**Polycis接入平台**

**说明文档**

北京资信物联科技有限公司

**目录**

[1. 产品综述 3](#_Toc23187)

[2. 功能描述 4](#_Toc25030)

[2.1. 资源管理功能 4](#_Toc12088)

[2.2. 统一部署功能 5](#_Toc24561)

[2.3. 统一配置功能 5](#_Toc31753)

[2.4. 监控管理功能 6](#_Toc28845)

[2.5. 日志管理功能 6](#_Toc31978)

[2.6. 计划任务功能 6](#_Toc2788)

[2.7. 业务管理功能 7](#_Toc13653)

[3. 智能运维系统说明 7](#_Toc13204)

[3.1. 管理首页 7](#_Toc4110)

[3.2. 资源管理 8](#_Toc17710)

[3.2.1. 模板管理 8](#_Toc5526)

[3.2.2. 主机 12](#_Toc9907)

[3.2.3. 主机初始化 14](#_Toc14154)

[3.2.4. 群组 15](#_Toc6380)

[3.2.5. 服务 16](#_Toc23620)

[3.2.6. 中间件 17](#_Toc12850)

[3.3. 统一部署 18](#_Toc1658)

[3.3.1. 中间件部署 18](#_Toc9628)

[3.3.2. 服务部署 19](#_Toc22966)

[3.3.3. 任务查看 20](#_Toc6296)

[3.4. 统一配置 22](#_Toc23378)

[3.4.1. 中间件配置 22](#_Toc8751)

[3.4.2. 服务配置 23](#_Toc14567)

[3.4.3. 修改主机密码 24](#_Toc10630)

[3.4.4. 任务查看 24](#_Toc5445)

[3.5. 监控管理 26](#_Toc17759)

[3.5.1. 监测中 26](#_Toc5753)

[3.5.2. 监控配置 29](#_Toc3237)

[3.5.3. 管理 31](#_Toc1190)

[3.6. 日志管理 31](#_Toc2115)

[3.6.1. 日志查询 31](#_Toc2298)

[3.6.2. 视图定义 32](#_Toc14557)

[3.6.3. 视图查看 33](#_Toc21774)

[3.7. 计划任务 34](#_Toc30497)

[3.7.1. 任务添加 34](#_Toc32359)

[3.7.2. 定时任务 35](#_Toc27592)

[3.7.3. 循环任务 35](#_Toc30461)

[3.8. 业务管理 36](#_Toc1155)

[3.8.1. 应用管理 36](#_Toc4846)

# 产品综述

连接平台是为用户提供便利的物联网快速海量接入、管理、并提供开放能力接口的云平台，赋予更多行业更方便更低成本的互联网能力，和金融、汽车、农业、环保、工业、能源、政府、安防等产业打造更高效更安全、更环保、更可持续发展的智慧地球。

连接平台以微服务分布式的设计思想，保证平台的大数据量访问和服务器的水平扩容能力，平台具有容灾能力，以及平台可视化运维管理保证了平台能够稳定高效的运行。

平台主要提供一下几个大部分服务：硬件接入、可视化管理、API能力接口、更加强大的定制化服务，下面进行逐一介绍。

硬件接入部分，平台对硬件接入部分提供丰富的协议支持，全面支持UDP、TCP、HTTP 等主流协议，以及对MQTT、Modbus、EDP的支持等等。可以覆盖到仓储行业、智能穿戴设备、车联网、金融行业、能源电力、工业制造、工业物联网、服务物联网、智慧城市等行业。硬件接入部分为各行各业提供了便捷、快速、高效、安全的接入服务。

可视化管理部分，可视化管理是平台是对平台用户物联网设备进行管理的一个可视化入口，用户可以通过企业账户登录到连接平台，对企业下的物联网设备进行一系列的操作，例如：查看、修改、更新、重启等，能够对物联网设备的整个生命周期中的每一个环节进行完善的管理。

API部分，平台提供了丰富的的功能，另外平台也开发了第三方API接口服务，以供上层服务商开发和用户定制更加个性化的服务，API服务接口支持WEB服务开发、应用服务开发、以及移动客户端的SDK，能够帮助个人、企业实现大规模设备的API服务开发，详情请参考官方文档。

定制化服务，除了平台提供的丰富、完善、便捷、安全的服务之外，我们还提供更加强大的定制化服务，详情查看官网。

# 功能描述

## 设备接入功能

TCP服务功能：连接平台底层模块提供了丰富的协议支持，TCP服务是其中一个重要的服务功能，该功能是给硬件提供一个TCP的服务端，能够让物联网设备通过TCP协议连接到平台上，当物联网设备达到一定的数量级后，TCP服务的物理设备可以横向的扩充，通过zookeeper服务监听和nginx的反向代理，将物联网设备的连接均衡的分配到不通的服务节点上实现海量接入时的服务分发和负载均衡。

编解码功能：平台通过自定义通用编码来实现物联网设备和高级开发语言之间的操作，物联网设备通过规定的编解码协议向上发送的二进制或十六进制的数据，平台通过编解码协议将物联网设备发送的数据解析为平台所需要的格式给上层业务使用；上层业务向下发送指令时，平台通过编解码协议将指令解析为物联网设备所需要的二进制或十六进制的形式来操作硬件做相应的行为。

Redis高速缓存功能：Redis是在工程启动的时候将物联网设备的一些公共数据和操作频繁的数据保存Redis中，不直接从传统的关系型数据库中读取这些数据，是平台的性能提升两倍以上，同时减轻了数据库服务的压力，当数据达到一定的数量级时Redis缓存数据库可以做横向扩充来满足大数据量和高吞吐的性能要求。

RabbitMQ消息订阅发布功能：RabbitMQ 实现了硬件和上层业务的解耦功能，硬件将上行的操作通过平台将操作行为加载到RabbitQM的生产者中，上层业务的消费者订阅生产者的消息，来进行相应的操作，降低了系统的耦合性。生产者和消费者是分开的，所以实现了系统的异步操作，而且RabbitMQ是内存数据库，性能上与磁盘和数据库操作相比有恨到的提升，很大程度的提高了系统的性能和吞吐量。

