://baike.baidu.com/item/%E5%9B%BD%E9%99%85%E7%94%B5%E4%BF%A1%E8%81%94%E7%9B%9F)1( ITU)，对物联网做了如下定义：通过[二维码](http://baike.baidu.com/item/%E4%BA%8C%E7%BB%B4%E7%A0%81" \t "_blank)识读设备、射频识别([RFID](http://baike.baidu.com/item/RFID)2) 装置、[红外感应器](http://baike.baidu.com/item/%E7%BA%A2%E5%A4%96%E6%84%9F%E5%BA%94%E5%99%A8" \t "_blank)、[全球定位系统](http://baike.baidu.com/item/%E5%85%A8%E7%90%83%E5%AE%9A%E4%BD%8D%E7%B3%BB%E7%BB%9F)和[激光扫描器](http://baike.baidu.com/item/%E6%BF%80%E5%85%89%E6%89%AB%E6%8F%8F%E5%99%A8)等信息传感设备，按约定的协议，把互联网与任何物品相连接，进行信息交换和通信，以实现智能化识别、定位、跟踪、监控和管理的一种网络[1]。

IBM3认为，IT产业下一阶段的任务是把新一代IT技术充分运用在各行各业之中，具体地说，就是把感应器嵌入和装备到电网、铁路、桥梁、隧道、公路、建筑、供水系统等各种物体中，并且被普遍连接，形成物联网。

随着物联网技术的不断发展和进步，物联网应用的开发在IT界受到越来越多的重视。与此同时，越来越多的用于机器人以及物联网的开发框架也被应用于开发中。但是各个框架各有特色以及缺点，且不能一次性满足开发时的底层抽象及顶层封装的需要，或者说是对各种不同的微处理器4的支持。

此外，机器人的开发，人工智能的发展也需要对嵌入其中的微处理器做应用开发来搜集数据、实现功能。

而目前已被大量使用在开发中的Node.js5是一个基于 Chrome V8引擎的 JavaScript6运行环境。 Node.js 使用了一个事件驱动、非阻塞式 I/O 的模型，使其轻量又高效[2]。但是它并不方便直接应用于物联网或机器人应用开发，因为他没有为各种微处理器提供统一的抽象的解决方案。为此，一批以Nodejs为基础的JavaScript框架应运而生。比较具有代表性的就是Cylon.js7，Cylon对各种未处理的底层抽象做的很完整，但是在对外提供独立功能调用时并不是一个最好的解决方案。

[[1]](#footnote-1)

为了满足工作和研究需要，我们需要开发一个能够抽象各种硬件平台使用同一种API8来控制，并且不需要关心硬件平台的详细实现只需了解其如何使用的物联网或机器人开发框架。并且结合时下SOA9的架构微服务的思想，通过接口对外提供这些平台的信息或进行设置

## 1.2 典型类似的物联网开发框架应用场景及优缺点

基于Nodejs的机器人及物联网应用开发框架有很多，每一种都有他独特的功能及其优缺点。是下面介绍一种应用较为主流的框架Cylon.js,分析其特征及优缺点。

Cylon.js是一个机器人JavaScript开发框架，可使用Node.js进行机器人开发和物理计算；它提供了一个简单但强大的方法来解决同一时间处理不同硬件设备的问题。Cylon.js支持包括ARDrone、Arduino等在内的18个不同的硬件平台。开发人员可以使用同样的API来操纵不同的设备。它的优势在于：

1. 基本框架结构简单，易于上手；
2. 采用插件(模块)组装的结构，支持多种软硬件平台；
3. 可扩展性较好，支持自定义扩展新的硬件平台；
4. 开源

它的缺点在于，他基于Nodejs，对硬件的内存有一定的要求。

## 1.3 微服务架构应用于JS框架开发的情况

### 1.2.1 微服务架构定义

微服务架构的重点是业务系统需要彻底的组件化和服务化，原有的单个业务系统会分割为多个可以独立开发、设计、运行和运维的小应用。这些小应用直接通过服务完成交互和集成。每个小应用从前端到控制层、逻辑层、数据库访问，数据库都完全是独立的。每个小应用除了完成自身的业务功能外，重点就是还需要调用外部其他应用提供的服务，同时自身也将自身的功能对外发布为服务，进入到单个业务系统内部实现真正的组件化。

微服务可以在“自己的程序”中运行，并通过“轻量级设备于HTTP型API进行交流”。通过这一点，我们可以将服务公开与微服务架构区分开来。在服务公开中，许多服务都可以被内部独立进程所限制，若某个服务需要增加某种功能，就必须缩小进程的范围。而在微服务架构中，我们只需要在特定的某种服务增加所需功能，而不影响整体进程。

### 1.2.2 微服务架构优势

微服务架构的优势有很多。首先，通过分解巨大单体式应用为多个服务的方法解决了复杂性问题。在功能不变的情况下，应用被分解为多个可管理的服务。每个服务都有一个用RPC-或者消息驱动API定义清楚的边界。微服务架构模式为很难利用单体式编码方式实现的功能提供了模块化的解决方案，由此，单个服务很容易开发、理解和维护。

