项目指导书

课程名称：航空叶片制造系统的大数据与云计算应用及实践

项目名称：大数据采集、存储与分析系统开发

杨立娟、李卓林

西安交通大学机械工程学院实验中心

2025 年 10 月

一、实验目的

1.了解智能制造产线架构、数据来源等；

2.熟悉智能制造产线大数据采集系统的原理与架构；

3.掌握数据通信方法，能够开发 web 网页，实现与产线系统的通信，获取数据，并进行实时数据可视化；

4. 掌握数据库使用方法，实现实时数据边缘计算与云存储；

5. 掌握大数据分布式存储与分布式计算原理，实现实时数据的分布式存储；

6. 能够利用信号处理和人工智能算法实现产线设备的状态监测和故障预测等；

7. 培养学生大数据系统开发的综合实践能力和创新能力。

二、实验软硬件

VUE、python、边缘服务器、产线、虚拟数控系统、MQTT 服务器等。

三、实验原理

1. 基于物联网的大数据系统架构

基于物联网大数据系统的远程数据采集系统模型的体系结构可以分为感知层，网络层和应用层三个部分，如图 1 所示。

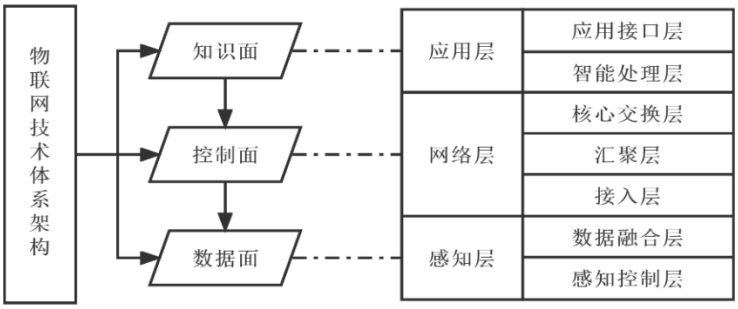


图 1 数据采集架构

感知层是物联网发展和应用的基础，包括传感器等数据采集设备与传感器网络。感知层以射频识别技术、无线通信、传感器技术等为主要技术，是物联网识别物体、采集信息的来源。在机械制造工业车间领域，感知层通常通过 PCI、API 以及 OPC UA、NC-Link 等协议进行数据采集并对所采集到的数据进行封装、处理与传输。

网络层是建立在通信网络和互联网基础之上的融合网络，由互联网、无线和有线通信网、网络管理系统和云计算平台等组成。网络层通过互联网、通信网络等实现信息的获取、交互、传输与存储。网络层通常使用 HTTP、MQTT、COAP、

XMPP 等传输协议进行数据或信息传输，并使用 MySQL、InfluxDB 等数据库对数据或信息进行有效管理。

应用层是物联网和用户的交互接口，是实现物联网的智能应用。应用层可以对感知层所采集到的数据进行处理、计算和分析，实现对物理世界的实时控制、精确管理和准确决策。应用层主要包括云计算、大数据等技术，并可通过 Web 开发在 APP 应用程序或 PC 网页端等终端实现广泛的智能化应用解决方案。

具体的工业大数据系统架构如图 2 所示，包含边缘端、云端、大数据平台、分析应用等。

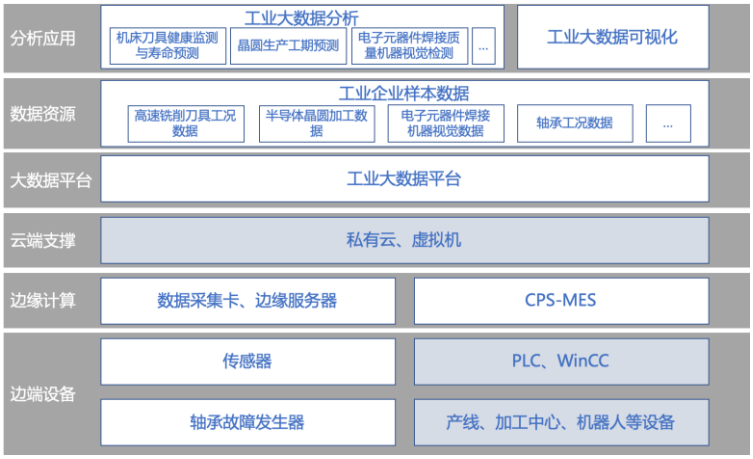


图 2 工业大数据系统架构

2. 智能制造学科交叉创新实践平台

智能制造背景下的智能制造技术人才应具有学科交叉复合融通的知识体系、工程实践能力和创新能力及工程社会意识。2016 年，筹建智能制造学科交叉创新实践平台，如图 3 所示，以微涡发动机核心零部件加工为载体，实现产品设计、加工制造、智能管理、物流服务等产品全生命运行、监控过程，将智能制造的机器人、数控机床、云平台、物联网、信息化管理等技术进行融合，具有无人化智能车间的基本功能和形态。