设备接入通知上报功能：物联网设备接入到连接平台时，物联网设备会携带的设备信息，平台将设备的基础信息通过编解码得到方式存放的Redis内存数据库中以备后续的业务处理使用。平台会异步的通知，某个设备的在线状态，将设备的状态通知给相应的业务做处理，保证了设备状态的及时更新。

## 网关检测功能

网关检测功能：物联网设备连接到平台的同时会建立一个双向心跳连接检测，物联网设备会给平台主动发消息来证明设备本身的状态是健康的，平台会给物联网设备回复一条消息证明平台的服务是健康的。当设备主动下线时，会上报一条消息到平台上，然后平台将该设备剔除。当平台长时间接收不到硬件的心跳消息或平台给物联网硬件发送消息没有响应，平台回将该物联网设备主动剔除。在保证服务和物联网设备同时处于健康状态的同时，及时的更新物联网设备的在线状态。

## REST API 服务功能

HTTP 服务功能：连接平台上层API提供REST风格的API接口，以供上层或前端调用。REST API 分为两种类型的接口，一种是提供查询的接口，另一种是操作类的接口。

查询类接口：查询类接口主要是供上层业务的页面展示和业务需求的数据做的查询操作。这些数据中包括历史数据、非实时性数据和实时数据，针对不同的业务需求分别提供了完善的接口。

操作类接口：操作类接口主要是对物联网设备进行的一些列操作行为的接口。这些接口中包括物联网设备整个生命周期中的所有行为的接口，例如：服务状态变更、设备状态变更、重新启动等等。

可扩展性：HTTP 服务是微服务节点中的一个服务节点，当用户量、数据量和访问量达到一定的数量级时，可以支持横向扩充，利用zookeeper服务注册、服务发现和服务监控，nginx 的反向代理功能来实现服务节点的负载均衡，可以实现系统负载能力成倍增长，理论上是没有上限的，很好的保证了系统的扩展能力。

## OTA 功能

物联网规模化部署面临着众多挑战，例如：众多设备近距离共终端和庞大数量的设备互联。这些挑战催生出对无线(OTA)固件升级的需求，它将是物联网系统必不可缺的一个组成部分。OTA的有效性已在许多应用程序中得到验证，例如：手机会通过定期升级来接收新功能和修补漏洞。

OTA 功能是保证物联网设备的生命周期的重要功能，通过OTA功能将物联网设备的固件进行升级，保证了固件和物联网设备硬件之间的结合更加的稳定；保证了物联网设备的功能在原有的基础上更加的丰富；保证了物联网设备能够更加节能高效的进行。OTA升级功能保证了物联网设备的整个生命周期的完整性。同时降低了行业成本。

本台的提供了简单、易用且非常丰富的OTA 升级功能，平台中硬件设备信息以列表的形式使用web页面进行可视化展示，用户可以通过该web页面来查看设备的设备版本、状态以及其他一些附属信息。

用户可以通过该web管理页面进行查看固件，上传固件等操作，平台提供单个文件同步上传和异步上传，多个文件的同步上传和异步上传。平台对固件版本进行统一的管控，保证了升级的安全性。

平台支持多种形式的升级操作，用户可以选择单个物联网设备进行实现性的的升级，为后面的批量升级和全部升级功能做好前期的安全性和稳定上的测试。在保证单个设备升级，设备不出现异常的情况下，用户可以通过勾选要升级的设备进行批量的升级，单个升级和多个升级都是以异步的形式进行操作，这样大大提升了用户体验的流畅性，当然，批量升级没有问题就没有必要对设备一个一个的勾选，平台提供区域性的升级功能，可以勾选某个区域的全部升级进行升级，如果用户的物联网设备的数量比较少的情况下，可以使用升级某个用户的全部设备。

为了进一步保证OTA功能的速度和安全性，我们是将用户上传的固件文件包以分片的形式存放到内存数据库中的，加快了固件在拉去固件文件包的时候的速度的。

OTA升级功能的流程是比较简单的，但为了保证速度和安全性，后台对OTA升级的功能中可能存在的问题进行非常好的管理，从用户角度的OTA升级功能是这样的，用户点击升级物联网设备，平台会异步的在规定响应时常内返回下发命令是否成功，用户再次查看物联网设备的信息，查看物联网设备是否升级成功。当然，平台提供了非常实用的检索功能，可以对单个检索条件进行简单的检索，也可以对多个检索条件进行更加准确的组合检索，对想查看的物联网设备进行检索。

如果物联网设备升级成功，web页面会对物联网设备的版本号和状态等信息进行展示，来查看物联网设备是否升级成功。如果在升级过程中，某些物联网设备升级失败，我们可以对物联网设备进行再次的升级，在物联网设备正常的情况下，保证物联网设备都能够进行快速、安全的升级。

## 容错功能

在系统尝试间的运行过程中，由于网络、用户操作的逻辑、硬件故障、断点等都是不可避免的，要保证数据的完整性和不丢失是非常重要的。接入平台在这种容错能力上做了多个方面的功能，主要包括：服务器双机热备、平台重发机制、断点保护机制、JTA事务管理等容错机制。

服务器双击热备，首先说明，应用服务器可以采用分布式的概念进行横向扩充，为了保证数据的安全性和应用的高可用性，接入平台的每个应用服务器节点都是采用主从复制的形式进行部署的，主双机热备按照工作中的切换方式分为：主-备方式(Active-Standby方式)和双主机方式(Active-Active方式)，主-备方式即指的是一台服务器处于某种业务的激活状态(即Active状态)，另一台服务器处于该业务的备用状态(即Standby状态)。而双主机方式即指两种不同业务分别在两台服务器上互为主备状态(即Active-Standby和Standby-Active状态)，接入平台采用主备方式(Active-Standby方式)进行容错的。

故障隔离，简单地讲，高可用(热备)就是一种利用故障点转移的方式来保障业务连续性和数据安全性。其业务的恢复不是在原服务器，而是在备用服务器。热备不具有修复故障服务器的功能，而只是将故障隔离。

