 单位代码 **10006**

学号 **18373303**

分类号 **TP311.5**

****

毕业设计(论文)

遥感图像处理任务可视化编排系统

的设计与实现

|  |  |
| --- | --- |
| 学院名称 | 软件学院 |
| 专业名称 | 软件工程 |
| 学生姓名 | 张潇菡 |
| 指导教师 | 沃天宇 |

**2022**年**6**月

北京航空航天大学

**本科生毕业设计（论文）任务书**

Ⅰ、毕业设计（论文）题目：

遥感图像处理任务可视化编排系统的设计与实现

Ⅱ、毕业设计（论文）使用的原始资料（数据）及设计技术要求：

采用Vue、Django前后端技术构建交互平台，实现方便简单的遥感图像任务编排。依赖Docker构建自动部署调度平台，将已有的镜像进行部署构建。借助Socket构建消息文件传输平台，实现可靠的消息传输

Ⅲ、毕业设计（论文）工作内容：

基于可视化技术，采用DAG对遥感图像处理相关算法的依赖关系进行编排建模，构建容器管理平台，实现后台自定义调度部署，提高遥感图像任务云上部署和调度的效率。主要内容可划分为可视化任务编排的设计与实现、云平台自定义调度与部署的设计与实现

Ⅳ、主要参考资料：

[1] Matsunobu Lysha M., Pedro Hugo T.C., Coimbra Carlos F.M.. Cloud detection using convolutional neural networks on remote sensing images[J]. Solar Energy,2021,230.

[2] 常志明. 遥感图像并行算法的研究及其网格服务的实现[D].国防科学技术大学,2004.

[3] 郭延辉. 高光谱遥感图像分类技术研究[D].陕西师范大学,2020.DOI:10.27292/d.cnki.gsxfu.2020.000010.

[4] 赵忠明,高连如,陈东,岳安志,陈静波,刘东升,杨健,孟瑜.卫星遥感及图像处理平台发展[J].中国图象图形学报,2019,24(12):2098-2110.

[5] Aliyun. AI Earth[EB/OL]. <https://engine-aiearth.aliyun.com/#/,>2022-04-05.

[6] SenseTime. SenseEarth[EB/OL]. <https://rs.sensetime.com/#tec-s-title,>2022-04-05.

[7] 城市数据派. 商汤科技SenseEarth平台:遥感影像实时智能解译[EB/OL]. <https://www.sohu.com/a/326117105_650480>,2022/5/3.

[8] 李乔,郑啸.云计算研究现状综述[J].计算机科学,2011,38(04):32-37.

[9] Kivity A, Kamay Y, Laor D,et al. kvm: the Linux virtual machine monitor[C]. Linux Symposium.2007.

[10] Al I P. Xen 3.0 and the Art of Virtualization[J]. proceedings of the ottawa linux symposium,2005.

[11] Docker Inc. Home - Docker[EB/OL]. <https://www.docker.com/>,2022-04-01.

[12] Brian Goff. CHANGELOG --rocket[EB/OL]. <https://rocket.readthedocs.io/en/latest/CHANGELOG/>,2022-04-01.

[13] 金子威. 基于K8S的Docker分布式容器自动化运维系统的设计与实现[D].中南民族大学,2018.

[14] Deepak Vohra. Docker Management Design Patterns[M]. Apress, Berkeley, CA: Apress,2017,45.

[15] The Kubernetes Authors. Kubernetes Documentation[EB/OL]. <https://kubernetes.io/docs/home/>,2022-04-27.

[16] The Apache Software Foundation. Apache Mesos - Documentation Home[EB/OL]. <https://mesos.apache.org/documentation/latest/>,2022-2-20/2022-04-11.

[17] 秦志威,栗娟,刘晓,朱梦圆.端边云协同环境下能耗感知的工作流实时调度策略[J/OL]. http://kns.cnki.net/kcms/detail/11.5946.TP.20211108.1227.008.html,2021-11-08/2022-4-10.

[18] The Apache Software Foundation. Apache Airflow Documentation[EB/OL]. <https://airflow.apache.org/docs/apache-airflow/stable/index.html>,2022-03-14.

[19] The Luigi Authors. Luigi Documentation[EB/OL]. [https://luigi.readthedocs.io,](https://luigi.readthedocs.io)2022-04-05.

[20] Argo Project Authors. Argo Workflows[EB/OL]. <https://argoproj.github.io/workflows/>,2022-3-14/2022-04-15.

[21] 孙庚泽.基于Docker的混合云应用编排方案研究[D].西安电子科技大学,2015.

[22] 高畅.大数据可视化编排与展示系统的设计与实现[D].北京邮电大学,2018.

[23] 郭爱春,李洪全,丁浩,商和龙.基于可视化云编排技术的智能终端业务应用[J].电工技术,2021(22):161-163.DOI:10.19768/j.cnki.dgjs.2021.22.054.

[24] Pradipta Ghosh, et al. Container Orchestration for Dispersed Computing[A]. Proceedings of the 5th International Workshop on Container Technologies and Container Clouds[C]. Davis, CA, USA: Association for Computing Machinery, 2019: 19–24.

[25] 开彩红,肖瑶,方青.基于人工蜂群算法的中继卫星任务调度研究[J].电子与信息学报,2015,37(10):2466-2474.

[26] 黄科.基于微服务的虚拟机自动化编排系统的设计与实现[D].电子科技大学,2019.

[27] 姚婧,宋锐,王壮,胡卫东.空间探测相控阵雷达系统的观测任务编排仿真[J].计算机仿真,2006(10):65-68+71.

[28] 刘伟权. 野外WSN的监测系统任务编排及调度的实现[D].西安工业大学,2015.

[29] X. M. Li, W. X. Wang, S. J. Tang, et al. A NEW CLOUD-EDGE-TERMINAL RESOURCES COLLABORATIVE SCHEDULING FRAMEWORK FOR MULTI-LEVEL VISUALIZATION TASKS OF LARGE-SCALE SPATIO-TEMPORAL DATA[J]. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 2020, XLIII-B4-2020:477-483.

[30] Eliseu Germano, et al. Workflow Management Platform for Orchestration of Ubiquitous Care Plan Services[A]. Proceedings of the Euro American Conference on Telematics and Information Systems[C]. Fortaleza, Brazil: Association for Computing Machinery, 2018: 8.

[31] 孙锐.鲁棒性\_百度百科[EB/OL]. [https://baike.baidu.com/item/鲁棒性/832302,](https://baike.baidu.com/item/%E9%B2%81%E6%A3%92%E6%80%A7/832302) 2022-04-23.

[32] 李臣亮.事件驱动架构及应用[J].软件世界,2007(21):44-45.

[33] 科睿思博.软件架构分类(转载) - 博客园[EB/OL]. <https://www.cnblogs.com/mikechang/p/5838845.html>, 2022-05-13.

[34] 赵然,朱小勇.微服务架构评述[J].网络新媒体技术,2019,8(01):58-61+65.

[35] 王湘文,陈建伦,陈纪铭.分层软件架构设计及其应用研究[J].福建电脑,2011,27(06):55-56+93..

[36] 廖志英,董安邦.基于C/S和B/S混合结构的管理信息系统运行模式[J].计算机工程与应用,2002(02):184-185+249.

[37] 李云云.浅析B/S和C/S体系结构[J].科学之友,2011(01):6-8.

[38] 籍慧文.B\S和C\S的架构分析[J].电子技术与软件工程,2017(05):202.

[39] S. Priyank,V. G. Silva, M. Kirikova , et al . An analysis of container-based virtualization technologies[J]. Asian Journal of Multidimensional Research,2021,10-11.

[40] Yuqian Lu, Xun Xu, Jenny Xu. Development of a Hybrid Manufacturing Cloud[J]. Journal of Manufacturing Systems,2014.

[41] Sitaram D, Harwalkar S, Sureka C, et al. Orchestration Based Hybrid or Multi Clouds and Interoperability Standardization [C]. 2018 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), November 23-24, 2018, Bangalore,India, IEEE 2018:67-71.

[42] Imran ,Ahmad Shabir,Kim Do Hyeun. A task orchestration approach for efficient mountain fire detection based on microservice and predictive analysis in IoT environment[J]. Journal of Intelligent & Fuzzy Systems,2021,40(3).

[43] Klein Peter,Mäuselein Maren,Cetin Nurhan. Development of innovative visualization and interaction concepts for dispatcher software by means of the user-centered design process[J]. VDI Berichte,2008.

[44] 蔡凌峰,魏祥麟,邢长友,邹霞,张国敏.故障场景下的边缘计算DAG任务重调度方法[J].计算机科学,2021,48(10):334-342.

[45] 齐舸. 异构云平台下多DAG工作流的节能调度策略研究[D].西安电子科技大学,2020.DOI:10.27389/d.cnki.gxadu.2020.000777.

[46] Wenzhong Zhu,Hualong Jiang,Shuwen Zhou,Mike Addison. The Review of Prospect of Remote Sensing Image Processing[J]. Recent Patents on Computer Science,2017,10(1).

[47] Stupina A , Zhukov E A , Ezhemanskaya S N , et al. Question-answering system[J]. Iop Conference, 2016, 155:012024.

[48] 伍尚锋,李洪赭,李赛飞.一种多云环境中应用编排方案设计[J].微型电脑应用,2021,37(05):1-4.

软 件 学院 软件工程 专业 182113 班

学生 张潇菡

毕业设计（论文）时间： 2021 年 月 日至 2022 年 月 日

答辩时间： 2022 年 6 月 日

成 绩：

指导教师：

兼职教师或答疑教师（并指出所负责部分）：

21 系（教研室） 主任（签字）：

注：任务书应该附在已完成的毕业设计（论文）的首页。

**本人声明**

我声明，本论文及其研究工作是由本人在导师指导下独立完成的，在完成论文时所利用的一切资料均已在参考文献中列出。

作者：张潇菡

签字：

时间：2022 年 6 月 日

遥感图像处理任务可视化编排系统的设计与实现

学生：张潇菡

指导教师：沃天宇

摘要

云计算的快速发展在一定程度上为数据量爆炸增长的遥感图像处理提供了有效的解决方案，然而遥感图像处理流程编排和调度面临诸多问题，主要有计算机专业知识要求高，运维难度陡增，开发效率低等问题。针对传统的通过编写脚本文件进行任务编排存在专业知识要求高、工作量冗余的问题，提出可视化云编排理念。本文实现可视化编排，将业务和技术解耦，并实现遥感图像任务云上自动部署和构建，提高遥感图像任务编排效率和云平台资源利用率。

为实现上述目标，本文结合C/S架构事务能力处理强的优点和B/S架构下网络易扩性和分布式的特点，构建C/S架构和B/S架构交叉使用的三层混合部署结构。B/S架构面向用户，使用SVG画布实现了可视化编排和可视化调度，简化用户的编排操作。C/S架构下的服务端负责云平台的维护，通过监听数据库中数据的变化创建服务，将服务调度到多个客户端上并发运行，利用云的强算力，加快遥感任务处理性能，增强遥感任务云上部署和调度的效率。

本文完成编码实现，并最终上线调试，结果表明本设计能实现任务的可视化编排和云上自动部署和构建，经过了全面的测试，达到了预期的效果，可支撑遥感图像处理领域专业人员高效地开展新流程开发和业务重构。

关键词： 遥感图像，可视化，云平台，混合架构

英文标题

Author : Xiaohan Zhang

Tutor : Tianyu Wo

Abstract

The rapid development of cloud computing has provided an effective solution for [processing](D:/Projects/youdao/Dict/8.10.3.0/resultui/html/index.html" \l "/javascript:;) [of](D:/Projects/youdao/Dict/8.10.3.0/resultui/html/index.html" \l "/javascript:;) [remote](D:/Projects/youdao/Dict/8.10.3.0/resultui/html/index.html" \l "/javascript:;) [sensing](D:/Projects/youdao/Dict/8.10.3.0/resultui/html/index.html" \l "/javascript:;) [image](D:/Projects/youdao/Dict/8.10.3.0/resultui/html/index.html" \l "/javascript:;) [data](D:/Projects/youdao/Dict/8.10.3.0/resultui/html/index.html" \l "/javascript:;) with the explosive growth. However, there are many problems in the process choreography and scheduling of remote sensing image processing, mainly including high requirements for computer professional knowledge, steep increase in the difficulty of operation and low development efficiency. In view of the problems of high professional knowledge requirements and redundant workload in the traditional task scheduling, the concept of visual cloud scheduling is proposed. This paper implements visual choreography to decouple business and technology, and realize automatic deployment and construction of remote sensing image task on cloud, so as to improve the scheduling efficiency of remote sensing image task and resource utilization of cloud platform.

In order to achieve the above objectives, this paper combines the advantages of strong transaction capability of C/S structure and the characteristics of extendibility and distribution of network of B/S structure and construct a three-layer hybrid deployment structure with the cross-use of C/S structure and B/S structure. B/S structure is user-oriented and uses SVG canvas to realize visual choreography and scheduling, simplifying user's choreography operations. The server in the C/S structure is responsible for the maintenance of the cloud platform.It creates services by monitoring the changes of data in the database and schedules the services to multiple clients for concurrent operation. The strong computing power of cloud is utilized to speed up the processing performance of remote sensing task and enhance the efficiency of remote sensing task deployment and scheduling on cloud.

This paper has completed the design and implementation of visual arrangement system of remote sensing image processing task, which can efficiently and simply complete the remote sensing image processing. Through comprehensive testing, it verified the good security, robustness and scalability of the system. This paper achieved the desired effect and can support professionals in the field of remote sensing image processing to efficiently develop new processes and business reconstruction.