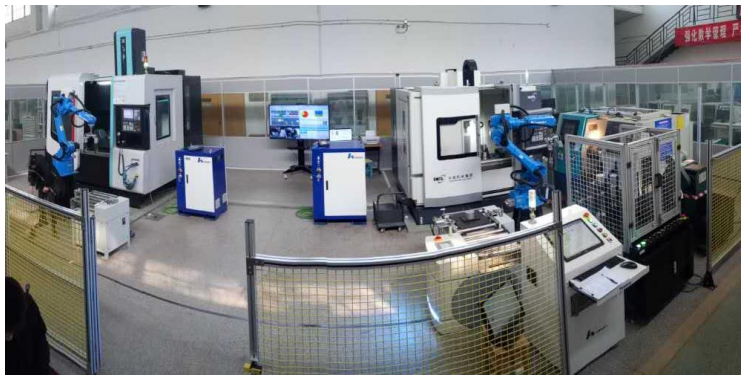


图 3 微涡发动机叶轮加工智能制造实践平台

整个平台建设以国家智能制造系统标准架构为参考，构建了设备层、采集与控制层、管理层、决策层的四层级架构体系如图 4 所示。产线的智能设备主要包

括：五轴加工中心、三轴加工中心、数控车床、2 台工业机器人、八工位自动料仓、AGV 小车、物料中转台等设备。

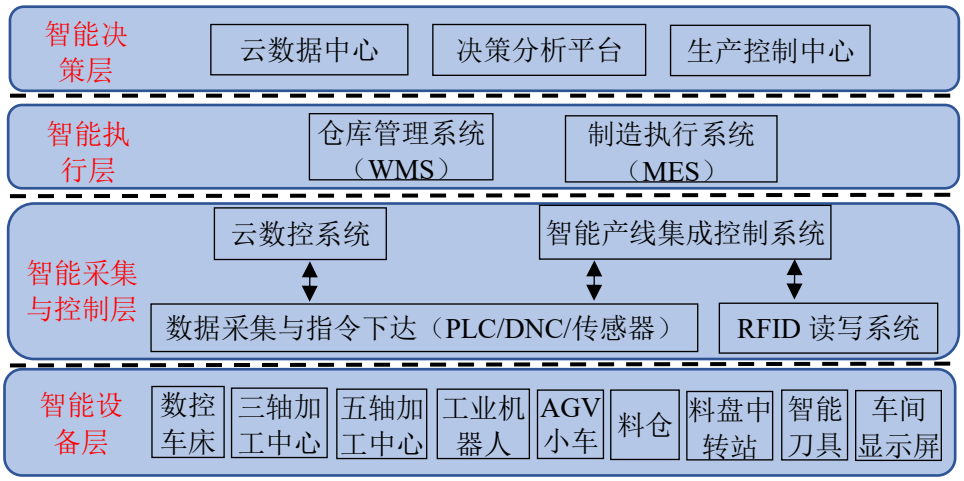


图 4 智能制造平台四层次架构体系

（1）智能制造产线多源数据采集系统

基于智能制造学科交叉创新实践平台构建的智能制造产线多源数据采集系统，硬件方面主要包括工控机、采集板卡、传感器、数控机床、web 服务器/交换机等，工控机搭载采集板卡，实现机床振动、声音、温度，数控系统内置信号可以根据需求选择，主要包括负载电流、切削速度、刀具参数、各个轴的位置等，通过局域网可进行数据采集。数据采集硬件如图 5 所示，采集端软件部署在工控机（上位机）上，连接到数控机床的局域网中，实现数据的采集与本地存储，主界面如图 6 所示。web 部署在 web 服务器上，连到校园网中，能够远程下发采集参数，控制设备进行数据采集。采集端与控制端通过 MQTT 协议建立连接，如图 7 所示，通过发布接口、订阅接口实现数据信息的交互，控制端通过 web 远程下发采集参数、控制设备采集数据，同时可监控采集数据的状态，并远程下载保存数据等，部分 web 界面如图 8 所示。