故障检测，故障检测是双机热备的任务，不同的双机检测点的多少决定了双机热备软件在功能和性能上的优势，接入平台分为系统级、应用级、和网络级三个方面。系统级检测主要通过双机热备软件之间的心跳提供系统的检测功能、应用级提供用户应用程序、数据库等的检测功能，网络级别的检测提供对网卡的检测及可选对网络路径的检测功能，灵敏的故障检测是应用服务的基础保证。

双机热备的资源指某种业务运行过程中所依赖的最小的关联服务，不同的双击软件所提供的资源多少页不相同，提供的可切换资源越多，软件应用的范围也越广，接入平台中提到的服务器资源主要包括，可切换的网络IP资源、计算机名、磁盘卷资源、服务器进程等资源。

一般情况下的双机切换时间为一到五分钟，而快速切换的时间为三到五秒钟。用户使用过程中基本上是无感知的

## 扩展功能

随着业务的发展，访问量和数据资源的不断扩充，以及高性能服务器高昂的价格，采用廉价机来进行横向扩展的的优势非常重要。

接入平台支持多种形式的横向扩展功能，其中包括web应用服务器横向扩展、中心服务横向扩展、数据库横向扩展以及业务的横向扩展。

Web应用服务器横向扩展，web应用服务器的横向扩展在高用户量的时候对web服务器的的性能能够进行很好的增加，横向扩展web应用服务器通过F5和zookeeper以及Nginx等技术进行web服务器的负载均衡，实现高吞吐量的访问，不仅增加了服务器的吞吐量，而且多个负载节点中存在其中的某个服务节点出现故障也不影响服务的正常使用，当一个节点宕机的时候，数据流向会均衡的分配到其他节点上。

中心服务横向扩展，中心服务主要处理的是复杂的业务逻辑，在高用户量的情况下复杂的业务逻辑处理会增加服务器的负载，可到会导致业务处理缓慢，这个时候可以通过中心服务横向扩展的方式来增加中心服务的节点数量来解决这个问题，同样的，不仅增加了中心服务的吞吐量，而且其中的某个节点宕机不影响其他节点的正常运行和使用。

数据库横向扩展，用户量的急剧增加和海量数据的产生，会导致数据库服务器的压力快速增加，如果不能妥善处理，可能会导致数据库服务的崩溃，将导致整个应用服务的停止。同样的，通过横向扩展数据库来增加数据库的负载能力，提高整个系统的性能。

业务方面的扩展，业务方面的扩展的方面就比较广泛啦，接入平台指的是，接入平台在增加新的业务的时候，不会影响到整个应用系统的运行，也就是说业务之间是解耦的，互相之间都是一个独立的业务。这种设计模式在以后的业务方面的扩展上具有非常便捷的特性，同时减少的成本。

## 日志记录分析功能

日志是一个系统不可缺少的一个重要组成部分，通常是系统的运行记录，日志一般分为两大类型。

系统日志，记录系统中硬件、软件和系统问题的信息，同时还可以监视系统中发生的事情。系统日志包括系统日志、应用程序日志和安全日志。接入平台提供了系统级别的日志记录和分析功能，实时监控应用服务器的硬件健康状态、软件的健康状态和系统运行时产生的问题，将这些服务器运行期间产生的日志保存下来，为后续的日志分析做数据存储。

平台应用日志，接入平台记录web服务器接收处理请求以及运行时错误等各种原始信息以.log结尾的文件。接入平台通过对日志进行统计、分析、综合就能有效地掌握网站运行状况，发现和排除错误原因，了解客户访问分布等，更好的加强系统的维护和管理。日志也是了解搜索引擎工作原理和搜索引擎对网页抓取频率的最佳途径。通过这个文件，可以了解搜索引擎什么时间、抓取了那些页面，以及可以知道是主搜索引擎还是从搜索引擎抓取了网站等的信息。

# 接入平台说明

## 整体架构设计

架构设计是需求分析到软件实现的桥梁，也是决定软件质量的关键。架构设计关系到系统的高性能、稳定性、可靠性、扩扩展性和可维护性等方面。

接入平台结合实际的应用场景和架构的设计思想，将系统的高性能、高吞吐、稳定性、安全性、可扩展性和可维护性考虑到架构的设计中。

接入平台架构图如下：



从系统架构图中可以清楚的看出主要可以分为三个大部分，第一部分是硬件部分，第二部分是协议处理部分部分，第三部分是上层业务部分，下面架构设计详细描述中将对以上三部分进行详细的描述。

### 整体架构设计概述

从系统架构图中可以清楚到看出分为三部分，硬件部分、协议处理架构部分和上层业务部分。

硬件设备部分，硬件设备部分指的是物联网设备，可以是物联网设备、车联网设备、农业物联网设备等。只要能够提供基础网络连接功能的设备都可以。

协议处理架构部分，协议处理架构部分是整个架构的核心，对下主要处理物联网设备的接入，对上主要提供API服务支持。

上层业务部分，上层业务部分指的是web工程，可以是系统本身的后台运维管理，也可以是为其他企业提供接口服务。

以上是对系统架构的简单描述，下面将对架构设计的各个部分进行更加详细的描述。

### 整体架构设计详细说明

如上文所述，整体架构设计主要包括三个大的方面，第一是硬件接入部分，第二是协议处理架构部分，第三是上层业务部分。

#### 硬件接入部分

硬件接入部分是整个接入平台的基础，硬件接入部分其实和硬件没有多大的

的关系，只要硬件能够提供基本的网络连接能力，就能够满足连接到接入平台上的。

接入平台硬件部分分在架构中的体现是TCP服务部分，TCP服务部分指的是接入平台通过提供一个TCP的服务端，能够提供给硬件设备进行连接的，硬件设备就相当于客户端。TCP服务部分的架构图如下：



从图中可以清楚的看出来TCP服务部分的详细的设计，TCP服务部分的各个部分的职责分工是非常明确的，这样可以明显的降低系统的耦合性。下面对TCP服务部分中的各个部分进行拆解说明。

首先，硬件需要通过TCP服务连接到接入平台，硬件连接到接入平台当然是

需要平台和硬件之间保持相同的协议，硬件设备才能够连接到接入平台。硬件设备连接到接入平台后进行其他操作。

然后，硬件和接入平台之间需要通信，硬件设备只能提供二进制、十六进制或ASCII的数据，对于像Java这类的高级语言当然是能够处理这些基础类型的数据的，但为了保证接入平台编码和解码的通用性，以及上层业务处理的便捷性，接入平台将硬件设备和接入平台之间的通信加入了编解码的功能。

