北京航空航天大学计算机学院

本科生毕业设计（论文）中期报告

论文题目：工业产线数据采集与管理系统设计与实现

学生姓名： 叶 静 波

学生学号： 17373176

专 业： 计算机科学与技术

指导教师： 沃 天 宇

学院(系)： 计 算 机 学 院

北京航空航天大学计算机学院

**2021年4 月5日**

**本科生毕业设计(论文)中期报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 学生姓名 | 叶静波 | 学号 | 17373176 | 专业班级 | 170614 |
| 指导教师 | 沃天宇 | 职称 | 副教授 | 单 位 | 北京航空航天大学计算机学院 |
| 题目类型 | □理论研究 ☑应用研究 □综述 □其它  如选“其它”，请自定义题目类型 | | | | |
| 题目来源 | ☑校内 □校外 | | | | |
| 毕业设计（论文）题目 | 工业产线数据采集与管理系统设计与实现 | | | | |
| **中期报告（阐述课题的目的、研究计划和研究方案、已完成的任务、下一阶段任务、参考文献等）**   1. **研究目的**   本课题的来源项目为“工业机器人云平台”，项目的最终目标是面向工业机器人智能运维需求，利用机器人生命周期中产生的数据进行可视化分析，突破多源时序数据的序列耦合、关联匹配和模式识别等关键技术，实现工业机器人的远程监控，并建立故障诊断和预测性维护模型，开发工业机器人智能运维云服务。要想实现分析预测的准确和及时，就必须有庞大的数据集作为模型建立的基础，因此设计和实现一个可靠、高性能、实时的云数据采集系统是该项目的关键。  工业产线设备协议的多样性以及本地监控系统的低维护性给工业数据采集、数据管理、数据安全和统一监控带来了难题。当今的工业产线领域中，OPC（OLE for Process Control）协议已经成为了一种标准。但是基于DCOM（Distributed Component Object Model）的OPC DA技术并不支持数据在Internet上的传输，使得在某些应用中无法满足要求。OPC UA（OPC Unified Architecture）协议由OPC发展而来，是对OPC的继承与升级，解决了传统OPC对Windows平台的依赖以及对分布式系统信息交换不适用等主要问题[1]。该规范规定了不同软件产品间进行数据交换的标准通信接口，将OPC UA运用到数据采集系统[2]，解决了系统因为工业设备通讯协议多样带来软件开发复杂多样的困难。OPC UA协议具有安全性高，效率高的特点。现代工业产线的数据采集和传输已经不再是单一、孤立的信息交换了，而是形成一种工业网络，网络中的任何设备之间都可以进行信息传输和共享，工业网络通过Internet网络进行连接,同时TCP/IP等开放性协议在工业网络中也广泛使用，使得工业网络安全性问题日益凸显。与传统的OPC相比，OPC UA还有一大优点就是官方加入了安全规范，OPC UA 规范支持数据在Internet 上远程传输[3]。  工业产线设备采集的数据具有时序数据[4]的鲜明特征，如数据和时间密切相关且绝大多数请求为写入请求，同时工业设备采集的数据大量性，造成数据访问低效率、延迟、低可靠性的问题，为此采用Kafka集群中间件、Mysql数据库和Redis缓存作为云端存储系统构建的基础，同时云端采用时序数据库InfluxDB作为离线有序数据分析的数据库。  本研究面向基于云平台的工业机器人远程监控需求。拟设计与实现面向工业机器人的数据采集与管理系统，实现多协议集成的数据交换接口，优化数据库性能，最后配合前端技术进行测试验证，满足工业机器人监控的实时性、云采集系统的可靠性和高性能要求。   1. **研究计划与方案**   1. 1月1日~3月1日：确定毕业设计具体内容及关键技术，阅读相关资料和文献，熟悉项目开发流程、后端架构及OPC UA协议，顺利操作运行工业机器人流水线，以及完成开题报告和答辩PPT。   1. 3月1日~5月10日：实现OPC UA客户端，部署数据库后端环境，部署OPCUA安全策略在客户端上；集成多协议数据采集接口；数据上云；集群管理和索引优化设计；结合前端技术对“产线”进行测试。 2. 5月11日~5月31日：展示运行效果，完成毕设论文并准备最终答辩。   **三、已完成工作**  **1.数据库设计**  使用消息队列接收边缘端数据的传输，然后持久化到本地磁盘，同时将热点数据保存到内存数据库，因此采用kafka消息队列可以实现高吞吐读写，对硬件驱动采集到的数据进行缓冲，为数据库存储、数据上云提供了数据来源。  从边缘端采集传输到消息队列的过程采用OPC UA、webSocket、MQTT协议，OPC UA客户端连接服务器进行传输，配置安全策略，进行证书认证和加密传输，保证了数据的安全性。  OPC UA协议规范里建立了安全通道，在客户端中，与服务器建立连接时会先建立安全通道open\_secure\_channel()，之后再根据用户名、密码和认证密钥建立会话。  设计数据库表，使用Django框架开发，建立数据库表：   * 数据库名称：local\_db * 默认端口：127.0.0.1:3306 * 用户名/密码：root/123456   该数据库存放关系数据，一个是产线机器基本信息，一个是机器运行坐标信息，由于数据量大，目前采用varchar32位存储。