Key words: remote sensing image, visualization, cloud platform, hybrid architecture

目录

[1 绪论 1](#_Toc20308)

[1.1 课题背景与意义 1](#_Toc31851)

[1.1.1 课题背景 1](#_Toc3895)

[1.1.2 课题意义 3](#_Toc24651)

[1.2 国内外研究现状 3](#_Toc2452)

[1.2.1 传统任务编排系统 3](#_Toc19805)

[1.2.2 通用任务编排系统的研究 5](#_Toc17283)

[1.2.3 特定任务编排系统的研究 5](#_Toc23361)

[1.3 研究目标与内容 6](#_Toc10598)

[1.3.1 研究目标 6](#_Toc13067)

[1.3.2 研究内容 6](#_Toc23893)

[1.4 论文组织结构 6](#_Toc24186)

[2 遥感图像处理任务可视化编排系统需求分析 8](#_Toc6640)

[2.1 总体需求分析 8](#_Toc11784)

[2.2 功能性需求 9](#_Toc21696)

[2.2.1 用户管理模块 9](#_Toc21579)

[2.2.2 客户端管理模块 9](#_Toc25732)

[2.2.3 可视化遥感图像处理模块 10](#_Toc2426)

[2.3 非功能性需求 11](#_Toc21287)

[2.3.1 性能需求 11](#_Toc24605)

[2.3.2 易用性需求 11](#_Toc55)

[2.3.3 可扩展性需求 12](#_Toc31785)

[2.3.4 安全性需求 12](#_Toc8193)

[2.3.5 鲁棒性需求 12](#_Toc10269)

[3 遥感图像处理任务可视化编排系统总体设计 13](#_Toc17615)

[3.1 技术架构设计 13](#_Toc24534)

[3.1.1 技术架构设计方案选择 13](#_Toc19162)

[3.1.2 遥感图像处理任务可视化编排系统技术架构设计 14](#_Toc26421)

[3.2 系统功能模块设计 16](#_Toc1251)

[3.2.1 用户管理模块的功能 17](#_Toc29390)

[3.2.2 客户端管理模块的功能 17](#_Toc13978)

[3.2.3 可视化遥感图像处理模块的功能 17](#_Toc23354)

[3.3 系统部署架构设计 18](#_Toc4925)

[3.3.1 部署架构设计方案选择 18](#_Toc22463)

[3.3.2 遥感图像处理任务可视化编排系统部署架构设计 19](#_Toc27146)

[3.4 数据库设计 20](#_Toc25447)

[3.4.1 用户管理模块的数据库设计 21](#_Toc14211)

[3.4.2 客户端管理模块的数据库设计 21](#_Toc18377)

[3.4.3 可视化遥感图像处理模块的数据库设计 21](#_Toc18836)

[4 遥感图像处理任务可视化编排系统详细设计与实现 27](#_Toc25354)

[4.1 部署架构的设计与实现 27](#_Toc22039)

[4.1.1 B/S模式架构搭建 27](#_Toc10968)

[4.1.2 C/S模式架构搭建 28](#_Toc7755)

[4.2 功能模块的设计与实现 31](#_Toc15485)

[4.2.1 用户管理模块 31](#_Toc20283)

[4.2.2 客户端管理模块 31](#_Toc2610)

[4.2.3 可视化遥感图像处理模块 33](#_Toc3627)

[5 遥感图像处理任务可视化编排系统测试和评估 43](#_Toc30808)

[5.1 测试目标 43](#_Toc22905)

[5.2 测试环境 43](#_Toc4428)

[5.3 功能性测试 43](#_Toc27041)

[5.3.1 用户管理测试 43](#_Toc27877)

[5.3.2 客户端管理测试 44](#_Toc23362)

[5.3.3 可视化遥感图像处理测试 45](#_Toc4322)

[5.4 非功能性测试 47](#_Toc29262)

[结论与展望 49](#_Toc31188)

[致谢 51](#_Toc2005)

[参考文献 52](#_Toc14406)

# 绪论

## 课题背景与意义

### 课题背景

随着遥感探测设备、遥感平台、数据通讯及其有关科技的发展，从现代遥感技术收集到的遥感数据数量呈现爆炸性增加的趋势，提高了遥感数据的光谱、空间和时间分辨率[1]，处理遥感数据实现了快速、动态和精准的高水平目标[2]，遥感图像数据处理能力也朝着高分辨率、多波段的数据处理方向发展[3]。数据量的急剧增加、遥感应用的迅速拓展对海量遥感图像数据的处理效率提出了极大的挑战，也对图像处理工作提出更高的要求：即缺乏专业能力的人员借助极少外力帮助，可以快速、高效完成复杂的图像处理过程[4]。

传统方法已经无法有效分析和处理海量的遥感图像数据，随着云计算的迅速发展，利用云来处理遥感图像数据已经成为趋势。AI Earth[5]提供遥感、气象等多源对地观测数据的云计算分析服务，用数据感知地球世界。SenseEarth[6]实现了遥感影像可视化云解译平台，以其高精度、高效率的强大计算能力，克服了以往卫星影像重访率低且解译技术时效性低的缺点[7]。云计算对存储和网络设备实现了虚拟化，通过互联网提供按需共享、动态和可伸缩的分布式服务。以管理人员管理技术资源、运维云平台的形式，云平台向遥感图像专业人士屏蔽底层细节，不受时间和地点的约束，向用户提供一站式云服务，实现云计算资源池性能最大化[8]。

云计算的应用依托于虚拟化平台运维。目前的虚拟化技术，主要分为以KVM(Kernel Virtual Machine)[9]和Xen[10]为代表的服务器虚拟化和以Docker[11]和rocket[12]为代表的操作系统层虚拟化(又称容器化)[13]。虚拟机在存储、网络和操作系统等方面模拟物理服务器，比物理服务器更可移植、更快，但仍然是完整的服务器系统。容器将密度和优化提升到一个新的水平，通过共享操作系统内核的形式[34]，达到了更轻量级、启动更快、管理更流畅的目标，在存储优化、CPU使用率、运维效率等方面带来的收益均比虚拟机带来的收益大得多[13]。

遥感图像应用处理可以分为遥感数据的接收与分发、数据处理与信息提取两个重要阶段[4]。如图 1.1所示，遥感数据的数据处理与信息提取过程可以初步分为预处理、信息增强、信息提取和融合等多个步骤，每个步骤又可以进一步细分，比如预处理过程可以细分为图像裁剪、几何校正和物理校正。遥感图像数据处理与信息提取过程就是针对不同场景，通过描述依赖关系将处理步骤灵活编排，在处理效果更好、处理效率更快的需求基础上，以个性化编排的方式，提取出个性化信息。基于容器编排的思想，提出把遥感图像处理过程分成彼此独立的功能区间，封装为容器，将对处理过程的编排解析为对容器的编排，将对过程的调用解析为对容器的调用。利用云的强算力和边缘的低延性的特性,将单个遥感任务拆分,调度到多个边缘节点的容器上,实现任务内和任务间的并行处理,提高任务处理效率。目前受欢迎的容器调度平台以Swarm[14]、Kubernetes[15]、Mesos[16]为代表，为开发者提供了新颖、便捷的软件集成开发、部署、运维。然而，传统的容器调度依赖于脚本的编写，对计算机专业的知识储备要求较高。Docker原生的Swarm依赖于Docker指令，Kubernetes是通过编写YAML文件完成容器编排，Mesos依托于C++、Java和Python语言编写的框架文件，需要查阅相应的API文档。遥感算法专业人员不善于使用这类平台。同时因遥感图像处理的数据庞大以及图像处理任务复杂多变的特点，遥感算法专业人员在编码过程中产生了大量的时间和人工成本。因此，开发一套简单易用的遥感图像可视化编排系统，具有较大的工程意义。

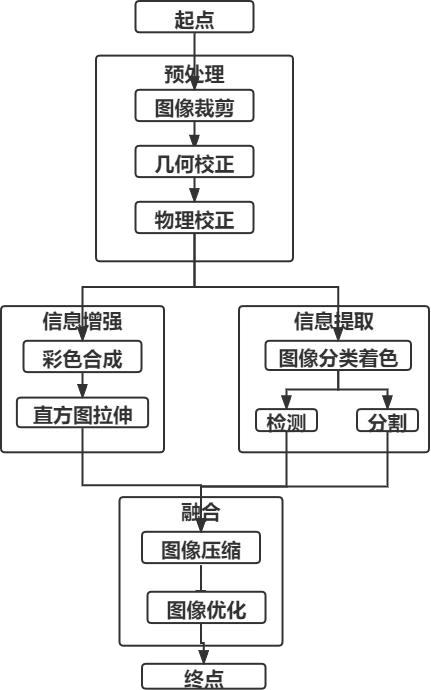


图 1.1 遥感图像数据处理与信息提取过程

### 课题意义

本文旨在利用可视化技术和云计算技术,基于可视化云编排理念，设计实现遥感图像处理任务可视化编排系统，通过模块化、拖拽式方式编排，根据应用场景快速构建处理流程，简化遥感图像任务编排流程，并通过构建Docker容器管理平台，实现遥感图像处理算法的自动云上部署和调度，加快处理流程落地，提高运维效率，解决因海量遥感图像处理步骤复杂、数据内容复杂而难以进行高效率 批量化处理的问题，为遥感图像专业人员提供有效处理遥感图像的手段，具有较大的实践意义。

## 国内外研究现状

### 传统任务编排系统

任务编排是指以任务为单位按照自定义的方式进行编排，描述任务执行过程。可将任务编排工作流建模为有向无环图[17](Database Availability Group,DAG)，该模型以任务为编排单位，利用依赖关系说明运行逻辑。具有相互依赖关系的任务之间具有执行先后关系,不具有依赖关系的任务之间也不具有相互约束关系,可以同时调度在不同的节点上同时运行,提高任务执行效率。如图 1.2所示,任务A和任务C之间不存在约束关系,可以同时运行。任务A和任务B之间存在依赖关系,需要先执行任务A再执行任务B。DAG本身并不关心任务内部发生的事，它只关心如何执行他们，包括运行的顺序重试的次数和执行是否超时。

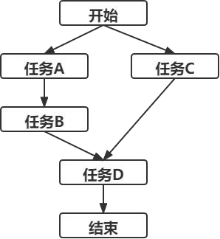


图 1.2 DAG工作流

近年来用于编排任务和数据工作流的各种通用工具快速出现。Apache-Airflow[18]是一个用于编排复杂计算工作流和数据处理流水线的平台，支持Mesos等资源调度,实现了完备的工作流可视化监控功能，其将工作流定义为代码，带来可维护性高、可测试性好和协作性好优点的同时，对用户在计算机专业知识储备方面的要求极高。Luigi[19]因其简单但可伸缩和可扩展的特点，可以容易地构建复杂的任务依赖关系图，适用于长期运行的流式批处理任务，但其缺点在于需要编写更多的自定义代码以实现任务。Argo[20]提供了基本的工作流功能，具有轻量级和功能全面的优点,基于Kubernetes编排并行作业,其通过DAG进行任务建模，描述工作流内部的依赖关系，支持高效地编排和运行计算密集型作业，但任务的编排依赖于编写类似于YAML的自定义领域专用语言(Domain Specified Language,DSL)代码，功能实现全面,但应用比较底层，不适用于用户。Apache-Airflow的示例脚本如图 1.3所示，Luigi的示例脚本如图 1.4所示，Argo的示例脚本如图 1.5所示。这些任务维护依赖于脚本，对计算机专业知识的要求较高，不利于遥感图像专业人士的使用。同时因海量遥感图像处理的数据庞大以及图像处理任务复杂多变的特点，用户在编码过程中产生了大量的时间和人工成本。综上所述，需要一种基于可视化技术的任务编排系统，来填补这些需求：无计算机专业背景的人员借助极少外力帮助，能够快速、高效的编排遥感图像任务。

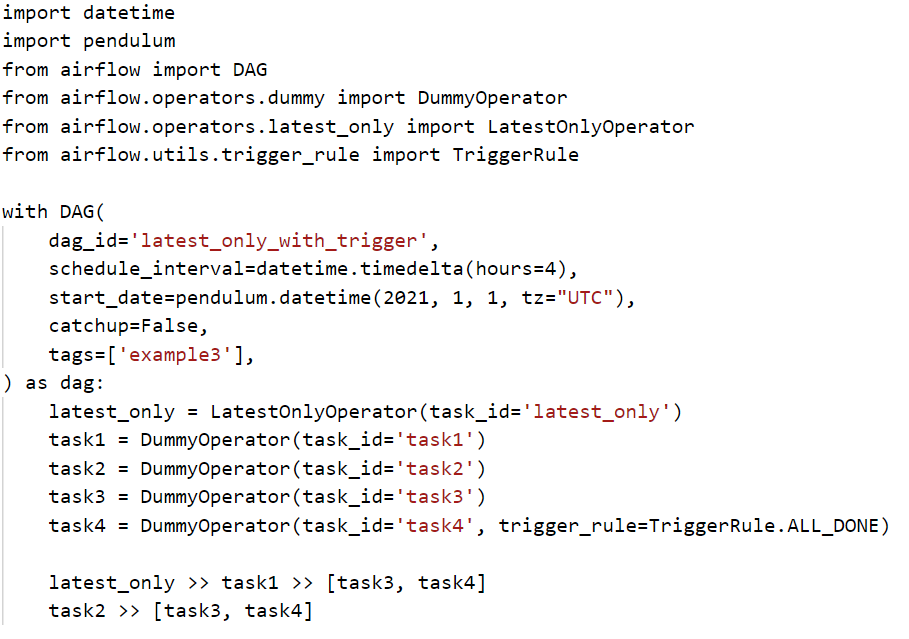


图 1.3 Apache-Airflow示例代码和工作流的可视化

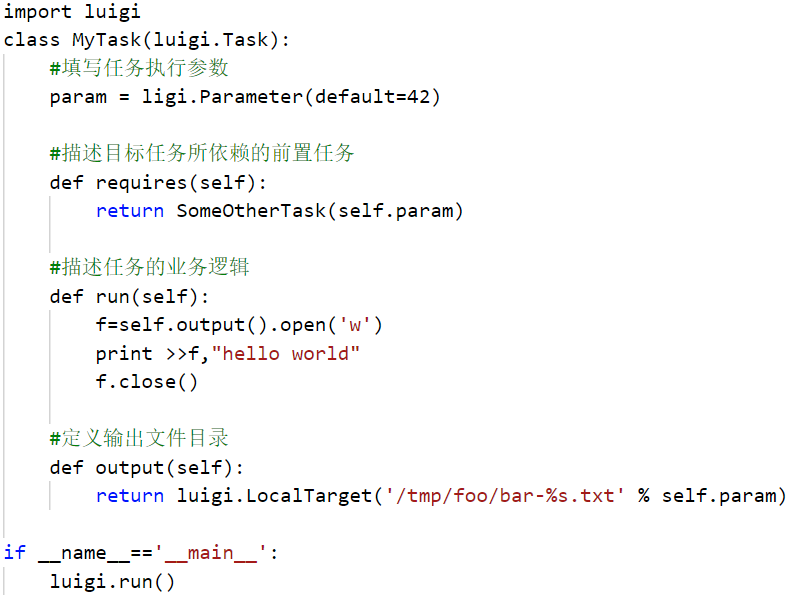


图 1.4 Luigi示例代码

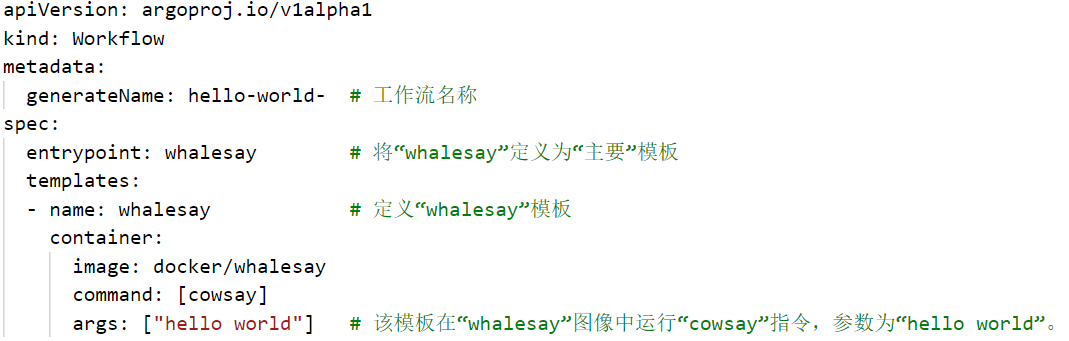


图 1.5 Argo一个简单规范的示例

### 通用任务编排系统的研究

任务编排的过程可以划分为三个步骤，包括把应用切分为独立的组件并做为镜像、利用配置文件或者拖拽控件的形式进行组件编排、利用虚拟机管理工具进行部署和运维[21]。国内外学者针对任务编排过程提出了不同程度的优化。针对部署速度慢以及适用场景单一的问题，文献[21]提出了基于Docker的混合云应用编排方案，提出了批量调度算法等性能优化算法，实现了良好的部署速度、分布均衡性和可靠性，但在监控完善性和模板管理上仍然存在不足。文献[22]实现了大数据编排与展示的可视化系统,存在多平台适配不好、可视化模型渲染速率较低、可视化编排不够灵活的问题。文献[23]实现了面向APP应用的可视化云编排系统,实现了业务和技术的有效解耦，促使应用的快速构建。文献[24]提出了基于Kubernetes的容器编排系统，支持集中式和分散式调度算法，实现了任务调度方式的最优化。关于多约束的组合与优化问题，文献[25]提出了约束规划模型，提出了综合考虑资源受限、任务约束、时间约束的任务调度方法。文献[26]实现了一种基于微服务的虚拟机自动编排技术，可以实现自动执行和自动部署，但编排模板受限，灵活度不高，同时任务调度策略并不是最优解。