接入平台解码功能，接入平台将应将上行的数据解码为像Java高级语言面向对象的思想的对象。接入平台将硬件上行的数据通过字节码暂时存放到Java的缓冲区中，然后，通过字节的输出流将缓冲区中的字节码按照一定的算法解析为Java的对象形式，平台提供通用的编解码功能和定制的编解码功能，能够解析绝大多数硬件上行的数据。

接入平台编码功能，上层业务对硬件下发指定的操作指令时，是通过Java高级对象的形式下发的，对于硬件来说没有好的方法来解析这种格式的命令格式。接入平台通过一定的算法对上层业务下发的命令进行编码，编码为硬件设备需要的格式，然后将编码过后的字节码放入缓冲区中，然后将字节码逐一发送给硬件设备。

接入平台的编解码是通过图中TcpServerHandler部分进行操作的，其中提供了丰富的编解码形式，可以满足绝大多少情况下的编解码需求。上行是TcpServerHandler将字节码解析为对象，下行是TcpServerHandler将对象解析为字节码的形式。

业务操作部分包含两个方面的内容，一种是响应部分，另一种是上报部分，这是两种形式的业务操作。

响应部分，响应部分指的是上层业务操作硬件部分，然后硬件部分得到响应的指令后进行一系列的动作，然后将处理的结果通过响应的形式返回给上层业务操作来保证业务的完整性。接入平台会将响应的结果编码，然后将解析过的结果存放的高性能的消息发布订阅队列中，用于被上层业务来消费消息发布订阅的消息。

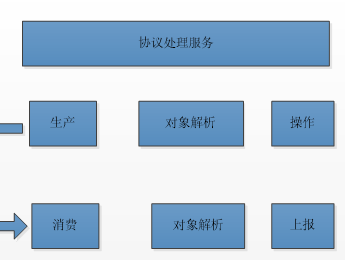
上报部分，上报部分指的是硬件主动上报的数据。这种情况不是上层业务对硬件操作后硬件反馈的数据，而是硬件在和平台保持连接的情况下主动给接入平台发送消息，然后接入平台将消息通过TcpServerHandler解析为对象，然后将对象存放到消息发布订阅队列中，用于背上层业务来消费消息发布订阅的消息。

硬件设备和接入平台之间会一直保持着一个双向的心跳，接入平台会和硬件发送心跳数据，然后硬件设备接收到接入平台的心跳数据回复给平台一个心跳数据，同时硬件设备也会主动个接入平台发送心跳数据，接入平台接收到硬件设备的心跳数据后回复硬件设备一个心跳数据，这样双向心跳都正常的情况下，说明接入平台和硬件设备都是正常运行的。如果平台连续发送心跳数据硬件设备没有回复，接入平台会将硬件设备踢下线。如果硬件设备连续发送心跳数据，接入平台没有回复，那么，硬件设备会主动下线并尝试重新注册连接。

#### 协议处理部分

协议处理部分指的是接入平台和硬件之间的协议约定处理，或者通用的协议处理方式，这样能够保证接入平台和硬件设备之间良好的运行。

协议解析部分处于中间的位置，对上层业务提供对应的服务，处理全部的下层业务。协议处理部分架构图如下图所示。



协议解析部分主要包含三个方面，第一是消息发布于订阅，第二是对象解析，第三是响应的操作。

对于下层业务响应或上报的数据存放到消息发布与订阅，也就是相当于生产者角色，协议处理部分将下层生产的消息取出来做相应的业务处理，相当于消费者，来完成响应或上报的完整流程。

对象解析，无论是上层业务下发指令还是下层业务上报或响应数据都需要对象解析的支持，将消息发布订阅队列中的消息解析为业务或硬件设备需要的格式进行处理，然后完成对应的消息响应或上报和命令的下发。

对于上层的具体操作来说，具体的业务需要下层来具体的实现，然后才执行具体的操作，上层具体的操作有影响着下层的业务逻辑的具体处理方式。

#### 上层业务部分

上层业务部分主要是提供一个HTTP的服务端，将HTTP服务暴露出去， 以供被调用来处理一定的业务逻辑，HTTP服务架构图如下图所示。



从上图可以清晰的看出，这部分是提供了一个HTTP服务。整个HTTP服务又可以分为HttpServerHandler部分和MQ接收类部分，HttpServerHandler部分部分封装了对数据解析的处理和MQ的相关的处理；MQ接收类部分封装了对响应部分的封装和上报数据的封装以及MQ处理的封装。

HttpServerHandler中包括了解析对象部分，将对象解析为业务处理需要的格式，然后进行相应的业务处理。通过Manager操作类来获取硬件设备的具体信息，然后将获取到的硬件设备数据进行解析，然后将解析过后的数据发送的MQ中，然后MQ等待响应的结果。

MQ接收类中主要包括对响应数据的处理和上报数据的处理。

响应部分，这部分通过MQ对象将硬件设备响应的数据取出来，然后根据uuid动态查询相应的HTTP对象，最后发送相应的响应。

上报部分，这部分通过MQ数据对象将数据从MQ中取出来，然后将取出来的数据封装成JSON的对象的格式，然后发送相应的请求，在得到响应后，MQ对对象进行封装，然后调用MQ消费者的相应方法发送消息，等待响应结束，这是一个完整的流程。

## .系统流程说明

系统流程图是描述系统物力模型的传统工具，能够清晰的看出系统的数据流向关系，使得整个系统的各个组件之间的数据流向能够清晰的展示出来。

接入平台流程主要包括两大方面：设备接入流程、设备操作流程。下面将针对系统流程图进行详细的说明。

### 设备接入流程

设备接入流程图，设备接入流程图指的是硬件设备连接到接入平台的完整流程。我们能够从图中清晰的看出硬件设备连接到接入平台的完整流程，设备接入流程图如下图所示。



从上图可以清晰的看出硬件设备连接到接入平台的完整流程。下面将对硬件设备连接到接入平台进行详细描述。

首先是硬件设备首次连接到接入平台，如果硬件设备是首次连接到接入平台。

由硬件设备发送连接接入平台的请求，接入平台和硬件设备之间创建连接通道，接入平台将硬件设备的设备信息保存到数据库，结束硬件设备数据保存到数据库的操作。

同时，硬件设备发送注册指令，接入平台将执行解码协议，然后查找uuid，通过查找的uuid分配设备的唯一标识id好，如果数据库中存在该设备唯一标识id则更新数据库中该设备的状态，然后通过编解码协议发送请求结束硬件设备连接接入平台的流程。