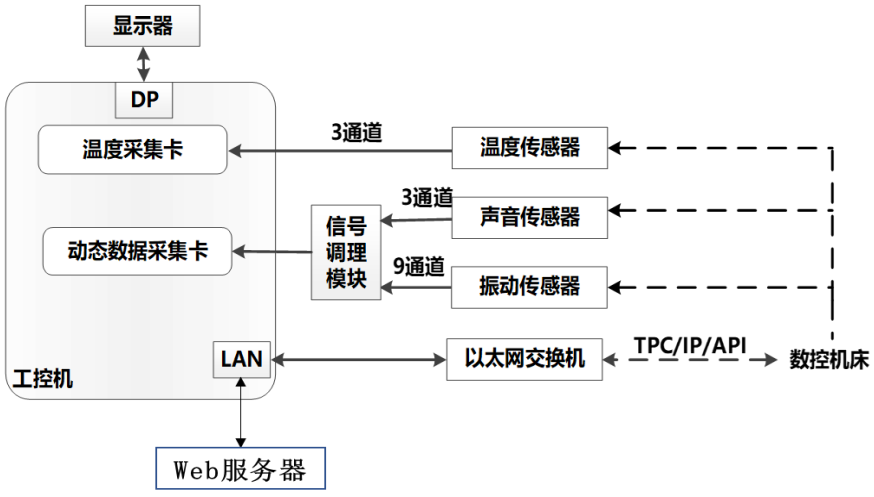


图 5 数据采集硬件

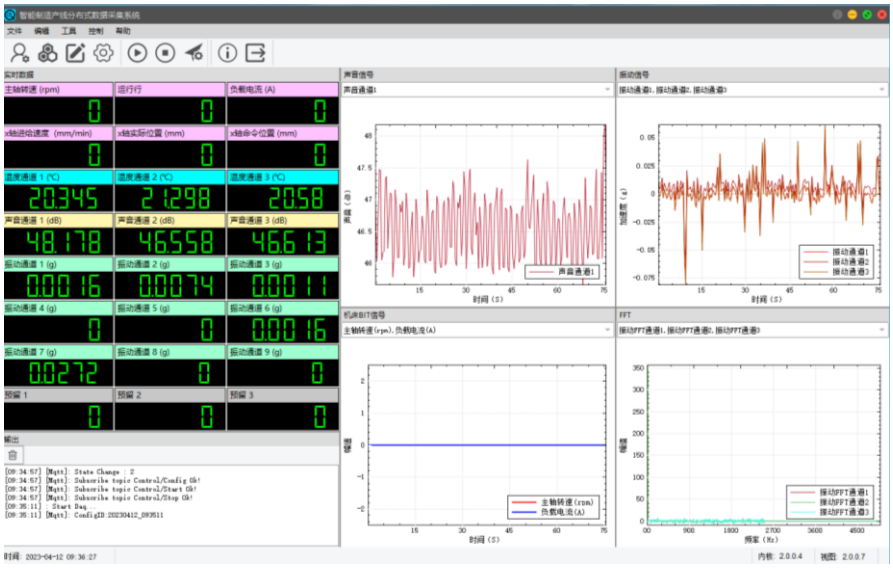
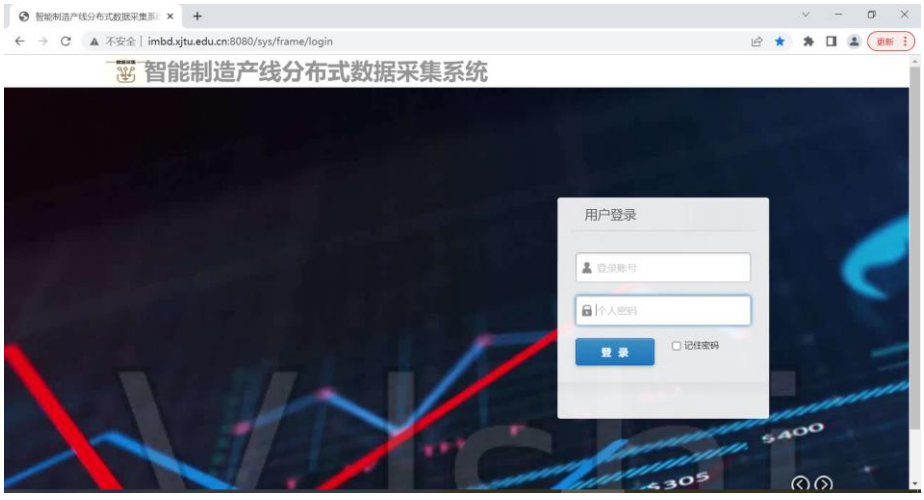
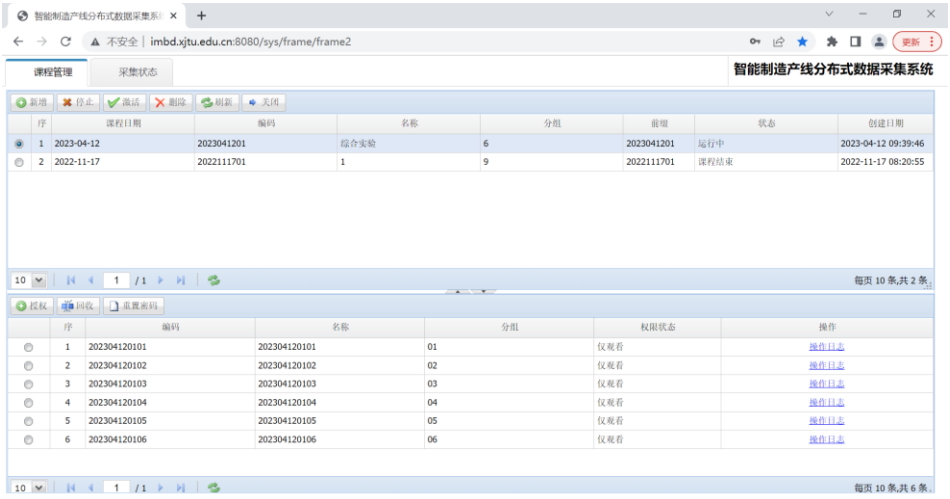