将Django默认数据库换为Mysql，创建数据库，数据库取名为local\_db，并设置字符集为utf-8：CREATE DATABASE local\_db CHARACTER SET utf8;  配置settings.py文件，配置Mysql数据库引擎安装mysqlclient库：pip install mysqlclient；执行同步操作python manage.py makemigrations, python manage.py migrate将数据更新到MySQL。  **2.模块具体实现**  （一）多协议采集模块  数据传输由边缘端主动发送过来，客户端采用订阅方式接收数据，因此客户端不需要管边缘端怎么发送和具体使用什么协议进行传输，只关心客户端自己支持的传输协议和安全机制，因此暴露接口给边缘端，与边缘端服务器建立连接。  1、OPC UA协议数据采集接口：此部分是采用OPC UA协议传输数据的接口，创建OPC UA客户端，与OPC UA服务器连接，进行数据订阅。  2、[集成多协议的数据采集接口](https://baike.baidu.com/item/%E7%A8%8B%E5%BA%8F%E6%8E%A5%E5%8F%A3" \t "https://baike.baidu.com/item/%E7%94%A8%E6%88%B7%E6%8E%A5%E5%8F%A3/_blank)：此部分在前面的基础上，加入了已实现的webSocket协议和MQTT协议的客户端，三个协议集成为一个接口，即抽象工厂模式。以便于边缘端服务器根据自己的硬件支持和需求在调用API时，通过携带相关的参数作为具体协议和安全机制的使用需求，进行数据传输。  该模块主要任务是接收设备边发送过来的数据，然后将数据放入本地单机kafka中，同时实现加密的安全传输。  创建OPC UA客户端，建立安全通道，创建会话，用户验证，认证证书和密钥，连接到OPC UA服务器端口，开始订阅消息。  安全策略：支持“无”、“Basic256Sha256”和“Basic128Rsa15”（OPC基金会不再将它视为安全）三种。  安全模式：当选择后两种安全策略时，安全模式需要选择“Sign”和“SignAndEncrypt”之一，同时需要上传数字证书和密钥证书。  相关配置：   * 用户名/密码：opcua/123456，可以对“匿名”标识符进行权限限制 * 安全策略：Basic256Sha256 * 安全模式：SignAndEncrypt * 证书/私钥：client\_cert.pem / client\_key.pem   使用OpenSSL生成CA证书，再使用CA证书生成服务器和客户端证书、私钥，客户端检验服务器证书，判定私钥是否正确：  client.set\_security\_string("Basic256Sha256,SignAndEncrypt," + cer\_path + "," + private\_key\_path + server\_cer\_path)  3.开启kafka服务，创建kafka生产者对象和消费者对象，这里采用双线程，分别代表生产者线程ProducerThread和消费者线程ConsumerThread。生产者将收到的数据根据key-value解析后送入kafka队列后，消费者从队列中拉取数据写入数据库；编写数据持久化操作函数，将消息队列里的数据存储在本地MySQL数据库中。  相关配置：   * num.partitions：默认分区数为1，单生产者和单消费者 * log.retention.hours：7天，log的保留时间，即写入的数据的过期时间   相关代码：  kafka\_producer = ProducerThread() kafka\_consumer = ConsumerThread() try:  kafka\_producer.start()  kafka\_consumer.start() finally:  kafka\_producer.terminate()  kafka\_consumer.terminate()  （二）数据库存取模块  数据库存取模块将存取操作封装成接口，便于数据库的管理，其中调用者分别为：前端request请求查询和下载、保存到本地，数据采集模块中数据持久化存储。  数据库的开发采用Django框架，model模型对应建立数据库表，view视图模块对应前端请求以及数据给前端展示，这些操作涉及到的数据库增删查改等函数都封装在该接口里。  数据库表-model：  class MachineInfo(models.Model):  id = models.CharField(max\_length=16, primary\_key=True) # 如果没有models.AutoField，默认会创建一个id的自增列  name = models.CharField(max\_length=32) # 操作名  value = models.CharField(max\_length=32) # 状态  update\_time = models.DateTimeField() # 操作时间，静态机器则为0  def save(self, \*args, \*\*kwargs):  self.update\_time = (self.update\_time.strftime('%Y-%m-%d %H:%M:%S')) # 自定义日期格式  super(MachineInfo, self).save(\*args, \*\*kwargs) # 