如果数据库中不存在该设备的唯一标识id则将接入平台生成的设备唯一标识id保存到数据，然后通过编码协议发送请求结束硬件设备连接接入平台的流程。

通过上述的简单几步操作就可以实现硬件设备和接入平台之间的连接。

### 设备操作流程

在3.2.1中通过简单的几步实现了硬件设备和接入平台之间的连接。然后就是硬件设备和上层业务之间的交互。

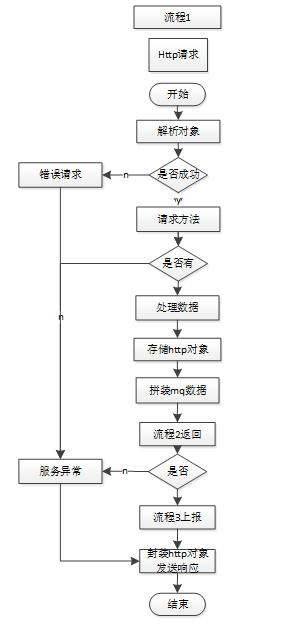
硬件设备和上层业务之间的操作主要由两种：上层业务请求和硬件设备上报。下面将针对这两种设备操作进行详细的描述。

#### 上层业务请求流程

上层业务通过向硬件设备发送相应的指令来让硬件设备来完成一系列的操作。该操作属于设备操作中的一种情况，通过流程图能够清楚的知道整个请求的数据流向和完整流程。

上层业务请求流程包括三个部分的流程展示，一个主流程和两个子流程，下面将对主流程和子流程进行详细的描述。

主流程图如下所示：



首先由上层业务发起HTTP请求到接入平台，接入平台接收到请求后将开始整个流程。

在接收到HTTP请求后，接入平台将开始对HTTP的请求就行解析，解析请求的格式以及参数。

如果解析对象失败，此时，接入平台并不会对这个请求进行任何的业务逻辑的处理直接将异常信息封装为HTTP对象返回个调用者，然后结束整个上层业务请求流程。

如果解析对象成功，将从解析对象中去除具体请求的方法，接入平台去查找是否存在该操作。

如果接入平台中不存在在该请求中请求的方法，那么，接入平台并不会对该请求进行任何的业务逻辑的处理，接入平台会直接抛出服务异常，然后，将错误信息封装成HTTP对象返回，最后，结束整个上层业务请求流程。

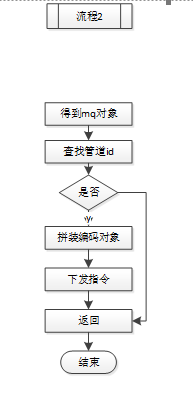
如果接入平台中存在该请求中请求的方法，那么，接入平台会将请求的参数按照业务逻辑进行解析处理。

按照业务逻辑处理数据成功后，接入平台会将该请求的HTTP对象存储到接入平台中。

存储HTTP对象成功后，接入平台会对HTTP对象中的数据进行拼装，将数据拼装成业务需要的格式。

将HTTP对象的数据拼装完成之后，接入平台会将拼装后的数据存储到MQ中，以供消费者消费数据。

然后，以上的业务流程都通过后会跳转到流程二中，再进行子流程的流程处理，流程二的流程图如下所示。



在上述的流程没有出错的情况下，才能够进入到流程二中，如果上述流程中其中一个流程出错，流程将不会进入到此流程中。

上一步中，接入平台将HTTP对象中的数据拼装成业务所需要的格式存放到了MQ中。

在此流程中，通过工具对象得到MQ的对象，然后将数据从MQ中取出来来做进一步的流程处理。

通过MQ对象将数据取出后，接入平台会根据取出来的数据来查找相应的管道id。

如果查找管道id失败，系统将不会对该请求进行任何其他的业务逻辑的处理，直接返回到流程一中。

流程二将错误结果返回给流程一，流程一也不会做其他更多的业务的处理，而直接将错误信息封装成HTTP对象返回。

如果查找管道id成功，接入平台会通过一定的算法将进行拼装编码对象操作。

拼装编码成功后，接入平台会将拼装编码后的，硬件能够识别的的指令下发给硬件。

然后完成上述流程后流程二就会结束，返回到流程一以继续进行流程处理，到此，流程二结束。

流程二结束后返回后，又进入流程一中的继续进行业务流程一的处理，来完成流程一的整个逻辑。

流程一中，会对流程二中返回的结果进行判断，根据返回结果信息，来进行相应的流程处理。

如果流程二中返回的结果是失败，那么，流程一将不会对业务逻辑进行更多的操作，接入平台会将错误信息封装为HTTP对象响应并返回，来完成完整的业务流程。

如果流程二中返回的结果是成功，那么，流程一会将流程流向流程三，来进行下一步的流程处理，流程三的流程图如下所示。



通过流程一和流程二，然后跳转到流程三，流程三将对整个流程进行进一步的流程处理。

流程三在接收到流程二中的命令后，然后执行了一系列的操作后，将处理的结果向上返回的过程。

硬件执行完成下发的指令后，硬件将处理的响应结果返回给上层业务的流程，来完成整个流程。

硬件响应的指令，接入平台是无法直接提供给上层业务的使用，通过协算法进行协议的解码。

协议解码成功后，接入平台会在指令库中进行返回结果的指令进行查找，验证接入平台是否存在硬件返回的指令。

平台存在该指令，然后，接入平台会对硬件返回的数据进行拼装，拼装成业务所需要的格式并存储到MQ中。

拼装MQ数据成功后，接入平台将发送MQ数据，然后完成流程二的整个流程。

流程二处理成功后，将返回流程一继续完成整个流程，流程一会根据流程三处理的结果进行下一步操作。

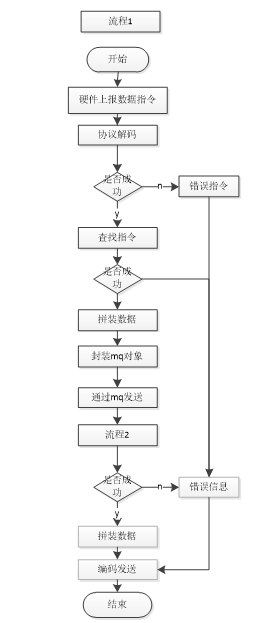
流程三处理完成后，流程一种将对整个流程中的处理的最终结果封装成HTTP对象并发送响应。