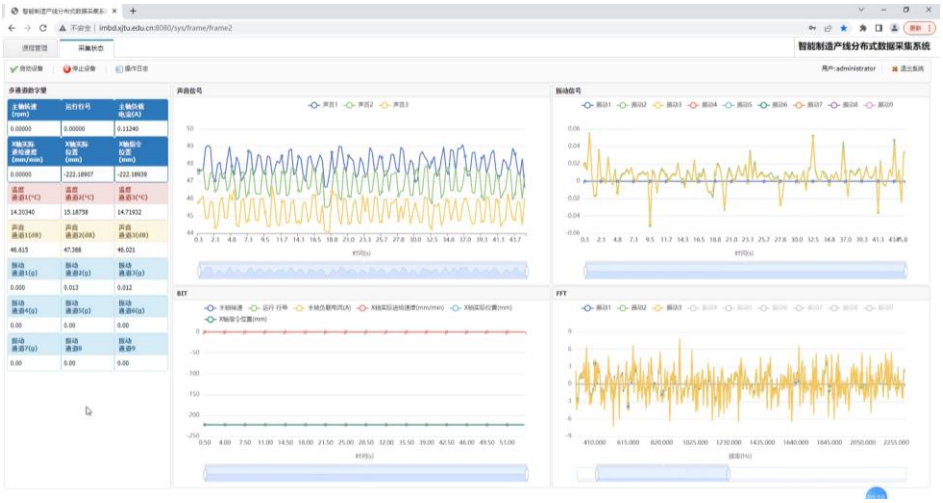
图 6 数据采集端软件主界面



（a）登录页



(b)课程管理界面



（c）数据采集显示界面

图 7 web 部分界面

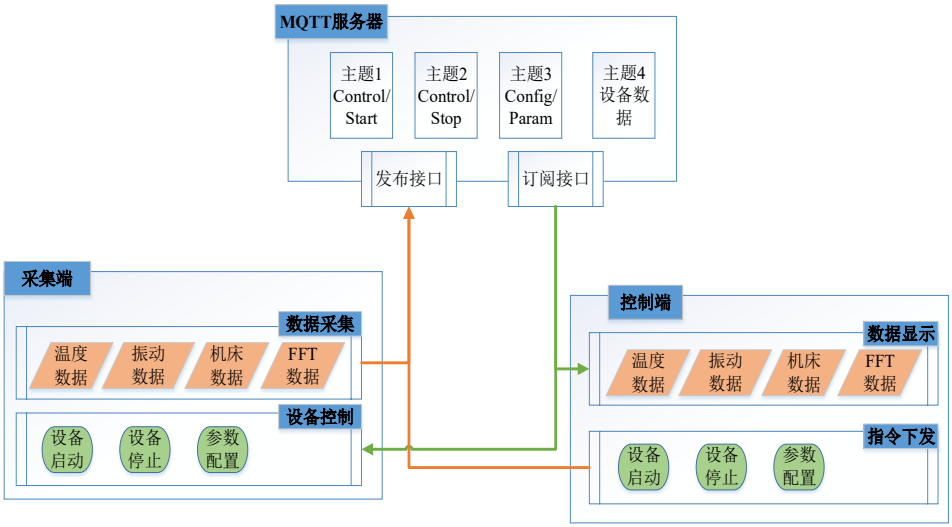


图 8 web 与数据采集端通信

（2）智能制造产线云边数据采集与存储系统

系统架构如图 9，硬件上包括数控机床、边缘服务器、交换机等。边缘服务器与数控机床通过局域网建立通信，基于 Nc-Link、MQTT 协议等，实现数据获取与传输，MQTT 服务运行于边缘服务器上，同时，边缘服务器上建立 Influxdb 时序数据库，通过 MQTT 服务实时订阅数据存储至时序库中。同时，与云端时序数据库通信，实现数据的云端同步存储。边缘服务器连接与校园网中，可远程获取数控系统数据，如图 10 所示。