以上通用任务编排系统的研究，提出的性能优化算法等为本文在实现编排、调度方面提供了技术基础。另外，虽然在任务编排系统上已经有了上述相关研究，但以上研究实现大多是基于Python、YAML等语言进行任务编排，在可视化编排方向研究欠缺。因此需要一个可视化的任务编排系统来填补这种需求。

### 特定任务编排系统的研究

目前大多数研究者倾向于研究面向任务编排系统的研究，而对特定领域的任务编排系统的研究较少。文献[27]采用观测目标数最大化原则进行编排，提出了相控阵雷达的观测任务编排算法，发挥雷达的最大性能。文献[28]实现了能够分布式检测的自动数采系统，在分中心上编排采集和上报任务，并维护一张任务表，进行任务的管理和调度。文献[29]考虑时空任务特征，构建了面向城市建设的可视化云-边-终端调度的灵活分配策略。文献[30]在护理领域提出建模以患者为核心的计算资源，围绕患者的特殊健康需求，接收以DSL语言描述的工作流并生成服务。在遥感图像领域，存在图像切割等特定类型的任务，对任务编排过程造成影响；根据处理需求，遥感任务处理涉及到大气校正、像素坐标等定量参数，需要用户的个性化设置。因此需要一个面向遥感图像的任务编排系统来填补这种需求。

## 研究目标与内容

### 研究目标

本文的主要目标是采用可视化技术，面向遥感图像处理搭建任务编排系统，实现可视化编排和工作流的监控，并基于Docker容器管理平台自动管理资源，达到提高遥感图像任务云上部署和调度效率的目标。

### 研究内容

本文从遥感图像处理任务编排和调度系统的实现方面进行研究，主要内容如下：

1. 构建便捷的可视化操作界面，实现任务的可视化DAG编排和工作流的监控。
2. 系统对任务DAG建模进行解析，形成JSON数据，易于机器的解析和生成。
3. 为了方便信息管理，设计MySQL数据库存储任务表和任务调度表，以状态为标志，对正常服务进行执行和故障服务进行重执行。
4. 边缘节点和核心节点交互采用C/S架构，本文实现在Linux系统上通过Socket方式与边缘节点进行通信。

## 论文组织结构

本文研究了遥感图像处理任务可视化编排系统的设计与实现，共分为6个章节，具体安排如下：

第一章，绪论。本章介绍了遥感图像处理任务可视化编排系统的课题背景和意义、国内外研究现状、研究目标和研究内容。

第二章，需求分析。基于需求调研，总结系统总体需求，确定论文实现目标，详细分析系统的功能性需求和非功能性需求。

第三章，总体设计。首先详细分析系统的总体设计，对系统的技术架构和部署架构进行详细分析，然后划分系统功能模块，阐明功能模块待实现的目标和大致方案。并利用类图和数据库表描述数据库设计。

第四章，详细设计与实现。首先描述系统架构的工作流程，接着介绍各个功能模块的具体实现方法，并对关键技术进行了详细分析。

第五章，测试。部署项目，进行功能性测试和非功能性测试，确保项目正常运行。

总结和展望。回顾遥感图像处理任务可视化编排系统的设计过程并总结工作，分析系统现存问题，提出优化方法。

# 遥感图像处理任务可视化编排系统需求分析

## 总体需求分析

本文设计实现的遥感图像处理任务可视化编排系统旨在实现三个需求：

1. 利用可视化技术实现可视化编排，提高提高遥感图像任务编排效率。
2. 依托Docker容器管理平台实现容器的自动维护和自动调度，简化遥感图像任务云上部署和操作的流程。
3. 综合考虑图像分割等特殊处理任务，构建适用于遥感图像的可视化编排系统。

遥感图像处理任务可视化编排系统的业务流程图如图 2.1所示。

C:\Documents and Settings\Administrator\Desktop\Image2.EMF

图 2.1 任务流程图

本系统有用户和管理员两种角色。用户在可视化界面管理数据信息。管理员负责云平台的维护和管理。

用户在可视化界面进行算子信息、任务信息、应用信息的管理。可以进行增删改查处理。针对任务信息，利用可视化拖拽进行任务信息编排；针对应用信息，可以通过图表的形式，实时查看应用执行情况，并获取应用执行的中间结果和最终结果。

系统对用户的操作进行响应，并将数据存入数据库。针对可视化编排数据，系统需要对数据进行校验，若为不合理的编排则返回系统提示，若为合理的编排数据则进行解析，完成DAG建模，生成拓扑排序。系统收到新建任务的请求时，会根据算子信息和编排信息进行解析，生成调度数据，以便服务端调度使用。

管理员负责服务端和客户端的部署管理。管理员自行完成客户端的部署，并在平台上管理客户端信息。新的客户端会和服务端建立消息连接。

服务端负责云平台上算子的维护和调度。服务端通过监听数据库中数据的变化，生成算子构建和任务调度服务，和多个客户端进行信息和文件的交互，将任务分散到多个客户端上并发调度。并针对故障服务完成重调度。

## 功能性需求

本系统的参与者包括用户和管理员。该项目的功能性需求可以划分为用户管理模块、面向管理员的客户端管理模块和面向用户的可视化遥感图像处理模块。

### 用户管理模块

用户管理模块的用例如图 2.2所示，包含退出登录、修改密码等基本操作。管理员具有创建和删除用户的特殊权限，保护了系统的私密性。

C:\Documents and Settings\Administrator\Desktop\Image2.EMF

图 2.2 用户管理模块用例图

### 客户端管理模块

客户端管理模块的用例如图 2.3所示。管理员可以管理客户端信息。管理员必须预先在目标IP上构建客户端，和服务端成功建立连接，随后管理员在平台上填写客户端信息，新建客户端，平台检验连接有效性，如为无效连接，则会新建客户端失败。管理员可以在平台上管理客户端的镜像存储地址、图像存储地址、管理员账号密码等信息，用于后续算子的构建和调度。

C:\Documents and Settings\Administrator\Desktop\Image2.EMF

图 2.3 客户端管理用例图

### 可视化遥感图像处理模块

可视化遥感图像处理模块的用例如图 2.4所示。该模块的参与者包括时间和用户，主要可以分为算子管理、应用管理、任务管理三个部分。

C:\Documents and Settings\Administrator\Desktop\Image2.EMF

图 2.4 可视化遥感图像处理模块

算子管理的用例包括查看算子列表、新建算子、删除算子、编辑算子、解析算子和部署算子。用户通过上传算子镜像，添加算子构建信息和算子调度信息，在平台上新建算子。平台每隔一段时间会查看一次数据库，从数据库中获取用户新上传的算子，生成算子构建服务，并执行服务，完成算子的自动构建。平台需要维护算子构建的一致性，针对构建失败和客户端新建的情况，采取相应的操作，保证所有算子在所有客户端上完成构建。

应用管理的用例包括查看应用列表、新建应用、删除应用和编辑应用。用户可以在应用编辑页面通过拖拽、连线和信息填写完成应用的可视化编排，生成有效的DAG建模，同时实现编排结果的重显示和重编排，优化用户的使用满意度。用户在保存DAG建模时，平台需要根据一定的逻辑校验，并将错误建模提示返回给用户；针对图像分割等特殊应用，应用生成独特的校验逻辑。平台对DAG建模数据完成校验和解析，生成拓扑排序，便于后续使用。

任务管理的用例包括查看任务列表、新建任务、删除任务、查看任务、解析任务和调度子任务。用户可以针对应用创建任务，平台获取应用的解析数据，生成算子调度服务，完成遥感图像任务在云平台上的自动调度，并针对调度失败的情况实现重调度，避免因网络异常等因素造成任务执行崩溃。在任务查看界面，用户可以查看任务在边缘节点上的分配情况和当前调度情况，并在任务调度结束后获取调度结果。

## 非功能性需求

### 性能需求

对于用户和管理员而言，数据库响应时间、页面加载时间和系统响应时间不应太长。页面加载和接口响应需要尽量快速，平台需要尽快地对用户操作做出响应，告知用户发生了什么。性能需求也对后端处理过程中的数据精准性提出了要求，包括不允许错删、多删和修改准确性等。针对算子的部署和任务的调度涉及到文件的传送和脚本执行，需要较长的执行时间。该系统的服务端采用多线程调度，用户无需等待实际的执行，先将数据提交，系统先返回后处理。对于服务的执行涉及到文件的传递，需要尽快无误地传送文件。任务的调度会优先分配在性能较好的客户端上，并实现任务内和任务间的并行调度，让多个客户端同时运转，处理信息，降低任务调度所需要的时间，提高性能。

### 易用性需求

易用性是指系统被用户理解和使用的能力。界面风格需要简洁明了、重点突出，用户操作需要简单易用、直观可见。通过操作方法明确、互动性按钮位置明显等方法帮助用户快速定位和操作。采用图表提示、拖拽等方法，让用户快捷获取所需要的信息，用鼠标完成一些比较复杂的操作。

### 可扩展性需求

可拓展性指系统在需求变更时提供的一种扩展能力，系统仅需要对原有代码进行少量的编写即可支持新模块的添加或者旧模块的修改。通过注释代码、封装功能模块等形式支持系统的可扩展性。另外提出对云平台的网络可扩展性提出需求，需要简化云平台的运维操作，管理人员通过极少步骤即可对云平台中边缘节点数量、信息进行管理。

### 安全性需求

系统的安全性体现通过身份鉴权、信息加密等方式，保护系统中信息的完整性和机密性。提高系统的防御能力，防止因有意或者无意的攻击造成用户资料的泄露和系统功能的滥用。另外在网络通信时，需要对用户密码等私密数据加密，保护用户隐私，提升数据安全性。

### 鲁棒性需求

系统的鲁棒性是指系统在异常情况，如输入错误、有意攻击[31]，能维持不崩溃、正常运行的能力，比如管理员新建用户时填写的账号名重复，用户编辑任务时DAG建模不符合规范(比如存在环或者算子入度与实际输入数量不匹配)，服务端进行调度任务时发生网络异常等情况导致调度失败等等，系统能发现错误的操作并能有合理的处理方式和用户反馈，恢复到原状态或稳定状态。

# 遥感图像处理任务可视化编排系统总体设计

## 技术架构设计

本文采用三层架构设计，基于高内聚低耦合的基本原则，将系统划分为三层，各层之间采用接口相互访问，通过构建与数据库表相对应的对象模型实体类，进行数据传递。通过设计结构简单、开发明确的系统架构，将开发工作重心集中在系统的核心业务逻辑上，提高开发效率，优化面向用户的功能性需求，提高系统质量。在编码过程中，基于提前制定好的接口标准，即可分别进行开发，大大提高系统的开发速度。

### 技术架构设计方案选择

软件系统宏观上层架构设计会对复杂软件设计的成功与否造成关键因素，主流系统架构设计有事件驱动架构、微服务架构、分层架构等，需结合系统特点，需采用适用于本系统的技术架构设计。

#### 方案一：事件驱动架构

事件是状态发生变化时，软件发出的通知。事件驱动架构是通过事件进行通信的软件架构，事件触发的消息在独立、非耦合的组件和服务之间传递。事件驱动架构提供瞬时过滤、聚合和关联事件的能力，实现快速有效地响应和处理事件[32]。事件驱动架构易于开发和维护大规模分布式应用程序和不可预知的服务或异步服务，有利于应用程序在高并发期间对来自各种源的变化做出响应，而不会过度消耗资源。

本项目对高并发的需求并不高，且事件驱动架构系统构建相对复杂，需要考虑远程通信、失去响应等情况对系统运行造成的影响，且难以支持原子性操作[33]。

#### 方案二：微服务架构

微服务架构是以一套微小的服务来开发和服务一个单独的应用，这些微小的服务根据业务功能划分，独立部署在各自的进程中，并通过轻量级通信机制进行通信。每个微服务可以独立更新，升级迭代更方便，容错率更高。同时在服务变更和升级后，仅需要重新部署该模块，而不是整个系统，更有利于可持续集成，可拓展性也更高[34]。然而，这种系统拆分的方法，需要从技术层面划分UI端、服务端和数据库端，使得一个简单的变动需要整个团队来适应和协调。同时，这对如何划分清晰的服务提出了挑战，要求团队的领导者具有足够的经验，也要求每个小团队能具备开发过程中所具备的所有功能。

本项目需求稳定，在系统的集成性和迭代性方面需求较小。并且本项目开发团队小，开发周期短，难以解决在微服务架构为满足负载均衡等非功能性需求涉及的复杂性问题，比如数据一致性问题、性能监控问题等等，难以构建稳定可靠的微服务系统。因此本项目不考虑采用微服务架构。

#### 方案三：分层架构

分层架构是将系统的组件划分到不同的层中，通过调用下层接口进行通信的软件架构。每一层的组件具有内聚性和抽象性，下层的工作细节对上层是透明的，具有高内聚低耦合的特点。分层架构设计结构简单，便于理解和开发，并可通过模拟接口的方式，实现每一层的独立测试。分层架构设计清晰，在编码过程中仅需提前定义好接口设计，即可专注于本层所用功能和技术，大大降低了层与层之间的依赖。当某一层具有新的实现时，可以在不修改其他层的情况下，实现顺利交互[35]。分层架构的复杂性较低，应用的额外开销更小，整体性能更好。

分层架构结构简单，具有可读性强、可测试强的优点，便于让团队尽快构建可靠稳定的系统，将更多精力花费在系统核心业务逻辑的设计和开发上，有利于项目的更新和维护。

综上几点考虑，为了构建更加稳定的系统，和方便开发，本系统采纳分层架构设计。

### 遥感图像处理任务可视化编排系统技术架构设计

系统技术架构设计如图 3.2所示。通过3.1.1节对系统技术架构方案的论证分析，本文服务端采用三层架构设计，将系统分为表现层、业务层和数据层。将数据存储、逻辑处理和显示代码分成单独的模块，描述层与层之间的通信方式，系统具有较好的可读性和可测试性。项目结构清晰，分工明确，系统具有较好的可维护性。数据层不包含任何代码，只有数据库，业务层和数据层之间通过对象模型的实体类对象来传递数据，有效避免表现层直接访问数据层，提高数据安全性。

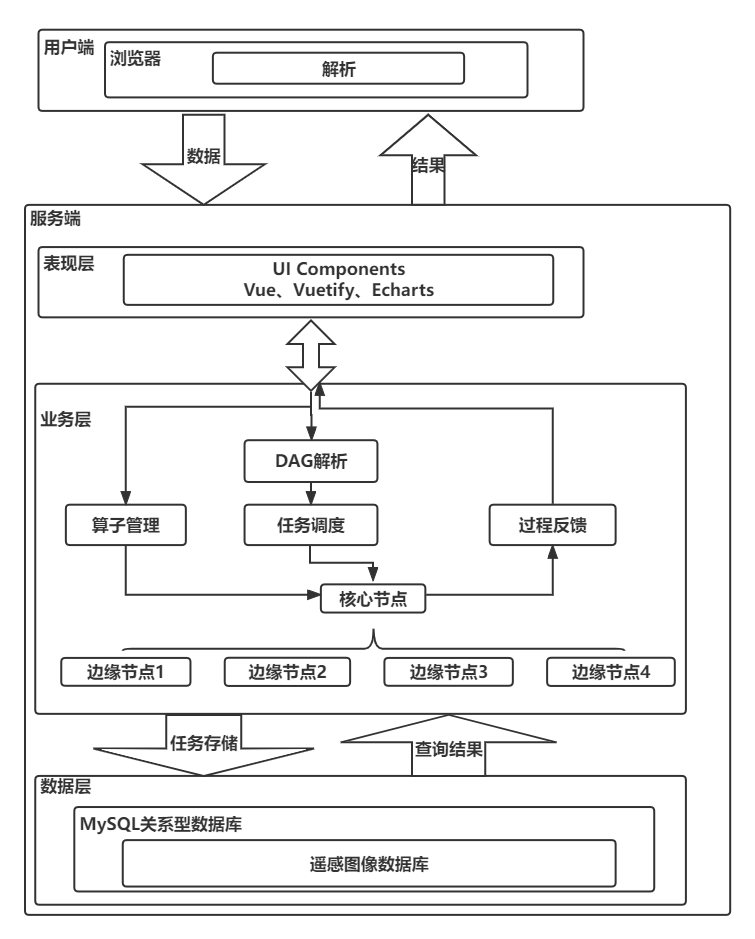


图 3.1 系统技术架构图

表现层是系统和外界的接口，提供系统和用户交互的方式。表现层编写合适的样式，从业务层获取数据，并进行简单的解析，以方便易读的方式表现给用户。页面上提供交互接口，可获取用户操作，将操作类型和操作相关数据转发给表现层，触发相应的逻辑运行，并获取运行结果，对页面信息进行更新，通知用户。网页编写采用Vue框架，基于组件化的思想，便于系统的开发和维护[47]。网页设计采用Vuetify样式库，保证整个网站风格和样式的统一性，提高信息的可读性。在数据可视化方面，分别采用SVG画布实现可视化DAG建模和工作流的监控以及Echarts图表完成任务信息统计。