最后整个上层业务请求流程结束。

#### 硬件设备上报流程

硬件设备通过接入平台上报数据来完成相应的操作。该操作数据设备操作中的一种情况，通过流程图能够清楚的指导整个请求的数据流向和完整流程。

硬件设备上报主流程图如下：



首先要说明的是，此流程图是硬件设备主动请求接入平台，然后接入平台做相应的业务流程的处理。

硬件设备主动请求接入平台，然后接入平台接收到硬件设备接收到请求后将开始整个流程。

在保证硬件设备和接入平台之间保持连接的情况下，硬件设备开始上报指令，然后流程开始。

接入平台接收到硬件设备上报的数据后，接入平台编解码部分会开始对硬件设备上报的数据进行解码。

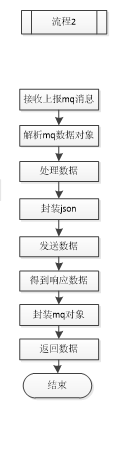
如果接入平台解析硬件设备失败，接入平台不会对硬件请求做过多的业务逻辑的处理，会抛出错误指令，然后交给接入平台的错误指令处理机制进行进一步的逻辑处理。

如果接入平台解析硬件设备成功，下一步接入平台将对硬件设备上报的数据进行数据拼装。

数据拼装成功后，接入平台会将数据封装成MQ需要的对象格式，并将对象存储到MQ中。

将封装对象存储到MQ中之后，接入平台会将该请求MQ发送到上层的业务中使用。

MQ对象发送成功后，流程一会将流程跳转到子流程流程二中去，然后做后续的逻辑处理，流程二如下图所示。



流程二在接收到流程一的请求之后，将会开始整个流程二的业务逻辑的操作，来完成流程二。

接收流程一种上报的MQ消息，用来完成整个的流程一的业务流程的处理操作。

接收上报MQ消息成功后，接入平台将解析MQ数据对象，然后，进行下一步的业务逻辑的处理。

接入平台解析MQ数据对象成功后，将接收的数据按照业务逻辑以一定的算法进行处理。

对数据进行处理后，接入平台将会对处理的数据以业务需要的格式封装成JSON的数据格式。

将数据封装成JSON格式之后，接入平台将封装过后的JSON数据发送给上层业务中。

向上发送JSON数据成功后，接入平台得到响应数据，处理后续的业务逻辑的操作。

接入平台得到响应后，下一步会将响应数据封装成MQ对象，处理后续的业务逻辑的处理。

封装MQ对象成功后，接入平台将返回数据，然后，返回流程一，最后结束整个流程二。

流程一在接收到流程二处理的结果，会根据流程二的处理结果，然后，进行下一步的业务逻辑处理。

如果流程二处理失败，接入平台将错误信息封装，然后，返回响应结果，最后结束整个流程。

如果流程二处理成功，接入平台会进行拼装数据，然后，进行下一步的业务逻辑的处理。

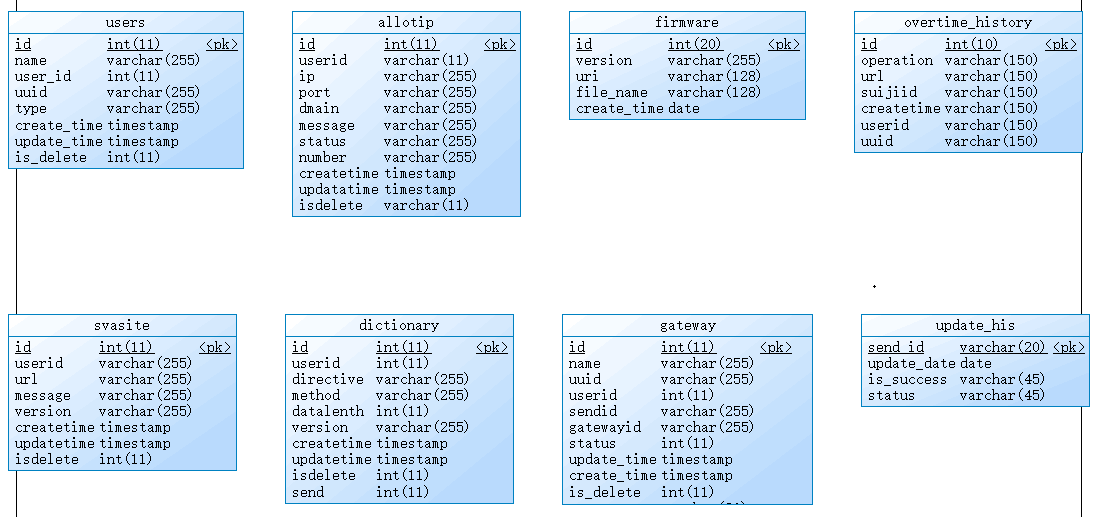
拼装数据成功后，接入平台的编解码部分会将处理结果进行编码，然后，将编码过后的数据发送给硬件。