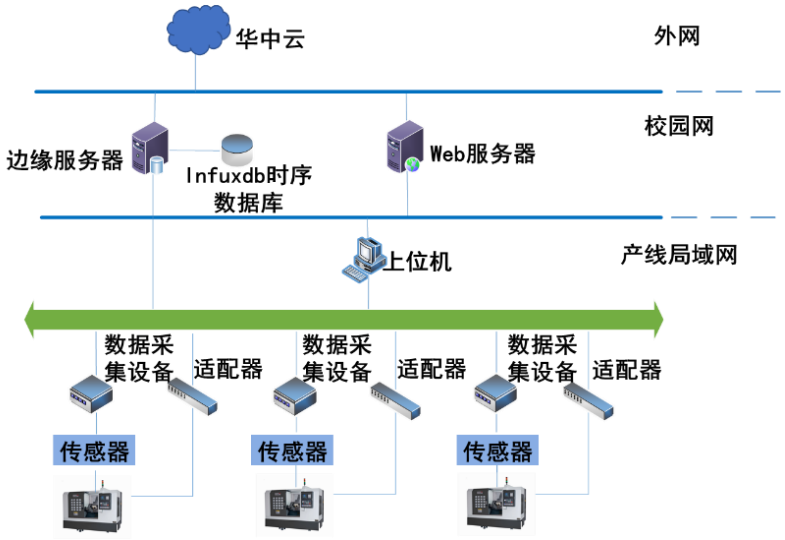


图 9 设备网络架构

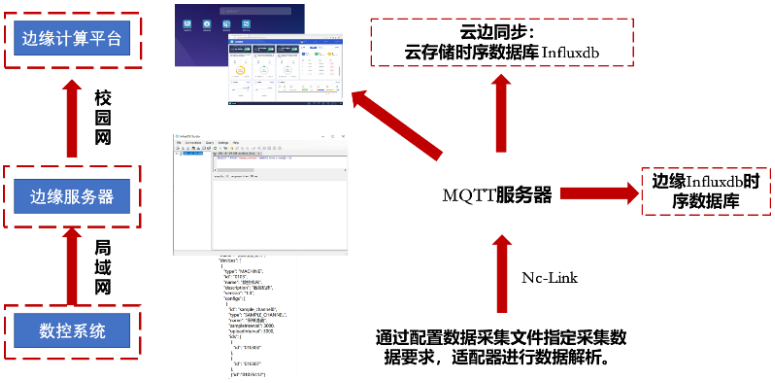


图 10 云边数据采集系统架构

3. 基于 MQTT 的数据传输原理

MQTT 是一种常用的基于发布/订阅模式（Publish/Subscribe）的数据传输方式，提供了传统客户/服务端模式 Client/Server 的一种替代方案。发布/订阅将发送消息的发布者（Publisher）与接收消息的订阅者（Subscribers）分离，两者不直接进行通信，甚至彼此之前都不知道对方的存在。发布者与订阅者之间的连接是由代理者（Broker）进行处理，代理者的工作是过滤所有发布者传入的消息，并且将这些消息正确的分发给订阅者，如图 11 所示。MQTT 协议中有三种身份：发布者（Publish）、代理（Broker）（服务器）、订阅者（Subscribe）。消息的发布者和订阅者都是客户端，消息代理是服务器，消息发布者可以同时是订阅者。MQTT 传输的消息分为：主题（Topic）和负载（payload）两部分：Topic：消息的类型，订阅者订阅（Subscribe）后，就会收到该主题的消息内容（payload）；Payload：消息的内容，是指订阅者具体要使用的内容。采用 python 实现 MQTT 通信的流程如图 12 所示，主要包括创建客户端、连接服务器、订阅/发布消息、监听消息等。

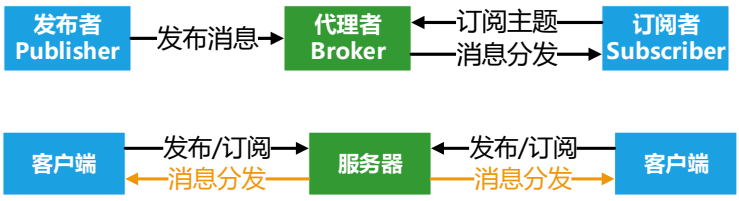


图 11 MQTT 原理

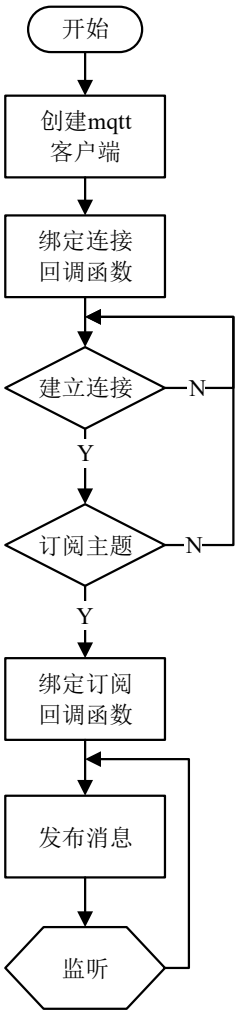


图 12 MQTT 通信开发流程

4. 数据库原理

数据库(DataBase, DB)是指长期储存在计算机内、有一定组织的、统一管理的、相关数据的集合。数据库中的数据按照某种数据模型组织、描述和存储，具有较小的冗余度、较高的数据独立性和易扩展性，并能为各种用户共享。数据库管理系统(DataBase Management System, DBMS)是指位于用户与操作系统(Operating System, OS)之间的数据管理软件。它为用户提供访问数据库的方法，包括数据库的创建、查询、更新、删除等。数据库应用系统的层次结构如图 13 所