业务层是数据层和表现层之间的纽带，将实际信息处理和信息显示分离，提供了良好的扩展性和可读性。其获取用户操作信息，翻译成相应的逻辑运行，将运行结果返回。数据的处理主要由基于Django框架的后端以及基于Python语言编写的核心节点和边缘节点完成。后端获取表现层请求，完成解析DAG数据和其他基本信息处理操作，将数据存入数据库。核心节点监控数据库中数据的变化，获取相应信息生成服务，将服务调度到边缘节点上执行，并将调度结果返还给表现层。因为核心节点启动两个线程分别监听数据库中算子信息的变化和任务信息的变化，并进行读写操作，因此利用锁来保护读写一致性。

数据层采用MySQL关系型数据库，支持多用户和多线程，满足业务层中后端和核心节点的并发读写需求，并通过维护事务一致性，实现了并发读写过程中数据的一致性。MySQL关系型数据库还具有维护性高、可靠性强、冗余性低的优点。其将系统管理基本信息和服务调度的过程信息存储在不同的表中，利用外键描述数据之间的关系，提高了可读性和灵活性。其接收来自业务层的数据查询请求，返回响应数据，并在算子信息和任务信息发生变化时，通知对该数据感兴趣的业务层模块。

## 系统功能模块设计

系统功能模块图如图 3.2所示，可为用户管理、客户端管理、可视化遥感图像处理三个功能模块，其中可视化遥感图像处理模块可细分为算子管理模块、应用管理模块和任务管理模块。

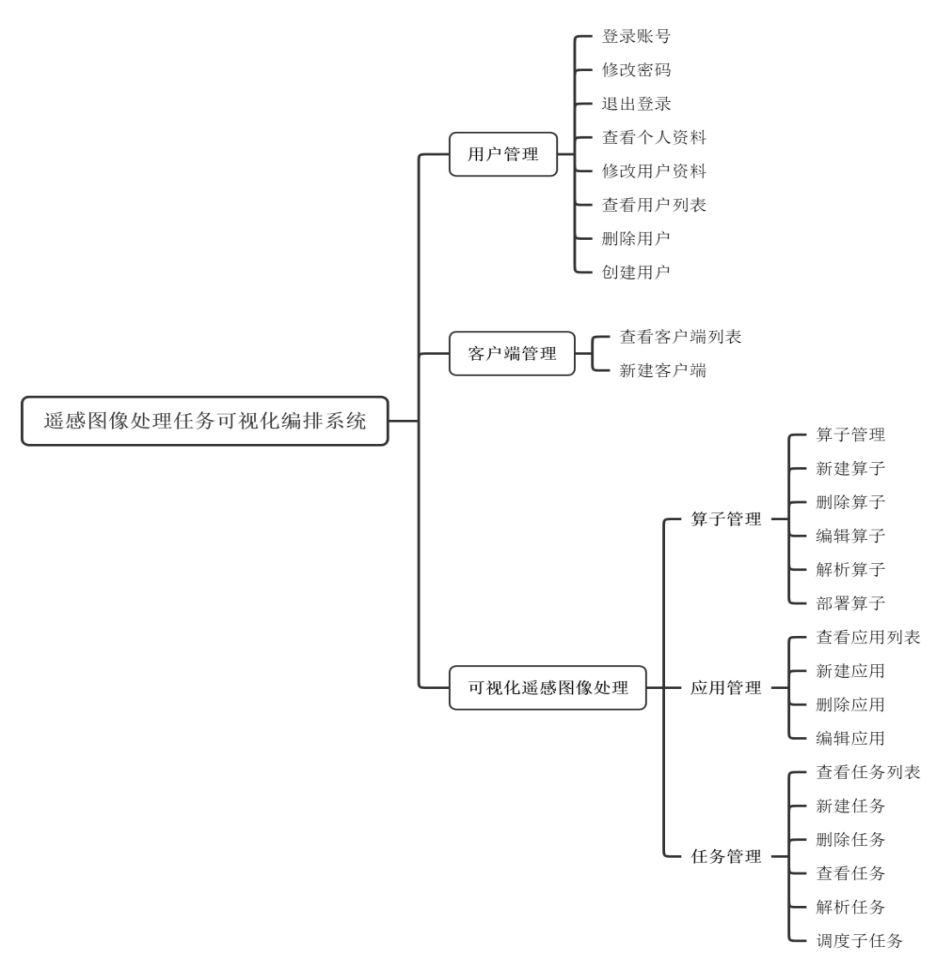


图 3.2 功能模块图

### 用户管理模块的功能

提供基本的登录注册功能。管理员具有创建用户和删除用户的特殊权限，在数据库中修改记录，并将创建的账号信息通过邮件告知对方。

### 客户端管理模块的功能

管理员在客户端管理模块上进行Docker云平台的维护和信息管理。管理员在平台上可以新建和删除客户端连接，管理客户端列表，并维护客户端调度所需的信息。管理员在平台上新建客户端前，需要先在目标IP上部署客户端，和服务端形成可靠的连接。随后，管理员在平台上添加客户端，输入目标IP、端口，并输入客户端控制信息。系统所需要的客户端控制信息包括管理员账号密码、本地镜像存放目录、本地图片存放目录等，分别运用于算子构建和算子调度的过程中。最后，系统利用IP和端口定位客户端连接，判断目标客户端的连接有效性，在连接无效的情况下会提示创建客户端失败。后续服务器在镜像构建和子任务调度时会获取数据库中的客户端控制信息，以支持服务的执行。

### 可视化遥感图像处理模块的功能

可视化遥感图像处理模块可以分为算子管理、应用管理和任务管理。

在算子管理模块，用户可以上传算子，由平台实现算子的自动构建，并负责维护Docker容器管理平台的部署一致性。考虑到遥感图像任务的处理涉及到更多自定义参数，用户在上传算子的时候，支持对算子参数、运行代码、输入路径、输出路径的自定义。考虑到算子分割等特殊任务，支持算子的运行有多个输入和输出，通过设置参数类型，分别为“输入”、“输出”和“无”来识别。

在应用管理模块，实现应用的可视化编排，简化编排流程。将算子视为单位，通过拖拽和连线进行可视化编排，补全算子参数信息，实现可视化DAG建模。平台需要完成对编排数据的检验和解析。同时考虑到分割图像和合并图像的特殊遥感处理任务，设计更加灵活的可视化编排。

在任务管理模块，实现遥感任务的自动调度和对工作流的监控。平台需要对新建任务数据进行解析，由核心节点生成服务，让任务调度到多个边缘节点上运行，实现任务内和任务间的并发，并针对调度失败的情况启动重调度。用户可以在任务详情界面进行工作流的监控，获取执行结果。

## 系统部署架构设计

C/S架构和B/S架构交叉使用的三层混合部署结构兼具事务处理能力强的特点和网络易扩性的特点，结合C/S模式和B/S模式的特点，具有更大的优势和应用前景。因此本系统采用C/S架构和B/S架构交叉使用的三层混合部署结构。

### 部署架构设计方案选择

#### 方案一：B/S架构设计

B/S(Browser/Server，浏览器/服务端)结构用通用的浏览器即可实现原来复杂的专用软件才能实现的强大功能。浏览器仅承担网页信息的浏览功能，没有任何业务处理的能力；服务端完成业务逻辑处理和信息存储管理，和数据库进行交互，实现数据的增删改查。B/S架构的提出，使得移动办公成为可能，只要拥有一台能上网的电脑即可使用，可移植性高，降低了系统维护和升级成本。然而，B/S架构将计算压力集中在服务端，当计算压力和访问量大时，服务端负载过重，难以高效响应。且HTTP可靠性较低，有可能造成网络故障，对于管理者来说，采用浏览器方式进行系统维护是不安全不可靠的。

#### 方案二：C/S架构设计

C/S(Client/Server，客户端/服务端)结构是基于资源共享思想提出的体系结构。该应用模式通常采用高性能工作站作为服务端，并配备有高性能数据库，进行数据管理，充分发挥客户端的处理能力，将任务合理分配到客户端上，降低服务端的数据负荷[37]。C/S架构具有强大的数据操作能力和事务处理能力，且开发模式简单，易于理解和接受。然而C/S组织结构只适用于局域网，无法应对用户对移动办公和分布式办公的需求。且C/S架构下客户端移植困难，维护升级成本较高。

#### 方案三：C/S架构和B/S架构混合设计

为了克服B/S结构和C/S结构的缺点，提出了C/S和B/S结构混合的体系架构，一些可以满足大部分访问需求的功能采用B/S结构，充分考虑用户利益，维护简单，易于操作，保持了“瘦客户端”的优点。C/S结构具有较强的事务处理能力，适用于分布式组织，将需要分布式扩展的功能合理分发到客户端上运行，充分利用客户端的性能，保持了“胖客户端”的优点。管理员也可以通过浏览器端的友好且灵活的界面发起请求，间接操控C/S架构下的服务端分发服务。上述结构将C/S和B/S模式混合应用，充分发挥两种模式的特点，具有良好的发展前景。

本系统需要具备用户需要的数据管理和任务编排的功能，又需要具备分布式容器的构建、管理和调度功能。为此，系统使用C/S架构和B/S架构交叉使用的三层混合部署结构，集齐C/S架构事务能力处理强的优点和B/S架构下网络易扩性和分布式的特点，实现用户模式下的共享性和易用性以及云平台的专用性和动态伸缩性[36]。

### 遥感图像处理任务可视化编排系统部署架构设计

系统部署架构图如图 3.3所示。

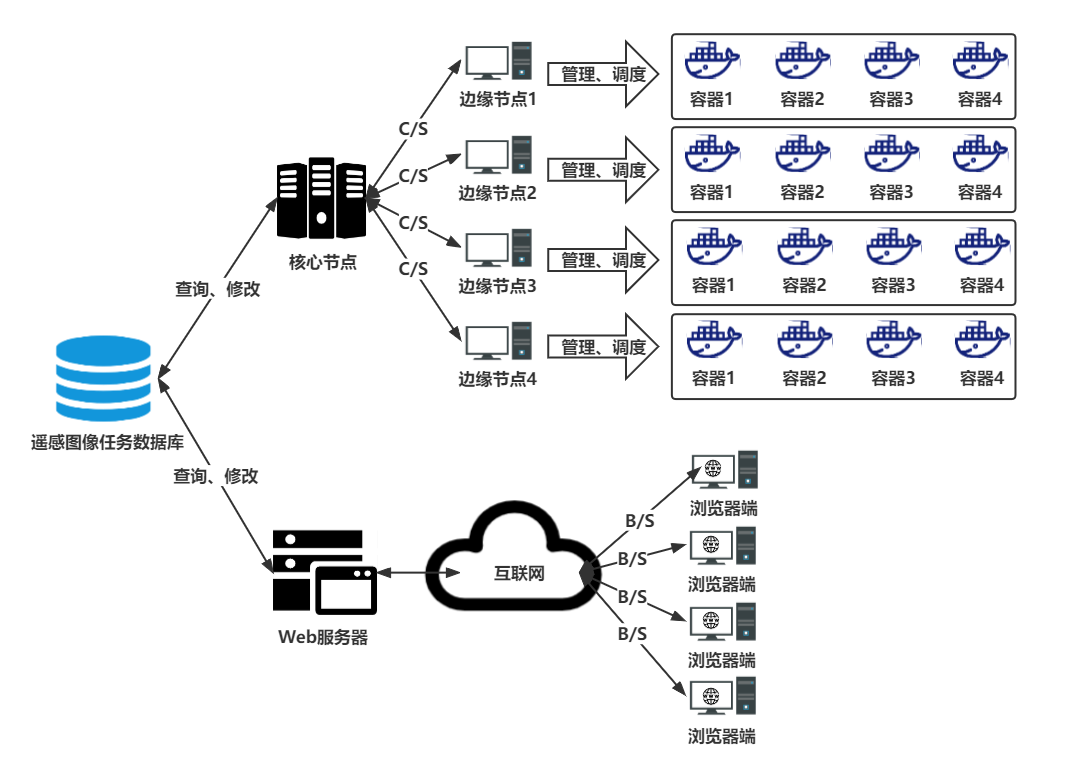


图 3.3 系统部署架构图

B/S结构面向外部用户，基于浏览器实现前端界面[38]，用户通过获取一个网址即可在本地查询。该架构向用户提供直观的前端操作界面和流畅的操作体验，实现用户需要的数据管理的功能。S端为Web服务器端，缓解浏览器端数据处理压力[38]，处理来自浏览器端的请求，解析数据，对遥感图像任务数据库进行操作；B端为浏览器端，提供直观方便的界面，为用户提供可视化编排、信息管理等功能并向服务端发送请求。

C/S结构面向管理员，整合服务资源，是整个任务编排系统的控制中心，维护Docker容器管理平台，负责管理云上容器的管理和调度。S端为核心节点，通过监听遥感图像任务数据库中算子和任务的变化，针对算子的部署和任务的创建处理底层逻辑，生成构建算子服务和子任务调度服务，将服务调度到目标边缘节点上执行。C端为边缘节点，负责和服务端通信，获取服务，对本节点上的容器进行管理，完成实际的算子构建和子任务调度的过程。C/S结构内部通信采用更加底层的TCP/IP协议，并根据文件和信息传输的实际情况自定义传输格式，避免了繁冗的HTML格式描述，增加数据传输效率[36]。

## 数据库设计

数据库类图如图 3.4所示。

C:\Documents and Settings\Administrator\Desktop\Image2.EMF

图 3.4 数据库类图

### 用户管理模块的数据库设计

user表记录用户和管理员的基本数据，包括用户名、密码、账号备注字段，如表 3.1所示。

表 3.1 user表

| 字段名 | 类型 | 描述 | 备注 |
| --- | --- | --- | --- |
| ID | int | 用户标志 | 主键 |
| username | varchar | 用户名 |  |
| password | varchar | 密码 |  |
| description | varchar | 账号描述 |  |
| is\_admin | boolean | 是否为管理员 | True表示为管理员，False表示为用户 |
| email | varchar | 邮箱 |  |

### 客户端管理模块的数据库设计

client表记录客户端的基本数据，包括客户端IP、端口、系统账号名、图像存储路径等字段，如表 3.2所示。

表 3.2 client表

| 字段名 | 类型 | 描述 | 备注 |
| --- | --- | --- | --- |
| ID | int | 客户端标志 | 主键 |
| IP | varchar | 客户端所在IP |  |
| port | int | 客户端所在端口 |  |
| operator\_dir | varchar | 镜像存储目录 |  |
| img\_dir | varchar | 图像存储目录 |  |
| password | varchar | 密码 |  |
| account | varchar | 账号名 |  |