完成上述的所有流程后，整个硬件设备的上报流程也就会随之结束。

## 数据库设计

### 关系型数据库设计

数据库设计在整个系统设计中非常重要的部分，按照数据设计的三范式：原子性、唯一性、独立性的原则的基础上，结合真实的应用场景设计符合生产实际情况的，如下所示是连接平台的关键表。



users表是用户表；

allotip表是操作表；

firmware表是固件信息表；

overtime\_history表是超时历史表；

svasite表是连接资源表；

update\_his表是更新历史表；

dictionary表是字典表

### 非关系型数据库设计

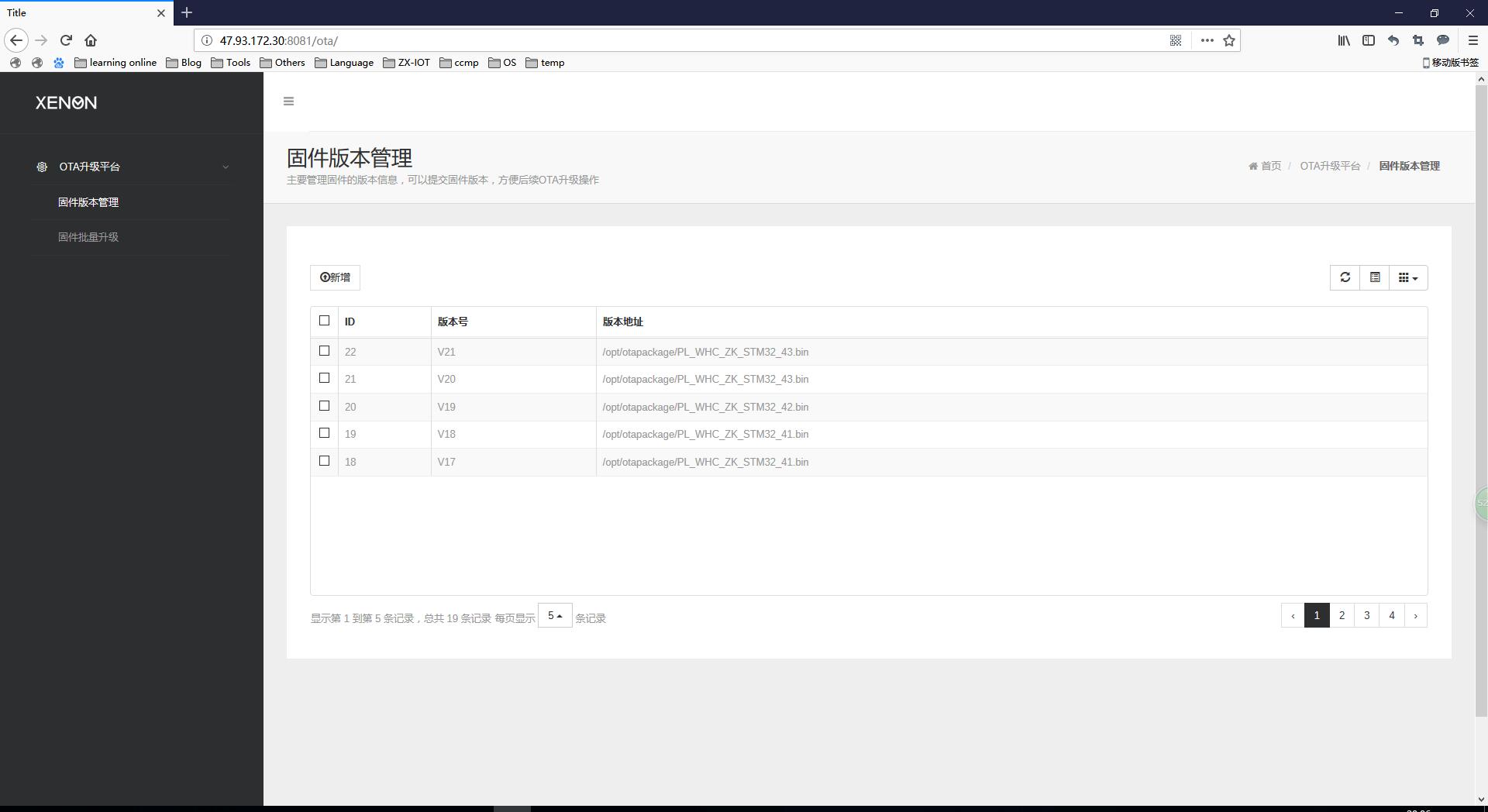
在系统中非关系型内存数据库是提高服务性能非常好的方式，连接平台将一些频繁操作的数据在项目启动的时候，将这些数据加载到内存数据库中来提高系统的访问速度。