示。通常所说的各类数据库（如 Oracle，MySQL）多指对应的数据库管理系统。

数据库种类众多，基于各类数据库特点和项目实际需求，本项目主要使用 InfluxDB 时序数据库、MySQL 数据库。

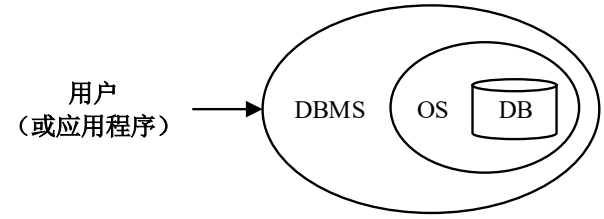


图 13 数据库应用系统的层次结构

InfluxDB 是一种流行的开源时序数据库，专门设计用于处理时间序列数据。InfluxDB 具有高效存储和查询的能力，采用自适应压缩算法和特定存储引擎，可以高效地存储大量的时序数据，并通过类 SQL 的查询语言（InfluxQL）提供丰富的查询功能，支持高并发写入，适用于处理实时数据流，并可定义数据保留策略自动删除旧数据以控制数据库大小。此外，InfluxDB 拥有插件生态系统，可扩展其功能，并具备高可用性和容错性，支持数据复制和故障转移。

InfluxDB 以组织(org)为最高级存储单位，每个组织下可以有多个数据桶(bucket)，每个数据桶内可存储任意类型的数据。与 MySQL 不同，InfluxDB 只需提前建立数据桶，而无需对每个数据桶进行设计。同时，InfluxDB 可以直接使用 API token 来确定访问用户的权限，而无需用户名和密码。因此可以为不同操作类型设置满足功能要求的最小权限 token，以保障数据安全。

采用 python 或 node 实现对 InfluxDB 的访问和操作，其写入数据流程如图 14 所示。

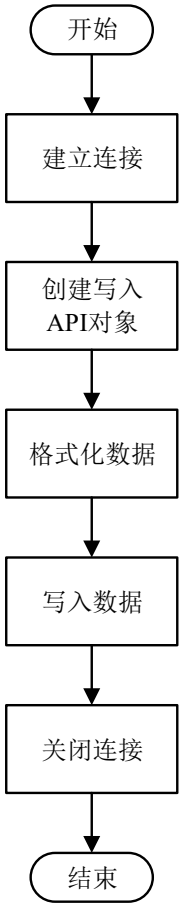


图 14 InfluxDB 数据库操作流程

MySQL 是一个关系型数据库管理系统，建立在关系模型的基础上，使用相互联系的二维表存储数据，表内每行存储一条信息。同时，MySQL 使用用户名和密码来识别用户身份，确定操作权限。MySQL 在储存数据前须首先进行数据库配置，创建数据库和表格并指明表格每列（字段）的数据类型。MySQL 使用 SQL（Structured Query Language，结构化查询语言）操作数据库，SQL 是具有数据操纵和数据定义等多种功能的数据库语言，具有交互性特点，能为用户提供极大的便利。

使用 Python 的 pymysql 库实现对 MySQL 数据库的访问和操作，其基本流程如图 15 所示。程序使用游标对象执行 SQL 语句，实现对数据库的增删改查。对查询操作，须获取返回值，以获得查询结果；对增删改操作，须提交操作，以完成对数据库的修改。