### 可视化遥感图像处理模块的数据库设计

算子管理模块包含operator表、operator\_parameter表和distribute\_wait表。

operator表记录算子的基本数据，包括镜像存储目录、输入文件路径、输出文件路径、容器执行代码、镜像文件名称等字段，并定义operator表的外键，如表 3.3所示。

operator\_parameter表记录算子参数的基本数据，包括算子名称、算子类型、参数顺序和参数描述等字段，并定义operator\_parameter表的外键，如表 3.4所示。

distribute\_wait表记录算子构建的基本数据，包括目标算子、状态等字段，并定义distribute\_wait表的外键，如表 3.5所示。

表之间的关系如下：

1. 算子和用户是多对一的关系。一个算子只有一位创建者，但一个用户可能创建不同算子，其后同理，不再赘述。
2. 算子参数和算子是多对一的关系。一个算子参数只能属于一个算子，但一个算子可能拥有多个算子参数。
3. 算子构建信息和算子是多对一的关系。 一个算子构建信息只针对一个算子，但一个算子可能有多个算子构建信息。
4. 算子构建信息和客户端是多对一的关系。一个算子构建信息只针对一个客户端，但一个客户端可能有多个算子构建信息。

表 3.3 operator表

| 字段名 | 类型 | 描述 | 备注 |
| --- | --- | --- | --- |
| ID | int | 算子标志 | 主键 |
| user\_ID | int | 用户 | 外键，指向user(ID) |
| file | varchar | 镜像存储目录 |  |
| filename | varchar | 镜像tar包名称 |  |
| input\_dir | varchar | 输入文件路径 |  |
| output\_dir | varchar | 输出文件路径 |  |
| code | varchar | 容器的执行脚本 |  |
| description | varchar | 算子描述 |  |

表 3.4 operator\_parameter表

| 字段名 | 类型 | 描述 | 备注 |
| --- | --- | --- | --- |
| ID | int | 算子参数标志 | 主键 |
| operator\_ID | int | 算子 | 外键，指向operator(ID) |
| name | varchar | 参数名称 |  |
| parameter\_type | varchar | 参数类型 | 分为input、output、none |
| sequence | int | 参数顺序 |  |
| description | varchar | 参数描述 |  |

表 3.5 distribute\_wait表

| 字段名 | 类型 | 描述 | 备注 |
| --- | --- | --- | --- |
| ID | int | 构建标志 | 主键 |
| operator\_ID | int | 算子标志 | 外键，指向operator  (ID) |
| client\_ID | int | 外键 | 外键，指向client(ID) |
| status | varchar | 构建状态 |  |
| infos | varchar | 算子构建相关参数 |  |
| start\_time | datetime | 开始构建时间 |  |
| log | varchar | 构建日记 |  |

应用管理模块包含app表、DAG\_node表、node\_parameter表和DAG\_line表。

app表记录应用的基本数据，包括名称字段，并定义app表的外键，如表 3.6所示。

DAG\_node表记录DAG图节点的基本数据，包括可视化坐标x、可视化坐标y、独特标识符等字段，并定义DAG\_node表的外键，如表 3.7所示。

node\_parameter表记录DAG图节点参数的基本数据，包括参数值等字段，并定义node\_parameter表的外键，如表 3.8所示。

DAG\_line表记录DAG图线段的基本数据，包括输入节点、输出节点等字段，并定义DAG\_line表的外键，如表 3.9所示。

表之间的关系如下：

1. DAG图节点和应用是多对一的关系。一个DAG图节点只属于一个应用，但一个应用可能含有多个DAG图节点。
2. DAG图节点和算子是多对一的关系。一个DAG图节点只应用一个算子，但一个算子可能对应多个DAG图节点。
3. DAG图节点参数和DAG图节点是多对一的关系。一个DAG图节点参数只属于一个DAG图节点参数，一个DAG图节点可能含有多个DAG图节点。
4. DAG图节点参数和算子参数是多对一的关系。一个DAG图节点参数只对应一个算子参数，一个算子参数可能产生多个DAG图节点参数。
5. DAG图线段和DAG图节点是多对二的关系。一个DAG图线段只指向两个DAG图节点，一个DAG图节点可能存在于多个DAG图线段。

表 3.6 app表

| 字段名 | 类型 | 描述 | 备注 |
| --- | --- | --- | --- |
| ID | int | 应用标志 | 主键 |
| user\_ID | int | 用户 | 外键，指向user(ID) |
| name | varchar | 应用名称 |  |
| DAG\_info | varchar | DAG数据解析 |  |
| description | varchar | 应用描述 |  |

表 3.7 DAG\_node表

| 字段名 | 类型 | 描述 | 备注 |
| --- | --- | --- | --- |
| ID | int | 节点标志 | 主键 |
| operator\_ID | int | 算子 | 外键，指向operator(ID) |
| app\_ID | int | 应用 | 外键，指向app(ID) |
| pos\_x | int | 可视化坐标x |  |
| pos\_y | int | 可视化坐标y |  |
| identity | int | 特殊标识符 |  |

表 3.8 node\_parameter表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 描述 | 备注 |
| ID | int | 节点参数标志 | 主键 |
| node\_ID | int | 节点 | 外键，指向node(ID) |
| operator\_parameter\_ID | int | 算子参数 | 外键，指向operator\_parameter(ID) |
| value | varchar | 参数值 |  |
| export | boolean | 是否导出 | 当对应operator\_parameter的parameter\_type为output时有用。 |

表 3.9 DAG\_line表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 描述 | 备注 |
| ID | int | 线段标志 | 主键 |
| from\_node\_ID | int | 外键 | 外键，指向DAG\_node(ID) |
| to\_node\_ID | int | 外键 | 外键，指向DAG\_node(ID) |

任务管理模块包含task表、task\_node\_status表、task\_output表和task\_node\_wait表。

task表记录任务总体的基本数据，包括名称、状态等字段，并定义task表的外键，如表 3.10所示。

task\_node\_status表记录子任务运行情况的基本数据，包括状态等字段，并定义task\_node\_status表的外键,如表 3.11所示。

task\_output表记录任务导出文件的基本数据，包括名称、存储地址等字段，并定义task\_output表的外键，如表 3.12所示。

task\_node\_wait表记录子任务调度情况的基本数据，包括状态等字段，并定义task\_node\_wait表的外键,如表 3.13所示。

表之间的关系如下：

1. 任务和应用是多对一的关系。一个任务只来自一个应用，但一个应用可能产生多个任务。
2. 子任务运行信息和任务是多对一的关系。一个子任务运行信息只属于一个应用，但一个任务可能含有多个子任务运行信息。
3. 子任务运行信息和DAG图节点是多对一的关系。一个子任务运行信息只对应一个DAG图节点，一个DAG图节点可以产生多个子任务运行信息。
4. 子任务调度信息和客户端是多对一的关系。一个子任务调度信息只运行在一个客户端上，一个客户端上可以调度多个子任务调度信息
5. 子任务调度信息和子任务运行信息是一对一的关系。一个子任务调度信息只应用一个子任务运行信息，一个子任务运行信息只产生一个子任务调度信息
6. 任务和任务输出文件是一对多的关系。一个任务可能产生多个任务输出文件，但一个任务输出文件只来自一个任务。

表 3.10 task表

| 字段名 | 类型 | 描述 | 备注 |
| --- | --- | --- | --- |
| ID | int | 任务标志 | 主键 |
| app\_ID | int | 外键 | 外键，指向app(ID) |
| user\_ID | int | 外键 | 外键，指向user(ID) |
| name | varchar | 名称 |  |
| input | varchar | 目标图像存储地址 |  |
| status | varchar | 任务状态 | 分为Start、Run、Pause、End |
| description | varchar | 参数描述 |  |

表 3.11 task\_node\_status表

| 字段名 | 类型 | 描述 | 备注 |
| --- | --- | --- | --- |
| ID | int | 子任务运行标志 | 主键 |
| task\_ID | int | 外键 | 外键，指向task(ID) |
| node\_ID | int | 外键 | 外键，指向DAG\_node(ID) |
| status | varchar | 子任务运行状态 | 分为Start、Run、Pause、End |

表 3.12 task\_output表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 描述 | 备注 |
| ID | int | 导出文件标志 | 主键 |
| task\_ID | int | 外键 | 外键，指向task(ID) |
| name | varchar | 文件名称 |  |
| file | varchar | 文件存储地址 |  |

表 3.13 task\_node\_wait表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 描述 | 备注 |
| ID | int | 子任务调度标志 | 主键 |
| status | varchar | 子任务调度状态 |  |
| start\_time | datetime | 开始调度时间 |  |
| rear\_wait\_ids | varchar | 子任务后置节点调度标志的拼接 |  |
| pre\_wait\_ids | varchar | 子任务前置节点调度标志的拼接 |  |
| infos | varchar | 子任务调度相关参数 |  |
| client\_ID | int | 外键 | 外键，指向client(ID) |
| task\_node\_status\_ID | int | 外键 | 外键，指向task\_node\_status(ID) |

# 遥感图像处理任务可视化编排系统详细设计与实现

## 部署架构的设计与实现

### B/S模式架构搭建

B/S架构下的服务端采用Django框架进行后端编写，在项目中配置mysqlclient，创建Model来操作MySQL遥感图像数据库中的十三张表。服务端为浏览器端提供接口，维护系统的逻辑，将用户和管理员提交的数据存储到遥感图像数据库中。

B/S架构下的浏览器端采用Vue框架进行网页编写，用Vuetify框架美化样式。根据需求分析，开发实现了9个前端页面：登录页、客户端管理页、算子管理页、应用管理页、应用编辑页、任务管理页、任务详情页、自定义错误处理页404、自定义错误处理页500。

应用编辑页采用原生SVG画布实现遥感图像任务的可视化编排，其通过Vue框架的component模块将算子封装为组件，将算子信息和位置信息组成组件的属性，通过连线完成组件和组件的连接，形成前置、后置关系，描述组件间的依赖关系，并用SVG图像的方式将结果呈现在网页中。该编排方法提取算子资源的基本参数，将底层细节透明化，使用户仅仅专注于应用编排的总体逻辑设计，而非代码语法的了解[48]，实现业务和技术的解绑。可视化编排的流程图如图 4.1所示。任务管理页和任务详情页使用Apache ECharts实现遥感图像任务调度的可视化查看和任务数据统计。

C:\Documents and Settings\Administrator\Desktop\Image2.EMF

图 4.1 可视化编排流程图

### C/S模式架构搭建

C/S架构下的服务端和客户端基于Python进行脚本编写。服务器作为核心节点，对遥感图像数据库进行数据监控，监听数据库中operator表、client表和task表的数据变化生成算子构建服务和子任务调度服务，分别存储到distribute\_wait表和task\_node\_wait表中。服务端每间隔一段时间会从distribute\_wait表和task\_node\_wait表中查询服务，和客户端进行消息和文件的传递，进行信息通信。客户端作为边缘节点，接收来自核心节点的信息，判断信息类型转而执行相应的操作。客户端收到算子构建服务和子任务调度服务时会执行服务并向核心节点反馈执行情况。服务器会不断更新数据库中服务的执行情况。

#### 信息传递

服务器和客户端通过Socket通信，进行消息的传递。消息包括服务信息和心跳消息。服务器广播心跳消息，并通过客户端能否正常返回判断当前连接是否正常，若连接不正常则将该异常情况反馈，并不再调度任务到异常客户端。服务器会通过监听数据库中算子信息和任务信息、客户端信息的变化，生成服务。服务包括以下两种：

1. 镜像构建服务。服务器需要维护所有边缘节点上镜像的一致性。当有新的客户端接入系统时，服务器会从数据库中获取所有的算子信息，生成所有算子到新客户端上构建的服务，存入distribute\_wait表；当有新的算子上传到系统上时，服务器会从数据库中获取所有的客户端列表，生成新算子到所有客户端上构建的服务，存入distribute\_wait表。
2. 子任务调度服务。服务器监听数据库，获取新建的任务信息并解析生成子任务。服务器定时从数据库中获取可执行子任务，基于贪心思想，在“空闲”客户端列表中优先采用性能好的客户端，将每个子任务分配到指定客户端上，并尽量实现任务内和任务间的并行，更新task\_node\_wait表；服务器发现有子任务因客户端故障，无法继续运行时，服务器会分配一个新的客户端，重新生成一个子任务调度服务，更新task\_node\_wait表。

服务端从数据库的distribute\_wait表和task\_node\_wait表中获取服务，利用JSON数据类型，将消息类型存储在cmd关键字中，将消息其余信息存在其余关键字中，并将JSON数据进行编码，转为二进制数据。客户端接收二进制数据并解码，还原为JSON数据，客户端利用cmd关键字进行判断，转而进行不同的操作。消息类型一共有以下三种类型：

1. load\_container类型为构建算子的命令，通过加载镜像的Docker命令将本地镜像tar包进行加载，需要参数信息如表 4.1所示。

表 4.1 算子构建参数

|  |  |
| --- | --- |
| 关键字 | 类型 |
| file\_location | 镜像本地存储地址 |

1. exec\_task类型为调度子任务的命令，通过创建容器、运行容器的Docker命令将指定镜像运行，输入指定参数，获取需要的结果，需要参数信息如表 4.2所示。通过构建数据卷的方式，实现容器内部和主机之间文件的传递和交互。

表 4.2 子任务调度参数

|  |  |
| --- | --- |
| 关键字 | 含义 |
| image\_name | 镜像本地存储地址 |
| container\_name | 容器名称 |
| local\_input | 本地输入文件目录 |
| local\_output | 本地输出文件目录 |
| container\_input\_dir | 容器输入文件目录 |
| container\_output\_dir | 容器输出文件目录 |
| code | 容器运行代码 |
| parameters | 容器运行参数 |
| password | 管理员密码 |

1. heart\_beat类型为心跳消息。心跳消息的作用是检查服务端和客户端之间的连接是否有效。

#### 文件传递

因遥感图像处理的特殊性，服务端和客户端直接需要交互文件，以支持服务的执行。交互文件情况如下：

1. 服务端和客户端之间传递算子执行的输入和输出图片。
2. 服务器把镜像tar包发送给客户端。

为了保证文件传递的稳定性，服务器和客户端通过建立TCP连接，进行文件的传递。服务端和客户端均会创建一个文件接收服务端，监听本地端口，接收文件。发送文件流程图如图 4.2所示。当节点需要发送文件时，会创建一个线程作为文件接收客户端端，通过已知的IP和端口，和目标机上的文件接收服务端建立TCP连接，先发送初始化消息，在发送文件内容。初始化消息包含四个信息：文件类型、本地文件大小、本地文件路径、目标机文件存储路径。文件类型包括镜像和图片两种类型。

C:\Documents and Settings\Administrator\Desktop\Image2.EMF

图 4.2 发送文件流程图

#### 连接维护

服务端维护一个客户端连接数组，数组中的每一个对象包含IP、端口、性能参数和状态信息。客户端状态转移图如图 4.3所示。

C:\Documents and Settings\Administrator\Desktop\Image2.EMF

图 4.3 客户端状态转移图

一方面，服务端在连接状态主动更改时进行数组更新；另一方面，服务端会和所有连接着的客户端不断发出心跳消息，如果在一定时间内收不到回复，则认为该连接失效，被动断开连接，在客户端连接数组中将指定客户端标注为“异常”状态。针对“异常”客户端，服务端不再调度服务运行，但仍会定期发送心跳消息，若收到正常回复则更新状态为“空闲”，并恢复服务调度。B/S架构下的后端也会定期和服务端进行一次通信，交换客户端连接数组，检查数据库中存储的客户端对象是否仍然和服务端建立有效连接。服务端需要调度算子构建和子任务调度服务时，会检查客户端数组中的对象，获取“空闲”状态的客户端，并利用客户端的性能参数进行排序，优先调用性能好的客户端，并更新状态为“忙碌”。客户端执行服务完毕后，状态恢复为空闲。