第二，这种架构使得每个服务都可以有专门开发团队来开发。开发者能能够自由选择开发技术，提供API服务。许多公司为了避免混乱，只提供某些技术选择。然后，这种自由意味着开发者不需要被迫使用长开发周期的项目起步时采用的过时技术，转而选择现在的技术。甚至于，因为服务都是相对简单，即使用现在技术重写以前代码也不是很困难的事情。

第三，微服务架构模式是每个微服务独立的部署。开发者不再需要协调其它服务部署对本服务的影响。通过使用这种改变可以加快部署速度。前端团队可以快速的部署变化。持续化部署在微服务架构模式下成为可能。

最后，微服务架构模式使得每个服务独立扩展。开发者可以根据每个服务的规模来部署满足需求的规模。甚至于，开发者能够使用更满足服务资源需求的硬件。比如，你可以在EC2 Compute Optimized instances上部署CPU敏感的服务，而在EC2 memory-optimized instances上部署内存数据库[3]。

### 1.2.3 微服务架构不足

Fred Brooks在30年前写道，“there are no silver bullets”，像任何其它科技一样，微服务架构也有不足。其中一个类似它的名字，“微服务”强调了服务大小，尽管小服务更乐于被采用，但是这只是开发的选择而不是最终的目的。微服务的目的是有效的拆分应用，实现敏捷开发和部署。

另外一个主要的不足是，微服务应用是分布式系统，由此会带来固有的复杂性。开发者需要在RPC或者消息传递之间选择并完成进程间通讯机制。更甚于，他们必须写代码来处理消息传递过慢或者不可用等局部失效问题。尤其是相对于单体式应用中通过语言层级的方法或者进程调用，微服务下这种技术显得更复杂一些。

微服务的另一个挑战是分区的数据库架构。实际应用中同时给多个业务分主体更新消息很普遍。这种数据更新对于单体式应用来说很容易，因为只有一个数据库。在微服务架构应用中，我们更新不同服务所使用的不同的数据库，今天高扩展性的NoSQL数据库和消息传递中间件并不支持这一需求。最终开发人员将不得不想出一个最终一致性的解决方案，从而对开发者提出了更高的要求和挑战。

另一个不足之处在于，微服务架构模式中，应用的改变将会影响多个服务。比如，在某个案例中，需要修改服务A、B、C，而A依赖B，B依赖C。在单体式应用中，开发者只需要改变相关模块再整合部署。而微服务架构模式就需要考虑相关改变对不同服务的影响。比如再上面的案例中，要先更新服务C，然后是B，最后才是A。

## 1.4 论文的主要工作和组织结构

本文主要介绍了由三个开发人员完成的基于Nodejs与微服务的JavaScript框架，主要用于Arduino、Raspberry Pi 以及其他GPIO和I2C总线设备的应用开发。介绍了项目架构的背景以及应用某些前沿技术框架的理论知识和分析。同时描述了这个项目的实现过程基本功能以及其应用的范围和前景。

第一章：概述和前言部分，主要介绍了项目背景，当前物联网及机器人开发领域热门框架的功能与不足，本项目所采取的架构以及运行环境的应用分析，并描述了该论文的主要工作与组织结构。

第二章：主要介绍基于Nodejs与微服务架构的JavaScript框架开发和实现过程中用到的相关技术知识与理论研究。

第三章：从项目概述和需求分析与概要设计三个方面，描述基于Nodejs与微服务的JavaScript机器人开发框架的整体概述和架构信息，同时分析和总结出功能性和非功能性需求，描述项目整体的设计框架以及通过功能划分对项目进行包设计和接口设计。

第四章：对本人负责的系统模块的详细设计和实现进行描述，其中包括对外接口以及和模块内部的类设计图。同时对部分关键代码进行分析解释。

第五章：总结该项目已实现的功能，探讨项目的缺点和不足，并指出该项目未来的扩展和发展方向。

1 http://baike.baidu.com/link?url=x718luTQBUcmtpDHh7\_8f6HwP4yf9P-omc4GTSrVBdrutwYz\_wab4JY51fqrW\_BqFa423-TdUjAejy3hu62pxHAjL-s1LEagv-9hobqgSkVbA\_jzpGP\_Zrqhdwvypud8

2 http://nodejs.cn/

3 <http://kb.cnblogs.com/page/521880/>（这部分是放到论文最后的引用部分的）

1. 关于国际电信联盟(ITU)的详细信息，参见：http://www.itu.int/en/pages/default.aspx

   2 关于RFID的详细信息，参见：http://www.baike.com/wiki/RFID

   3 关于IBM的详细信息，参见：https://www.ibm.com/cn-zh/

   4 关于微处理器的详细信息，参见：<http://www.mscbsc.com/cidian/baikeajh>

   5 关于Nodejs的详细信息，参见：http://nodejs.cn/

   6 关于JavaScript的详细信息，参见：http://www.w3school.com.cn/js/index.asp

   7 关于Cylonjs的详细信息，参见：https://cylonjs.com/

   8 API：Application Programming Interface,[应用程序](http://baike.baidu.com/item/%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F)编程接口

   9 关于SOA的详细信息，参见：http://www.jdon.com/soa.html [↑](#footnote-ref-1)