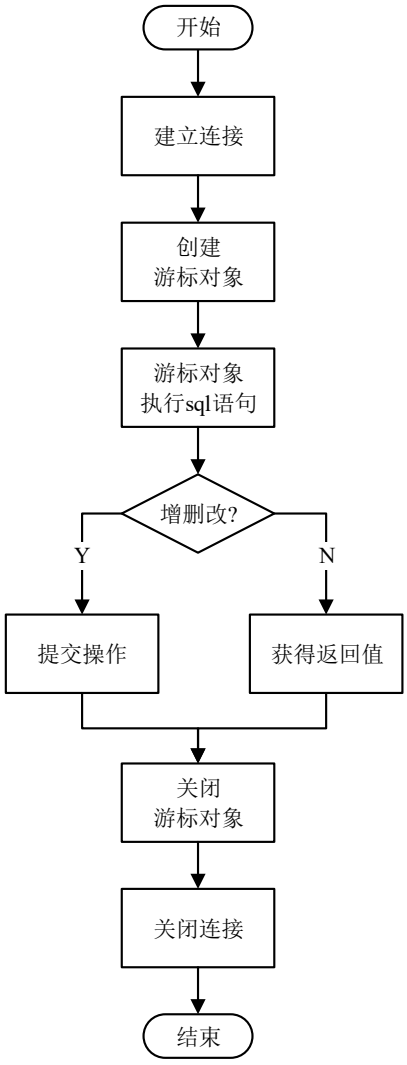


图 15 MySQL 数据库操作流程

5. 工业大数据平台

工业大数据平台是工业大数据存储、管理、计算的核心，通过先进的大数据技术，实现数据的分布式存储、索引、查询，以及归一化、冗余剔除、异常检测等预处理，并提供管理界面，对平台的运行进行监控和维护，架构如图 16。

图 16 工业大数据平台架构

平台基于开源的 Hadoop 架构，Hadoop 按位存储和处理数据具有非常高的可靠性。Hadoop 是在可用的计算机集簇间分配数据并完成计算任务的，这些集簇可以方便地扩展到数以千计的节点中。Hadoop 能够在节点之间动态地移动数据，并保证各个节点的动态平衡，处理速度非常快。Hadoop 能够自动保存数据的多个副本，并且能够自动将失败的任务重新分配。因此，平台具有高可靠性、高扩展性、高效性、高容错性。

大数据存储中心采用分布式文件系统 HDFS 和分布式数据库 HBase，如图 17，把数据分布存储到私有云平台提供的多个计算机节点上。该模式具有成本低、支持海量数据、扩展性好等优点。存储中心提供数据查询、数据管理界面，采用 Hive HQL 作为数据操作语言，可对数据进行查询、修改、删除等操作。

图 17 分布式文件系统 HDFS

本项目主要采用工业大数据平台实现实时获取数据的分布式存储以及数据的增删改查等。

6. 开发框架——Vue

Vue 是用于构建用户界面的 JavaScript 框架，可以通过简洁的模板语法，声明式地描述 HTML 输出；可以通过数据绑定和依赖追踪，自动检测数据变化，快速更新视图；可以通过组件系统，将复杂的用户界面拆分成可复用的组件，提高开发效率；可以与第三方库或现有项目结合使用，也可以用来开发完整的单页应用。

Vue 应用框架被设计为自底向上逐层应用，如图 18，依次与 API 接口、网络服务、第三方库、页面容器等建立连接，并通过各页面 JS 模块进行底层编译，

最终通过浏览器 HTML 网页框架搭建，CSS 界面渲染及 JavaScript 脚本语言生成客户端网页界面。

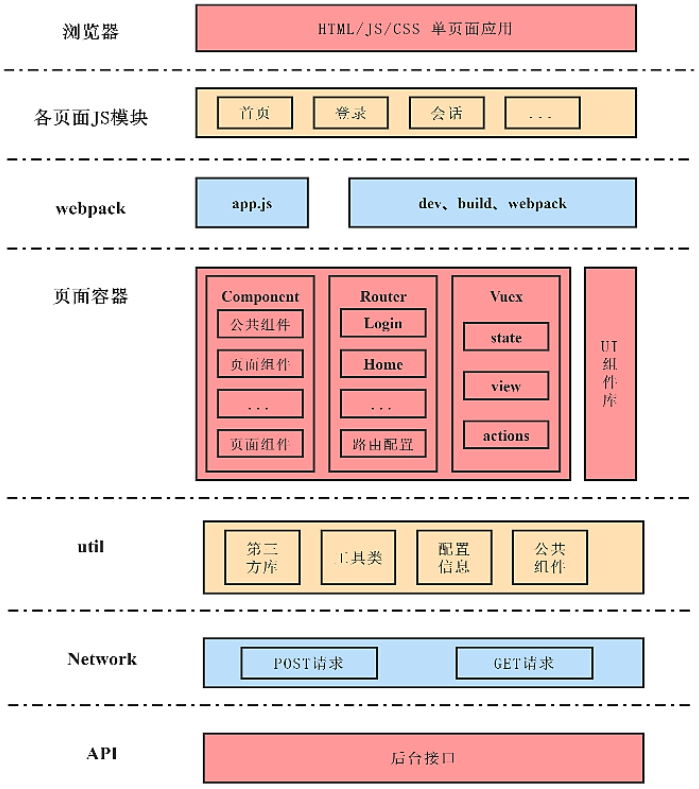


图 18 Vue 架构

四、项目任务

项目任务主要包括 7 部分，如图 19 所示。

1. 智能制造产线大数据采集与分析系统框架设计与开发

了解大数据采集与分析系统的框架结构和通信原理，基于 Vue 设计并搭建大数据采集与分析系统框架，创建项目文件以及欢迎界面、数据采集界面、数据可视化界面、数据分析界面等多个页面和后端文件，实现多个页面之间的切换，并根据 MQTT 通信原理，设计数据采集界面的基本元素。