## 功能模块的设计与实现

### 用户管理模块

用户管理模块需要对数据库中的用户进行增删改查操作。

在使用者登录时，会对使用者的账号权限进行判断，假若是管理员，成功登录后会跳转到面向管理员的客户端管理页面；假若是用户，成功登录后会跳转到面向用户的应用管理页面。另外，管理员具有特殊权限，可以新建、修改和删除用户信息。管理员可以点击“新建用户”的按钮，在弹窗中输入账号名、密码和账号备注，后端在校验账号不存在且密码符合规范成功后，返回创建成功的提示并将账号名、密码等用户信息发送到指定邮箱中，通知新用户。

### 客户端管理模块

客户端管理模块需要对B/S架构下的客户端信息进行管理，提供可视化界面对客户端的信息进行增删改查。

客户端管理页面效果图如图 4.4所示，管理员可以查看当前客户端列表以及连接状态，获取“异常”客户端信息进行后台维护。

管理员点击“新建客户端”的页面效果图如图 4.5所示，新建客户端顺序图如图 4.6所示。用户在客户端管理页面单击“新建客户端”按钮，前端出现新建客户端的弹框，用户输入客户端定位信息，即IP和端口，和客户端控制信息，包括镜像存放绝对路径、图像存放绝对路径、管理员账号、密码等信息。用户单击“新建”按钮，前端在检验参数非空后会发送请求，把新建客户端的信息传给后端，后端首先会和服务端进行通信，服务端在客户端连接数组中检查，查看该客户端是否已经和服务端建立连接，如果不在，后端则会返回客户端建立连接失败的错误码，要求用户检查连接，重新填写数据。经过连接验证后，后端才会将客户端信息存储在client表中。



图 4.4 客户端管理页面

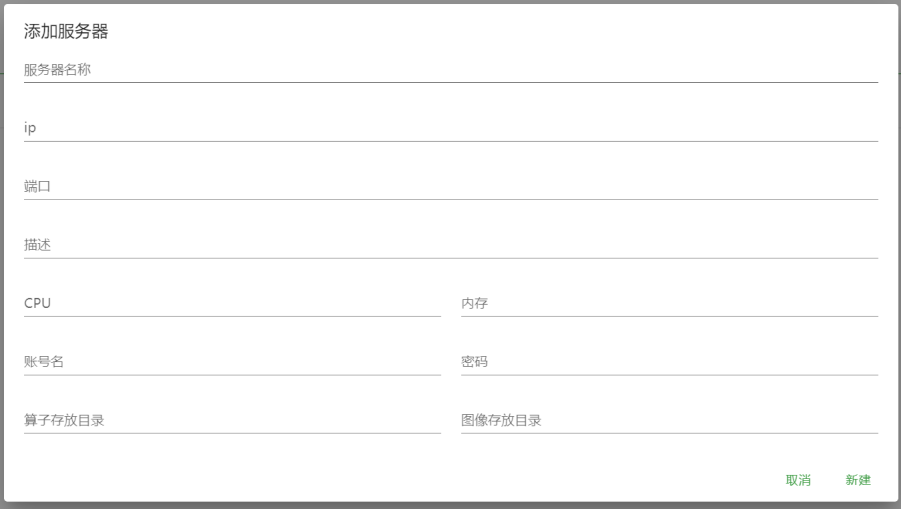


图 4.5 新建客户端效果图

C:\Documents and Settings\Administrator\Desktop\Image2.EMF

图 4.6 新建客户端顺序图

### 可视化遥感图像处理模块

#### 算子管理模块

算子管理模块中的功能和操作涉及到对数据库中的operator表、operator\_parameter表和distribute\_wait表进行操作，包括用户对平台上的算子进行增删改查的管理和服务端维护算子在客户端上的一致性部署两个功能。

用户通过上侧导航栏的“算子管理”按钮进入算子管理的页面，该页面会获取平台管理的所有算子信息，并进行分页显示。针对每个算子信息条目，可以点击“详情”按钮，平台会显示弹窗，显示算子的名称、运行代码、算子参数的详细信息；可以点击“删除”和“修改”按钮，删除和修改算子信息。用户也可以上传自己的算子，点击“新建”按钮，平台显示新建按钮的弹窗，新建算子弹窗效果图如图 4.7所示。用户在弹窗上输入信息，并上传打包好的镜像。“运行脚本”指镜像内初始文件的运行命令，“输入路径”参数为镜像输入数据卷的目录绝对路径，“参数”指镜像内启动文件的运行参数，会按照输入的顺序将所有参数值进行拼接，用空格分隔，作为运行脚本的参数，“输出路径”参数为镜像输出数据卷的目录绝对路径，实现容器内和主机间文件的交互。其中参数类型分为“输入文件”、“输出文件”和“空”。在算子调度时，根据参数类型进行判断参数是否为文件，如果是文件，则需要相应的操作完成客户端和服务端文件的交互。用户点击“新建”按钮，前端会先进行信息校验，如果参数为空，则会提示用户将算子信息补充完整。校验成功后，前端会向后端发起新建算子的请求，后端将算子信息存储到operator表和operator\_parameter表中。



图 4.7 新建算子效果图

B/S下的服务器保证每个客户端都部署有系统上所有的镜像，通过监听遥感图像数据库中算子数据、客户端数据的变化，生成算子构建服务存储到distribute\_wait表中，保证部署的一致性。算子构建服务执行流程图如图 4.8所示。服务器每隔5秒查询数据库的distribute\_wait表，获得待部署的算子信息，会和目标客户端进行消息的交互，依次完成tar包的分发、镜像的加载，实现算子自动构建。并针对失败的执行过程会进行重执行，以防止系统的短暂异常导致的构建失败。

C:\Documents and Settings\Administrator\Desktop\Image2.EMF

图 4.8 执行算子构建服务流程图

可以通过命令行工具查看构建过程和构建成功的结果。新建算子的服务端构建输出如图 4.9所示，添加客户端的服务端构建输出如图 4.10所示。

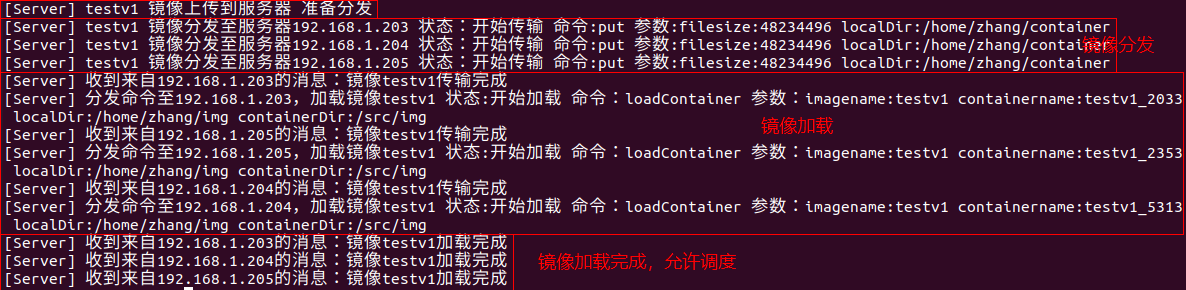


图 4.9 新建testv1镜像，所有客户端均收到消息进行构建



图 4.10 添加客户端192.168.1.205，客户端对平台上的所有镜像进行构建

算子构建服务的状态转移过程如图 4.11所示。

C:\Documents and Settings\Administrator\Desktop\Image2.EMF

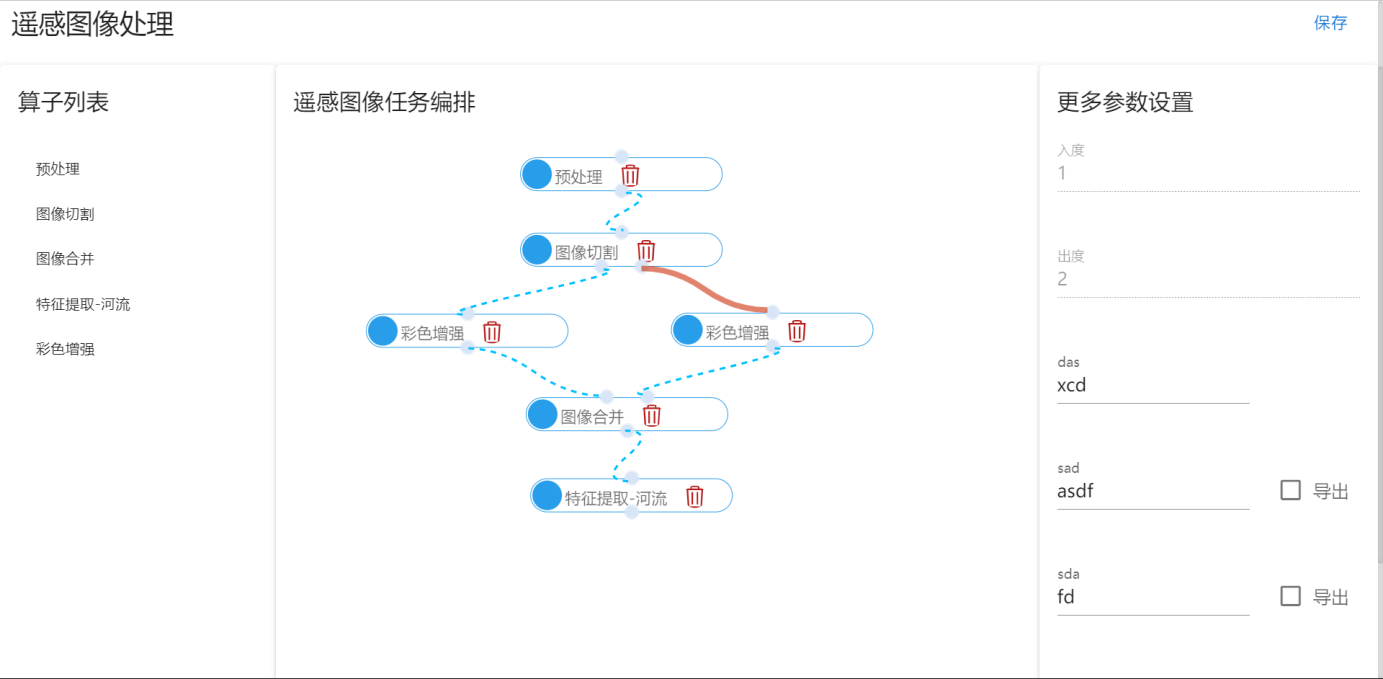
图 4.11 算子构建服务状态图

算子构建过程一共有5个正常状态和3个异常状态。正常状态包括start、file\_loading、file\_loaded、msg\_running和end,描述算子正常构建过程。异常状态包括file\_failed、msg\_failed和failed,描述算子构建过程中因网络或者硬件等因素造成的文件传递失败和构建失败，服务器针对异常失败的情况会采取重构建的手段，防止因系统暂时异常造成系统崩溃。如果失败次数过多，服务器会拒绝重构建，并将构建错误的日志反馈给管理员，管理员可根据报错信息检查客户端的网络环境和硬件环境，在检查确认无误后会重启构建任务。重构建也保证了客户端平台上算子的一致性部署。

#### 应用管理模块

应用管理模块中的功能和操作涉及到对数据库中的app表、DAG\_node表、node\_parameter表和DAG\_line表进行操作，基本操作有新建应用、修改应用、查看应用列表和删除应用，包括可视化编排和DAG建模数据解析两个核心功能。

可视化编排页面效果如图 4.12所示。页面首先会调用后端，获取用户上传的所有算子列表，显示在页面的左侧。用户点击算子列表中的一个算子条目，捕获鼠标按下事件onmousedown，前端会新建一个绝对定位的算子对象，该对象包含算子相关信息、定位信息以及含有一个独特标识符。页面监听鼠标移动事件onmousemmove，使得该算子对象随着鼠标的拖拽而移动。页面详情的中间是一个SVG画布，鼠标拖拽算子对象到SVG画布上，画布监听到鼠标松开事件onmouseup，认为算子进入画布，在画布内生成一个相似的新对象，即DAG图的节点。用户通过拖拽，完成节点和节点之间的连线，描述算子之间的前置后置关系，便利快捷地进行任务编排。在连线过程中，需要对DAG图进行校验，平台对算子的参数进行解析，一个输入文件对应一个入度，一个输出文件对应一个出度，计算出每个算子的入度和出度后，节点的入度不能大于算子的入度，节点的出度不能大于算子的出度。用户进行错误的连线时，前端会出现提示，并取消错误连线。用户点击SVG画布中的每个算子，前端会获取算子的详细信息，显示在页面右侧，用户需要对算子参数信息进行填写。

图 4.12 可视化任务编排

平台将DAG图解析为由nodes和lines组成的JSON数据，JSON数据样例如图 4.13所示。nodes由多个node组成,一个节点对应一个node，保存有算子信息、节点位置、独特标识符数据。lines由多个line组成，一条连线对应一个line，包含前置节点信息、起始位置、后置节点信息和结束位置。每个node会有独特标识符，每个line利用node的独特标识符识别前置节点和后置节点。WEB服务器将JSON数据和数据库中的DAG\_node表、node\_parameter表和DAG\_line表进行映射，完成对DAG图数据的存储和读取。实现DAG图的保存和重新读取。



图 4.13 JSON数据样例展示

用户完成任务编排后，点击“保存”按钮，前端首先会完成DAG图的校验和节点参数的校验。其中DAG图的校验逻辑包括不能成环、所有节点的入度等于相应算子的入度、节点参数的校验逻辑为不能为空。符合校验逻辑后，前端调用新建应用的接口，并根据应用信息填写请求参数。后端在数据库中添加、修改应用的记录，并利用拓扑排序算法对DAG图数据进行解析，将DAG节点串行化，获取每个DAG节点的后置节点组和前置节点组，以拓扑排序的顺序，将DAG图解析数据以JSON数组的形式存储到app表的DAG\_info字段中。后端返回保存成功的信息，前端跳转到应用管理的界面。

#### 任务管理模块

任务管理模块中的功能和操作涉及到对数据库中的task表、task\_node\_status表、task\_output表和task\_node\_wait表进行操作，基本操作有新建任务、查看任务、查看任务列表，包括任务调度和可视化任务调度两个核心功能。

用户通过上侧导航栏进入任务管理界面，前端会调用后端接口，获取任务信息列表，在此页面显示。用户可以在此界面新建任务。用户选择任务列表上方的“新建”按钮，前端会显示弹窗，要求用户输入任务名称，选择目标应用，上传目标遥感图像。所有信息输入完成后，用户点击弹窗中的“新建”按钮，前端调用后端接口，并根据任务信息填写请求字段。后端在数据库中添加任务的记录。

服务端创建子任务调度服务的顺序图如图 4.14所示。服务端监听数据库中任务信息的变化，当发现有新任务创建时，会获取新任务的目标应用的DAG图解析数据，进行解析，将JSON数组的每一个对象视为一个子任务，初始化子任务运行状态，存储到task\_node\_status表中，并根据参数信息进行解析，生成子任务调度信息，存储到task\_node\_wait表中。