## 功能详细描述

### OTA功能

OTA是接入平台的重要功能，OTA能够保证硬件设备的持续更新，保证硬件设备的生命周期得到持续。

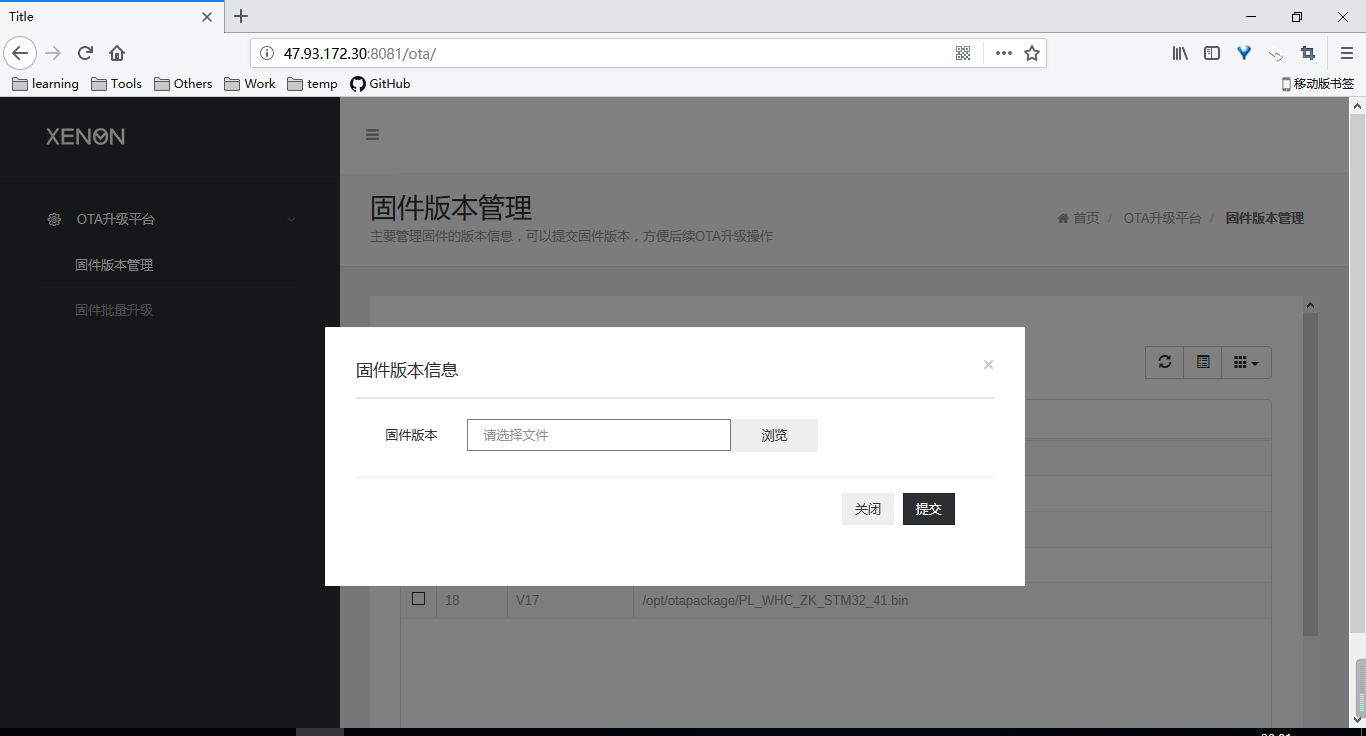
如下图所示是OTA模块的固件版本管理页面：



从这个页面可以看出，固件版本管理中包括“新增”，“更新”，“筛选”，“排序”，“复选”，“分页”等操作。

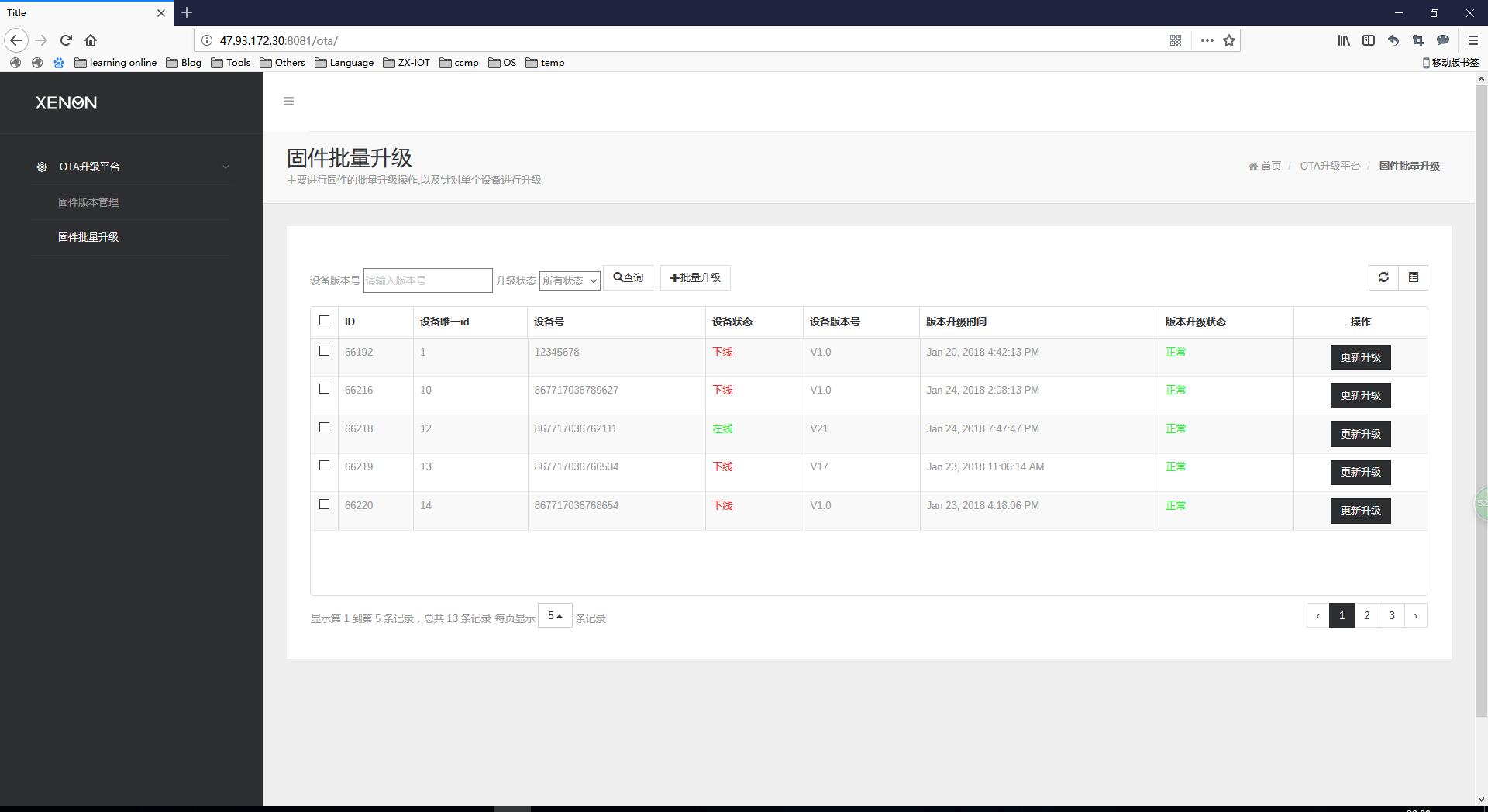
列表信息中，固件ID、版本号、版本地址；

“新增”页面如下所示：



从页面上可以看出，上传固件文件窗口支持浏览文件功能，以及支持上传多个文件。

固件批量升级页面如图所示：



列表中包括设备ID、设备唯一id、设备号、设备状态、设备版本号、版本升级时间、版本升级状态、和操作。

从页面上可以清楚的看出，固件批量升级页面包括根据“设备版本号”，“设备升级状态”，进行组合查询，“分页”，可以进行单个设备“更新升级”，勾选多个设备进行升级，以及区域性的全部设备升级。