2. 基于工业网络的数据获取

利用 Python 编写后端程序，与前端数据采集界面通信，读取 MQTT 服务器信息、MQTT 主题信息等，并连接 MQTT 服务器、订阅数据；

3. Web 网页的设计与数据可视化

设计数据可视化界面，建立与后端通信，解析获取到的数据，将数据发送至前端界面，实现数据可视化。

4. 基于数据库的数据存储与管理

创建边缘 influxdb 数据库和云端数据库，通过后端 python 或在 vue 中编写程序，连接 influxdb 服务，实现数据边缘存储和云存储。创建 MySQL 数据库，

编程实现数据在 MySQL 数据库的存储。

5. 大数据分布式存储与分布式计算

基于工业大数据平台开发数据通信接口，实现实时获取数据的分布式存储，并能够在平台中进行数据增删改查、可视化等。

6. 大数据分析与可视化

利用信号处理和人工智能算法分析设备运行数据，实现设备状态预测。

7. 分析结果的集成与可视化

将数据分析模型集成到大数据采集与分析系统，能够在网页中调用分析模型，进行设备状态预测，并将分析结果显示出来。

图 19 实验任务

五、系统开发架构与实验工具

本次实验基于 VUE 架构开发，前端界面基于 JavaScript 语言，后端采用 python 开发。后端系统通过 MQTT 协议通信获取设备，相当于开发一个 MQTT 客户端。MQTT 服务器依赖于设备，一般可运行在与设备连接的边缘服务器上。本次实验采用虚拟的 MQTT 服务器，替代设备上运行的 MQTT 服务器，该虚拟 MQTT 服务器基于五轴加工中心数据开发，与实际设备的数据类型、格式完全一致。整个架构如图 20 所示。MQTTX 软件是一个测试 MQTT 通信的软件，在本实验项目中，仅用于测试 MQTT 服务器是否运行正常，相当于一个 MQTT 客户端用来连接 MQTT 服务器，通过订阅和发布消息测试 MQTT 服务器是否正常运行。后端与前端通信，要求将获取的数据显示在界面中。同时，将数据存储至 influxdb 数据库和 MySQL 数据库中，基于 MySQL 数据库将数据上传至工业大

数据平台中，实现分布式存储。设备故障预测算法采用提供的数据训练模型，在输入新数据时，能够调用模型，实现设备状态预测，并将结果显示在界面中。

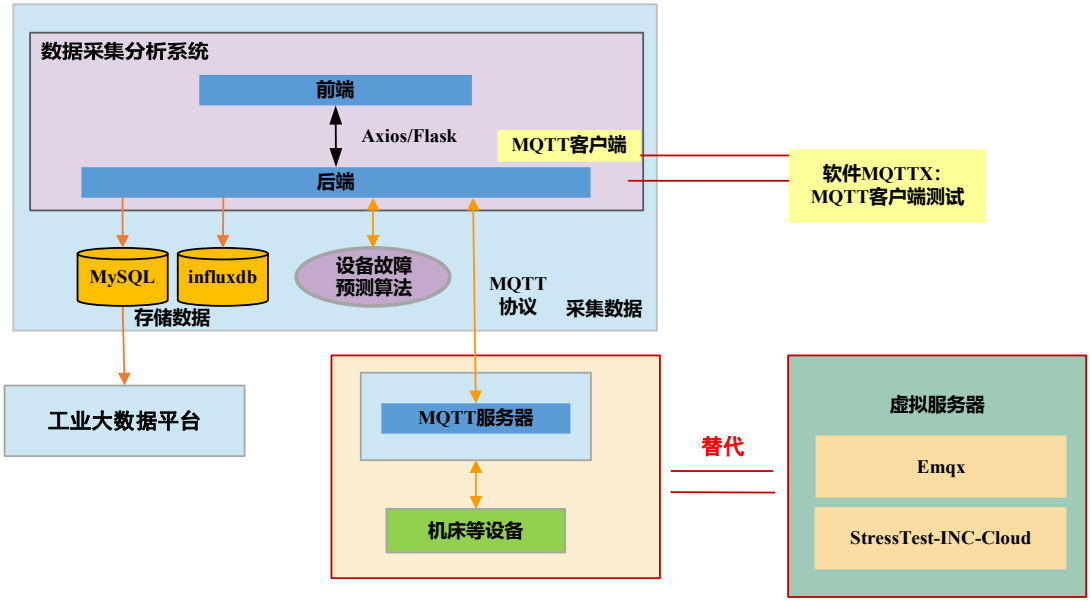


图 20 系统架构及实验工具