C:\Documents and Settings\Administrator\Desktop\Image2.EMF

图 4.14 创建子任务调度服务顺序图

服务端执行子任务调度服务的顺序图如图 4.15所示。服务端每隔5秒会读取task\_node\_wait表，获取所有子任务调度信息，利用任务调度状态进行筛选，获取状态为“可执行”的子任务调度信息，优先考虑性能较好的“空闲”客户端进行调度，并尽量使所有客户端处于忙碌状态。如客户端均处于“忙碌”状态则不再调度。服务端若收到子任务调度失败的消息，该调度失败的子任务优先级会高于其他可执行子任务，优先在性能较好的客户端上调度。调度失败次数过多的子任务不再调度，该子任务所属的任务不再执行，处于执行异常状态。C/S模式下的服务端可同时和多个客户端进行通信，通信过程中不需要等待客户端返回。多个客户端可并发运行任务节点。实现任务内的并发和任务间的并发。

C:\Documents and Settings\Administrator\Desktop\Image2.EMF

图 4.15 执行子任务调度服务顺序图

本系统通过实现任务内子任务间的并发和任务间的并发，加快任务调度速度。如果子任务和子任务之间存在前置和后置的关系，则认为两个子任务之间存在直接依赖关系；前置子任务的前置子任务和后置子任务的后置子任务之间存在间接依赖关系。存在直接或者间接依赖关系的子任务之间存在调度时间上的先后关系；不存在直接或者间接依赖关系的子任务之间不存在调度时间上的先后关系，可以通过分配不同的客户端实现子任务间的并发，降低任务的总体执行时间。通过实现将任务和任务分配在不同的客户端上并发执行，可以降低系统的总体执行时间。

本系统通过设计节点的运行状态实现子任务的调度和重调度。如图 4.16所示，节点的运行状态，分别为不可执行、可执行、正在执行、执行结束，其中可执行状态又可以进一步细分为3类。每个节点唯有在其所有前置节点变更为执行结束的状态时才可从不可执行状态变更为可执行状态。

一个节点运行结束后，会更新task\_node\_status表和task\_node\_wait表中相应节点的执行状态为执行结束，并利用task\_node\_wait表中后置节点信息，减少其后置节点的执行条件，更新遥感图像容器数据库中可执行节点的信息。服务端获取子任务调度列表并以“可执行”状态为条件进行筛选获得可执行子任务。可执行子任务一定具备可执行条件，不需要判断他是否属于特定任务，不需要按照固定的顺序调度。服务端给可执行子任务分配客户端，在调度过程中不断更新task\_node\_wait表中的调度状态。服务端可以同时和多个客户端进行通信，原理是通过task\_node\_wait表中的调度状态识别当前通信执行到哪一步，进而执行下一步，发送特定消息。当客户端发生故障返回错误信息或者长期不返回时，服务端视为此子任务调度失败，通过重调度或重新分配客户端的方法使子任务恢复调度成功的状态，不影响后续节点的执行。当调度失败次数过多，服务端判定此为不可修复的故障，返回错误码，需要管理员检查调度环境。

C:\Documents and Settings\Administrator\Desktop\Image2.EMF

图 4.16 子任务调度服务状态图

节点的执行分为3个步骤：输入文件的发送、容器执行脚本的发送、输出文件的回传，服务端步骤详情如下：

1. 输入文件的发送：服务端从task\_node\_wait表中获知哪些调度参数为输入文件，将输入文件从服务端传递到客户端的指定目录，传递完成，即可视为容器运行的条件已经完成。
2. 容器执行脚本的发送：服务端发送容器执行相关参数，并通过构建输入文件数据卷和输出文件数据卷，实现容器虚拟环境下和主机目录之间的文件传递。客户端接收到容器执行的命令和参数后，会执行相应的容器。
3. 输出文件的回传：在执行完成后，客户端向服务端发送执行完成的消息并回传输出文件。服务端接收输出文件并将文件复制、重命名至该节点的后置节点所需的输入文件相对应的目录。

图 4.17展示了预处理节点执行过程中服务端的输出，随着步骤的不断执行，服务端更新节点调度状态。

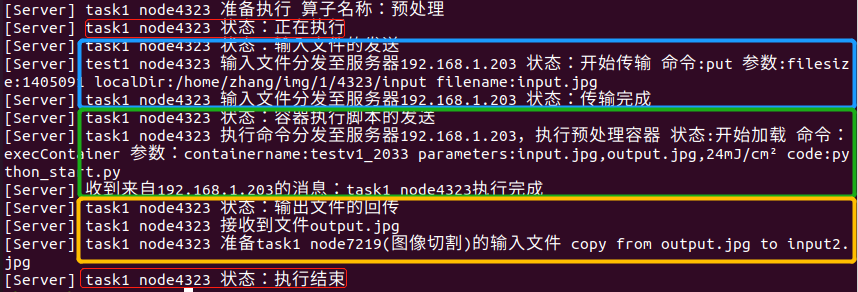


图 4.17 预处理节点执行过程下，服务端的输出

客户端会接收文件、处理命令，转去执行对应的脚本。

用户可以在任务管理页面选中一条任务，点击“详情”按钮会进入任务详情界面，进行工作流的监控。用户可以在实时查看任务调度过程，包括客户端的分配情况以及执行情况，并获取最终调度结果。任务详情界面效果图如图 4.18所示。页面利用SVG画布实现了工作流的监控。首先根据页面大小，将客户端和子任务视为单位，均匀分布在页面的上册和左侧，随后获取该任务的DAG任务流以及子任务调度信息，包括分配的客户端信息和调度状态信息，将子任务封装为组件，在页面位置上放置，和目标机以及算子形成对应关系。在SVG画布上绘制连线，描述子任务之间的依赖关系和子任务的调度情况。其中尚未调度的任务还没有被分配客户端，所以暂不显示。

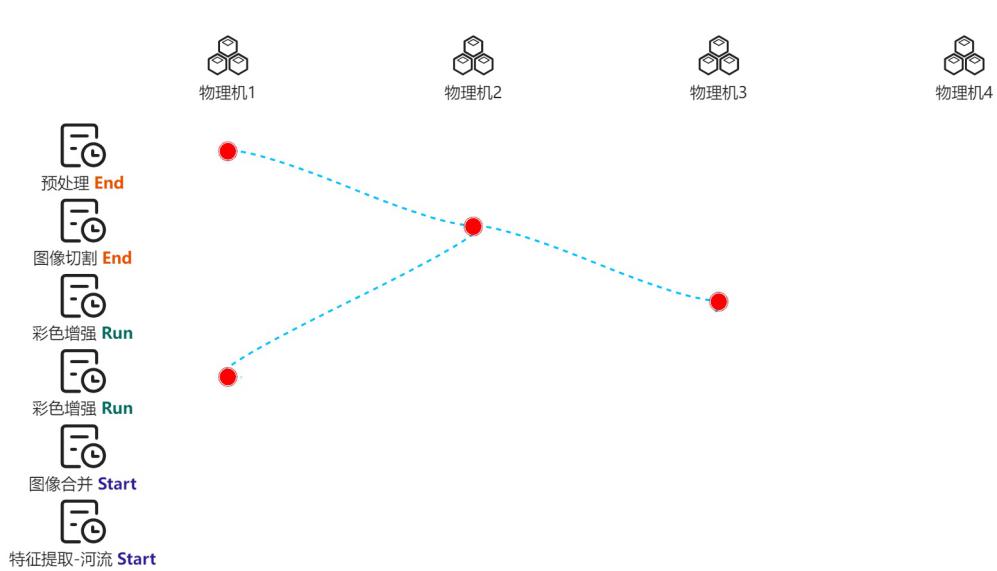


图 4.18 任务详情界面效果图

# 遥感图像处理任务可视化编排系统测试和评估

## 测试目标

针对系统的功能性需求和非功能性需求设计测试用例，保证系统能对使用者操作进行正常的处理的同时，实现系统的非功能性需求。在测试过程中，帮助研发人员发现系统缺陷以及不足，并进行纠正。

## 测试环境

测试过程中，数据库和系统代码均部署在远程服务器上，用户连接内网，在本地浏览器测试系统。其中远程服务器的系统测试环境为：

CPU：4核

内存：16GB

外存：40GB

操作系统：Linux

数据库：MYSQL 8.0

本地浏览器的测试环境为：

CPU：Intel(R)Core(TM) i5-6300HQ CPU @2.30GHz

内存：8GB

外存：1TB

浏览器：Google Chrome

## 功能性测试

系统的功能性测试围绕系统的3个功能模块设计了测试用例，对模块的每个功能点的测试均通过，证明系统能够满足遥感图像处理任务可视化编排系统的所有功能性需求，达到最初的设计目标。

### 用户管理测试

表 5.1 用户管理测试样例

| 编号 | 功能描述 | 测试步骤和期望目标 | 测试结果 |
| --- | --- | --- | --- |
| 1.1 | 登录账号 | 用户输入用户名和密码后，点击登录按钮后，用户可以进入系统。 | 达到期望目标 |
| 1.2 | 查看个人资料 | 用户点击查看个人资料按钮后，查看到自己的用户名、邮箱、手机号等个人信息页面。 | 达到期望目标 |
| 1.3 | 修改密码 | 用户单击“修改密码”按钮，依次输入旧密码和新密码，点击“确认修改”按钮，系统修改用户密码。 | 达到期望目标 |
| 1.4 | 退出登录 | 用户单击“退出登录”按钮，用户随机退出登录状态，重新返回到系统登录界面。 | 达到期望目标 |
| 1.5 | 修改用户资料 | 管理员点击修改按钮，输入修改后的用户信息，点击确认后可以看到修改后的用户详情信息。 | 达到期望目标 |
| 1.6 | 查看用户列表 | 管理员通过导航栏的“用户管理”进入用户管理界面，可以查看到用户列表信息 | 达到期望目标 |
| 1.7 | 删除用户 | 管理员在用户列表中选择一条用户信息，单击“删除”按钮，管理员将看到已删除的用户不再在用户列表中。 | 达到期望目标 |
| 1.8 | 创建用户 | 管理员点击新建用户按钮，输入用户的账号名、姓名、邮箱和账号备注信息，点击确认成功注册用户。 | 达到期望目标 |

### 客户端管理测试

表 5.2 客户端管理测试样例

| 编号 | 功能描述 | 测试步骤和期望目标 | 测试结果 |
| --- | --- | --- | --- |
| 2.1 | 新建客户端 | 管理员点击新建客户端按钮，输入IP、端口等信息，点击确认新建用户。 | 达到期望目标 |
| 2.2 | 查看客户端列表 | 管理员通过导航栏的“客户端管理”进入客户端管理界面，可以查看到客户端列表信息 | 达到期望目标 |

### 可视化遥感图像处理测试

#### 算子管理测试

表 5.3 算子管理测试样例

| 编号 | 功能描述 | 测试步骤和期望目标 | 测试结果 |
| --- | --- | --- | --- |
| 3.1 | 查看算子列表 | 用户通过导航栏的“算子管理”进入算子管理界面，可以查看到算子列表信息 | 达到期望目标 |
| 3.2 | 新建算子 | 用户点击新建算子按钮，输入算子名称、运行代码、参数等信息，点击确认新建算子。 | 达到期望目标 |
| 3.3 | 删除算子 | 用户选择要删除的算子条目，然后单击“删除”按钮，用户将看到已删除的算子不再在算子列表中。 | 达到期望目标 |
| 3.4 | 编辑算子 | 用户点击修改按钮，输入修改后算子名称、运行代码、参数等信息，点击确认后，可以看到修改后算子详细信息页面。 | 达到期望目标 |
| 3.5 | 解析算子 | 服务器监听到算子信息和客户端信息变化，生成算子构建服务保存到数据库 | 达到期望目标 |
| 3.6 | 部署算子 | 服务端获取算子构建服务，和客户端进行信息通信，完成算子构建 | 达到期望目标 |

#### 应用管理测试

表 5.4 应用管理测试样例

| 编号 | 功能描述 | 测试步骤和期望目标 | 测试结果 |
| --- | --- | --- | --- |
| 4.1 | 查看应用列表 | 用户通过导航栏的“应用管理”进入应用管理界面，可以查看到应用列表信息 | 达到期望目标 |
| 4.2 | 新建应用 | 用户点击新建应用按钮，拖拽算子进行可视化编排，进行连线，输入算子参数信息，点击确认新建应用。 | 达到期望目标 |
| 4.3 | 删除应用 | 用户选择要删除的应用条目，然后单击“删除”按钮，用户将看到已删除的应用不再在应用列表中。 | 达到期望目标 |
| 4.4 | 编辑应用 | 用户点击编辑按钮，可以看到已经保存好的应用信息，对编排信息和参数进行进行重新填写，点击保存后，应用保存并跳转到应用管理页面。 | 达到期望目标 |

#### 任务管理测试

表 5.5 任务管理测试用例

| 编号 | 功能描述 | 测试步骤和期望目标 | 测试结果 |
| --- | --- | --- | --- |
| 5.1 | 查看任务列表 | 用户通过导航栏的“任务管理”进入应用管理界面，可以查看到任务列表信息 | 达到期望目标 |
| 5.2 | 新建任务 | 用户点击新建任务按钮，输入目标应用、任务名称，并上传目标图像，点击确认新建任务。 | 达到期望目标 |
| 5.3 | 删除任务 | 用户选定当前要删除的一个任务条目，接着单击右侧“删除”按钮，用户将发现已被删除的任务条目不再出现在任务列表中。 | 达到期望目标 |
| 5.4 | 查看任务 | 用户点击任务按钮，可以进入任务详情界面，实时查看子任务调度情况和当前执行情况 | 达到期望目标 |
| 5.5 | 解析任务 | 服务端监听到到数据库中新建任务的消息，解析任务，分配客户端，生成子任务调度服务存储到数据库中 | 达到期望目标 |
| 5.6 | 调度子任务 | 服务端获取子任务调度服务，和客户端进行交互，实现任务内和任务间的并发 | 达到期望目标 |

## 非功能性测试

系统的非功能性需求主要反映在系统的性能、易用性、可扩展性、安全性和鲁棒性方面。

系统的性能体现在页面的响应较快、较稳定的文件传输和较优的任务并发调度。系统基本响应时间均在1s以内，满足了用户基本性能需要。普通遥感图像大小一般在100MB以内，传递一般在30s以内完成；较大的镜像，通常少于1GB，也能在5分钟内传输完成。服务端采用多线程，在传送文件的过程中，也可以和客户端进行消息通信，并尽量将任务同时分配到不同的客户端上，保持让客户端忙碌，处理信息，让服务端同时和多个客户端进行通信，降低任务调度所需要的时间，提高性能。

系统的易用性体现在构造逻辑清晰的界面和提供可视化图表分析数据。系统采用合理的布局和样式搭配，利用图标进行操作提示，风格简单明了，用户操作直观可见。管理人员利用可视化界面实现遥感图像处理系统云平台信息的维护，由服务端实现容器的自动调度，简化管理员和用户的操作。采用图表展示、拖拽绘图等可视化方法，让用户快捷获取所需要的信息，进行操作。平台统计任务运行情况，通过饼图清楚显示，如图 5.1所示。