调用父类save  class Meta:  db\_table = 'MachineInfo' # 表名  verbose\_name = "machineInfo"  数据库插入数据：  def insert\_machine(data):  machine = models.MachineInfo(id=id, name=name, value=value, update\_time=time)  machine.save()  前端访问数据接口：  def get\_data\_from\_db\_send\_to\_user(request):  """  API: 给前端调用获取数据的接口  @param request: 前端请求  @return:  """  **四、关键技术或难点**   1. **多数据库架构设计与实现**   后端采用kafka消息中间件，双线程实现数据推拉，持久化到MySQL数据库，同时将数据存到Redis缓存，需要确保确保数据读写一致性和无错性，同时不能发生死锁。   1. **集群管理机制及高效索引的建立**   Kafka和Redis集群化之后，需要设计和实现一个集群管理机制，使得对消息队列和Redis的读写能够保证无错性和一致性。同时，选择一种恰当的数据分片策略是提升数据库整体写入性能、缓解写入压力的关键。  **五、下一阶段计划**  （1）数据上云：将数据采集并上传到阿里云服务器；  （2）优化设计：kafka和Redis采用集群，提高读写性能；云端以时间为key的形式存储在influxDB数据库中，以便用于训练；数据库索引优化，进一步完成系统后端开发；  （3）测试：结合前端技术对“产线”进行测试   1. **预期结果**   将本文实现的技术结合已开发的前端技术形成数据云采集与可视化系统，对实验室“工业螺旋桨安装产线”进行测试，预期效果主要针对于系统的实时性、可靠性、易维护性以及高性能方面。系统能够准确实时地采集工业产线的数据，并进行解析处理、高效存储和上云，以便将数据用于可视化分析和实时监控。  **八、参考文献**   1. ZHAO Y H, NIE Y J, WANG Y L, et al. Overview of OPC UA technology[J]．Naval Chemical Defense, 2010(2): 33-37. 2. LI J X. Research on equipment data acquisition and remote monitoring system for intelligent factories[D]. Nanjing: Nanjing University of Aeronautics and Astronautics, 2018. 3. OPC UA Part 2 - Security Model 1.03 Specification[M]. OPC Foundation. 2015. 4. Wei W W S. Wei W W S . Time Series Analysis: Univariate and Multivariate Methods, 2nd edition[M]. New York: Pearson Education, 2006. 5. 许申声. 四轴机器人的OPC UA数据采集客户端开发及安全性研究[D]. 重庆: 重庆邮电大学, 2018. 6. LIU W. Research on MES data management system based on OPC UA [D]. Beijing:Beijing University of Posts and Telecommunications, 2019. 7. ZHANG C, WU M L, LU J Q, et al. Design of 828D CNC machine tool group remote monitoring system based on OPC UA [J]. Machinery Manufacturing & Automation, 2018, 47(6): 186-189． 8. XU B B. Design and implementation of key modules of data acquisition and monitoring system based on OPC UA [D]. Xi’an: Xi’an University of Science and Technology, 2017. 9. Mizuya T , Okuda M , Nagao T . A case study of data acquisition from field devices using OPC UA and MQTT[J]. 2017: 611-614. 10. 谢青松. 面向工业大数据的数据采集系统[D]. 湖北: 华中科技大学, 2016. 11. 禹鑫燚, 殷慧武, 施甜峰, 等. 基于OPC UA的工业设备数据采集系统[J]. 计算机科学, 47(11A): 609-614. | | | | | |
| **指导教师意见（课题难度是否适中、工作量是否饱满、进度安排是否合理、工作条件是否具备、是否同意开题等）：**  指导教师签名：  年 月 日 | | | | | |
| **学院（系）意见：**  审查结果： □ 同 意 □ 不 同 意  学院（系）负责人签名：  年 月 日 | | | | | |