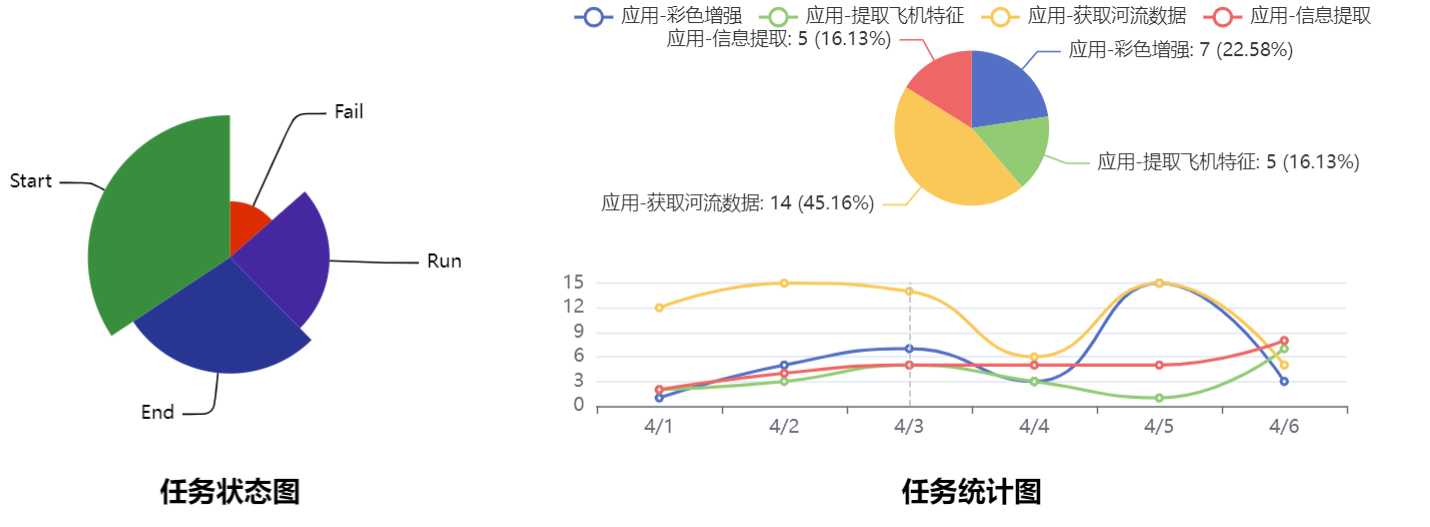


图 5.1 任务信息可视化图表

系统的可扩展性体现在对新建客户端的兼容性和功能模块化编写。当前系统连接有4个客户端，管理员想要创建新的客户端连接，只需要获取客户端代码，修改内部的IP、端口等配置信息，在任意一台Linux系统上运行即可。客户端通过Socket通过和服务端建立连接，服务端也会自动维护客户端连接数据。对于新建的客户端，平台提供了客户端管理界面可以对客户端的信息进行后续的新建和修改。平台将系统功能划分为模块，后续如要开发新的功能，可直接编写新的模块添加到系统中，不需要对原有项目进行较大的改动。

系统的安全性体现在用户和管理员身份的鉴别和数据接口的保护。前端页面无法被越权访问，当用户和管理员访问存在的路由时，系统会进行状态和权限判断，并针对越权操作的行为提示权限不够，如为未登录状态会跳转到登录界面，如为登录状态但权限不够则跳转到网站首页，防止用户误操作或有意攻击造成的数据破坏。另外，后端接口无法被越权调用，针对后端所有接口，在进行相关的处理前会先进行鉴权判断，判断调用者的权限，如果存在超出权限的调度操作，则不再进行后续的操作，直接返回权限错误的错误码。

系统的鲁棒性体现在系统对异常输入和内部错误的发现和处理。管理员新建用户时所填写的账号名称如果重复，系统则会显示该账号名称已存在并拒绝登录。用户在进行任务可视化编排过程中，容易出现不符合DAG建模规范的地方，比如成环或者不符合算子连接要求，系统会对用户的错误操作进行提示，效果如图 5.2所示，直到用户修改正确，才能提交应用。每个应用中的算子参数信息如果为空，会导致算子调度失败，所以当用户提交空白算子参数值时，系统也会进行提示。针对算子构建和子任务调度的过程，会收到网络和硬件等因素的干扰导致执行服务失败，本平台通过设计服务执行过程中的多状态转移，实现重构建和重调度，保证系统的正常运行。并针对无法修复的调度和构建问题，系统会返回错误码给管理员进行提示。



图 5.2 系统撤销不符合DAG建模规范的操作并提示

# 结论与展望

总结

本项目结合任务编排的痛点和遥感图像处理的实际需求，设计了遥感图像处理任务可视化编排系统，实现了可视化拖拽式编排和工作流的监控，以及云平台的自动管理。

本文首先进行需求调研，发现遥感图像专业人士在图像处理方面存在效率低下和操作复杂的痛点，并进行国内外研究现状调研，提出可视化云编排的理念，构建一个适用于遥感图像领域的任务编排系统。然后本文总结出总体实现目标，详细分析系统的功能性需求和非功能性需求。结合需求，本文先进行总体设计，对系统的部署架构和设计架构进行构思，提出采用B/S架构和C/S架构混合的三层架构模式，并利用类图和数据库表进行了数据库设计。在总体设计的基础上，对架构模式和功能模块的实现提出了详细设计，通过顺序图、状态图和流程图等形式阐明功能模块的设计细节。最后，作者将系统部署在远程服务器上，对系统的功能性需求和非功能性需求进行测试。项目符合预期目标，

本文完成了遥感图像处理任务可视化编排系统的设计与实现，基于B/S和C/S混合架构体系，实现了可视化任务编排和遥感图图像任务云上自动部署和调度，通过测试，基本实现预期目标。

本人工作内容

1. 进行需求分析，设计系统总体架构；
2. 编写前端界面代码，完成可视化编排设计；
3. 实现后端所有接口，完成用户管理、客户端管理、可视化遥感任务处理三大模块，构建基于B/S架构和C/S架构的三层混合架构；
4. 针对系统功能模块设计测试统计，完成非功能性测试。

展望

对照本文的研究目标，当前的遥感图像处理任务可视化编排系统已经基本满足前文描述的系统功能性需求和非功能性需求，但是在一些方面还是存在缺点，有许多改进的空间：

1. 调度算法的优化。本系统在进行任务调度时，会将子任务分配到不同的客户端上进行并发运行，当前调度速率还能接受，但是之后生成更多子任务和加入更多客户端时服务端调度响应时间会线性增加，因此需要考虑通过把客户端的处理任务、子任务的复杂性等因素加入调度算法中，尽量充分利用服务端和客户端的处理性能。
2. 镜像的管理。用户可能会上传几GB的较大镜像，如果用户同时上传较多镜像，这会给服务端造成较大的发送文件的压力。后续考虑实现统一的文件管理系统，实现服务端和客户端之间的文件共享。

# 致谢

这个项目是我大学以来第一次全栈式开发，在开发过程中，我对可视化技术有了更进一步的认识，对Django和Docker完成了从0到1的理解。在开发过程中，更是体会到学无止尽，让我更加谦虚，更加向往接下来的学习生涯。如今的圆满结束，随之而来的是大学生涯的结束。告别骤然而至，还是些许迷惘。

首先，我要感谢沃天宇老师，作为我的导师，他仔细认真、严谨求实，给了我极大的帮助。得益于沃老师为我创造的实验机会，我能在工作环境舒适的工位上专心编码，并使用实验室提供的服务器进行系统部署和测试。在过去的半年里，沃老师关注实验进度，在实验进展方面提出了专业意见，让我对实验的思考更加深入，对项目进行了进一步的优化。在论文完成工程中，从选题到论文撰写，沃老师认真严谨，严格把关，指出稿件中的具体问题，提供了最为中肯的建议。在论文撰写方面，因个人经验不足，出现许多低级问题，沃老师都非常耐心地解答。

其次要感谢实验室学长迂博和马博的支持。开题后，我对项目的开发方向存在错误理解，对项目的落地技术更是一头雾水，还好有实验室的学长细心指导，为我分析题目，纠正实验思路，在代码编写方面提出了细致的建议，在论文撰写提出了很多中肯的修改建议。

其次，我要感谢我在大学生活中遇到的每一位老师和导员。老师们教授的专业知识让我不断明白学无止尽的道理，老师们认真的态度更是让我明白学习不可松懈的道理。感谢老师们的谆谆教诲，让我更加坚定了往后余生要不断努力的方向。

同时，我要感谢我在大学生活中认识的每一个人。感谢你们陪我度过四年的快乐和难过，在生活和学业中给予帮助。在大学四年中，在学业和生活的迷茫之际，因为有你们的陪伴，我才顺利坚持熬过低谷，走向更美好的明天。

最后，我要特别感谢我的父母，感谢在二十年多来的生活中，父母给予我的细致关怀，还有对我在人生道路上的帮助。

参考文献

1. Matsunobu Lysha M., Pedro Hugo T.C., Coimbra Carlos F.M.. Cloud detection using convolutional neural networks on remote sensing images[J]. Solar Energy,2021,230.
2. 常志明. 遥感图像并行算法的研究及其网格服务的实现[D].国防科学技术大学,2004.
3. 郭延辉. 高光谱遥感图像分类技术研究[D].陕西师范大学,2020.DOI:10.27292/d.cnki.gsxfu.2020.000010.
4. 赵忠明,高连如,陈东,岳安志,陈静波,刘东升,杨健,孟瑜.卫星遥感及图像处理平台发展[J].中国图象图形学报,2019,24(12):2098-2110.
5. Aliyun. AI Earth[EB/OL]. <https://engine-aiearth.aliyun.com/#/,>2022-04-05.
6. SenseTime. SenseEarth[EB/OL]. <https://rs.sensetime.com/#tec-s-title,>2022-04-05.
7. 城市数据派. 商汤科技SenseEarth平台:遥感影像实时智能解译[EB/OL]. <https://www.sohu.com/a/326117105_650480>,2022/5/3.
8. 李乔,郑啸.云计算研究现状综述[J].计算机科学,2011,38(04):32-37.
9. Kivity A, Kamay Y, Laor D,et al. kvm: the Linux virtual machine monitor[C]. Linux Symposium.2007.
10. Al I P. Xen 3.0 and the Art of Virtualization[J]. proceedings of the ottawa linux symposium,2005.
11. Docker Inc. Home - Docker[EB/OL]. <https://www.docker.com/>,2022-04-01.
12. Brian Goff. CHANGELOG --rocket[EB/OL]. <https://rocket.readthedocs.io/en/latest/CHANGELOG/>,2022-04-01.
13. 金子威. 基于K8S的Docker分布式容器自动化运维系统的设计与实现[D].中南民族大学,2018.
14. Deepak Vohra. Docker Management Design Patterns[M]. Apress, Berkeley, CA: Apress,2017,45.
15. The Kubernetes Authors. Kubernetes Documentation[EB/OL]. <https://kubernetes.io/docs/home/>,2022-04-27.
16. The Apache Software Foundation. Apache Mesos - Documentation Home[EB/OL]. <https://mesos.apache.org/documentation/latest/>,2022-2-20/2022-04-11.
17. 秦志威,栗娟,刘晓,朱梦圆.端边云协同环境下能耗感知的工作流实时调度策略[J/OL]. http://kns.cnki.net/kcms/detail/11.5946.TP.20211108.1227.008.html,2021-11-08/2022-4-10.
18. The Apache Software Foundation. Apache Airflow Documentation[EB/OL]. <https://airflow.apache.org/docs/apache-airflow/stable/index.html>,2022-03-14.
19. The Luigi Authors. Luigi Documentation[EB/OL]. [https://luigi.readthedocs.io,](https://luigi.readthedocs.io)2022-04-05.
20. Argo Project Authors. Argo Workflows[EB/OL]. <https://argoproj.github.io/workflows/>,2022-3-14/2022-04-15.
21. 孙庚泽.基于Docker的混合云应用编排方案研究[D].西安电子科技大学,2015.
22. 高畅.大数据可视化编排与展示系统的设计与实现[D].北京邮电大学,2018.
23. 郭爱春,李洪全,丁浩,商和龙.基于可视化云编排技术的智能终端业务应用[J].电工技术,2021(22):161-163.DOI:10.19768/j.cnki.dgjs.2021.22.054.
24. Pradipta Ghosh, et al. Container Orchestration for Dispersed Computing[A]. Proceedings of the 5th International Workshop on Container Technologies and Container Clouds[C]. Davis, CA, USA: Association for Computing Machinery, 2019: 19–24.
25. 开彩红,肖瑶,方青.基于人工蜂群算法的中继卫星任务调度研究[J].电子与信息学报,2015,37(10):2466-2474.
26. 黄科.基于微服务的虚拟机自动化编排系统的设计与实现[D].电子科技大学,2019.
27. 姚婧,宋锐,王壮,胡卫东.空间探测相控阵雷达系统的观测任务编排仿真[J].计算机仿真,2006(10):65-68+71.
28. 刘伟权. 野外WSN的监测系统任务编排及调度的实现[D].西安工业大学,2015.
29. X. M. Li, W. X. Wang, S. J. Tang, et al. A NEW CLOUD-EDGE-TERMINAL RESOURCES COLLABORATIVE SCHEDULING FRAMEWORK FOR MULTI-LEVEL VISUALIZATION TASKS OF LARGE-SCALE SPATIO-TEMPORAL DATA[J]. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 2020, XLIII-B4-2020:477-483.
30. Eliseu Germano, et al. Workflow Management Platform for Orchestration of Ubiquitous Care Plan Services[A]. Proceedings of the Euro American Conference on Telematics and Information Systems[C]. Fortaleza, Brazil: Association for Computing Machinery, 2018: 8.
31. 孙锐.鲁棒性\_百度百科[EB/OL]. [https://baike.baidu.com/item/鲁棒性/832302,](https://baike.baidu.com/item/%E9%B2%81%E6%A3%92%E6%80%A7/832302) 2022-04-23.
32. 李臣亮.事件驱动架构及应用[J].软件世界,2007(21):44-45.
33. 科睿思博.软件架构分类(转载) - 博客园[EB/OL]. <https://www.cnblogs.com/mikechang/p/5838845.html>, 2022-05-13.
34. 赵然,朱小勇.微服务架构评述[J].网络新媒体技术,2019,8(01):58-61+65.
35. 王湘文,陈建伦,陈纪铭.分层软件架构设计及其应用研究[J].福建电脑,2011,27(06):55-56+93..
36. 廖志英,董安邦.基于C/S和B/S混合结构的管理信息系统运行模式[J].计算机工程与应用,2002(02):184-185+249.
37. 李云云.浅析B/S和C/S体系结构[J].科学之友,2011(01):6-8.
38. 籍慧文.B\S和C\S的架构分析[J].电子技术与软件工程,2017(05):202.
39. S. Priyank,V. G. Silva, M. Kirikova , et al . An analysis of container-based virtualization technologies[J]. Asian Journal of Multidimensional Research,2021,10-11.
40. Yuqian Lu, Xun Xu, Jenny Xu. Development of a Hybrid Manufacturing Cloud[J]. Journal of Manufacturing Systems,2014.
41. Sitaram D, Harwalkar S, Sureka C, et al. Orchestration Based Hybrid or Multi Clouds and Interoperability Standardization [C]. 2018 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), November 23-24, 2018, Bangalore,India, IEEE 2018:67-71.
42. Imran ,Ahmad Shabir,Kim Do Hyeun. A task orchestration approach for efficient mountain fire detection based on microservice and predictive analysis in IoT environment[J]. Journal of Intelligent & Fuzzy Systems,2021,40(3).
43. Klein Peter,Mäuselein Maren,Cetin Nurhan. Development of innovative visualization and interaction concepts for dispatcher software by means of the user-centered design process[J]. VDI Berichte,2008.
44. 蔡凌峰,魏祥麟,邢长友,邹霞,张国敏.故障场景下的边缘计算DAG任务重调度方法[J].计算机科学,2021,48(10):334-342.
45. 齐舸. 异构云平台下多DAG工作流的节能调度策略研究[D].西安电子科技大学,2020.DOI:10.27389/d.cnki.gxadu.2020.000777.
46. Wenzhong Zhu,Hualong Jiang,Shuwen Zhou,Mike Addison. The Review of Prospect of Remote Sensing Image Processing[J]. Recent Patents on Computer Science,2017,10(1).
47. Stupina A , Zhukov E A , Ezhemanskaya S N , et al. Question-answering system[J]. Iop Conference, 2016, 155:012024.
48. 伍尚锋,李洪赭,李赛飞.一种多云环境中应用编排方案设计[J].微型电脑应用,2021,37(05):1-4.