****



本 科 毕 业 设 计

院 系

专 业

题 目

年 级 学 号

学生姓名

指导教师 职 称

提交日期

南京大学本科生毕业论文（设计）中文摘要

毕业论文题目：

院系 专业 级本科生姓名：

指导教师（姓名、职称）：

摘要：

\*\*项目背景(为什么要做这个项目)。

\*\*技术简介(项目开发中使用了哪些技术，有何特点)。

\*\*项目组整体完成了项目中的哪些功能。

\*\*本人在项目中承担了哪些工作。

关键词：

南京大学本科生毕业论文（设计）英文摘要

THESIS：

DEPARTMENT：

SPECIALIZATION:

UNDERGRADUATE:

MENTOR:

ABSTRACT：

The waves of Industry 4.0 promotes the traditional industry restructuring brought by IoT technologies greatly. IoT technologies play an important role in the reduction of the industrial process, the improvement of efficiency and providing better products and services.

The controller platform based on SDN framework satisfies the demand of IoT. The controller is able to deploy and config the entire basic infrastructure automatically by combining network management with application service into a platform which is flexible and extensible.

In the controller platform, the interaction between the controller and devices is crucial. The module of device access is the critical module which is in charge of establishing the connection with devices, handling the messages reported from devices and passing the messages to devices. Meanwhile, the performance of the module determines the quality of the controller. If the device access module has a poor performance, the excellent performance of devices and the other modules of the controller would be wasted.

The module of data access uses the architecture of microservice. Spring Boot is employed as the basic framework to implement microservice. The data access module connects with the other modules by using Grpc, an open source RPC framework. The TCP connection between the module and devices is implemented by Mina. Besides, the whole controller separates the front-end and back-end. As for front-end, we use AngularJS ,CSS3 and HTML5. An Nginx Server is employed to complete the interaction between the front-end and back-end.

The controller platform is huge, providing a sophisticated business solution. We hope the industry manage numerous devices with a very low human cost.

I am responsible for developing the data access module. The developing of the connection between the data access module and the other modules and the TCP mock device for the test and verify is my main job.

KEY WORDS: \*\*项目，\*\*技术(关键名词)1，\*\*技术(关键名词)2，…..

SDN架构 简介：

1. SDN 简介：

SDN 即 Software Defined Network(软件定义网络),

之所以这样做一是因为这样可以减轻设备侧的负载，当所有的业务和逻辑由控制层完成之后，交换机只会简单的负责报文的转发工作，减轻运营商基础设施的压力；二是这样可以对业务快速扩展，允许业务更加复杂。由于使用了 SDN 架构，业务逻辑的处理都有控制器来做，当要扩展或修改业务逻辑时，主要修改的是控制器，对设备的改动不大。

Spring Boot 简介

Mina 简介

Grpc 简介

目 录

图目录 4

表目录 5

第一章 引言 1

1.1 项目背景 1

1.2 国内(外)工作流建模技术研究现状 2

1.X可补充二级标题 3

1.X.1 可补充三级标题 3

1.X.2 可补充三级标题 3

1.Z 论文的主要工作和组织结构 3

第二章 \*\*技术概述 4

2.1 \*\*技术 4

图2.1 \*\*图 4

2.1.1 \*\* 4

图2.2 \*\*图 4

2.1.2 \*\* 4

表2.1 \*\*表 4

2.2 \*\*技术 4

2.X \*\*本章小结 4

第三章 \*\*系统需求分析与概要设计 6

3.1 \*\*项目整体概述 6

3.1.1 \*\*可补充三级标题 6

3.1.2 \*\*可补充三级标题 6

3.2 \*\*系统的需求分析 6

3.2.1 业务流程 6

3.2.2 \*\*可补充三级标题 7

3.3 \*\*系统的概要设计 7

3.3.1 \*\*可补充三级标题 7

3.3.2 \*\*可补充三级标题 7

3.X \*\*本章小结 7

第四章 \*\*项目\*\*模块的详细设计与实现 8

4.1 \*\*模块概述 8

4.2\*\*模块的详细设计 8

4.2.1 \*\*可补充三级标题 8

4.2.2 \*\*可补充三级标题 8

4.3 \*\*模块的实现 8

4.3.1 \*\*可补充三级标题 8

图4.1 MyWorkController类代码 9

4.3.2 \*\*可补充三级标题 9

4.X \*\*本章小结 10

第五章 总结与展望 11

5.1 总结 11

5.2 展望 11

参考文献 12

致谢 13

图目录

图目录 4

表目录 5

第一章 引言 1

1.1 项目背景 1

1.2 国内(外)工作流建模技术研究现状 2

1.X可补充二级标题 3

1.Z 论文的主要工作和组织结构 3

第二章 \*\*技术概述 4

2.1 \*\*技术 4

图2.1 \*\*图 4

图2.2 \*\*图 4

2.2 \*\*技术 4

2.X \*\*本章小结 4

第三章 \*\*系统需求分析与概要设计 6

3.1 \*\*项目整体概述 6

3.2 \*\*系统的需求分析 6

3.3 \*\*系统的概要设计 7

3.X \*\*本章小结 7

第四章 \*\*项目\*\*模块的详细设计与实现 8

4.1 \*\*模块概述 8

4.2\*\*模块的详细设计 8

4.3 \*\*模块的实现 8

图4.1 MyWorkController类代码 9

4.X \*\*本章小结 10

第五章 总结与展望 11

5.1 总结 11

5.2 展望 11

参考文献 12

致谢 13

表目录

图目录 4

表目录 5

第一章 引言 1

1.1 项目背景 1

1.2 国内(外)工作流建模技术研究现状 2

1.X可补充二级标题 3

1.Z 论文的主要工作和组织结构 3

第二章 \*\*技术概述 4

2.1 \*\*技术 4

表2.1 \*\*表 4

2.2 \*\*技术 4

2.X \*\*本章小结 4

第三章 \*\*系统需求分析与概要设计 6

3.1 \*\*项目整体概述 6

3.2 \*\*系统的需求分析 6

3.3 \*\*系统的概要设计 7

3.X \*\*本章小结 7

第四章 \*\*项目\*\*模块的详细设计与实现 8

4.1 \*\*模块概述 8

4.2\*\*模块的详细设计 8

4.3 \*\*模块的实现 8

4.X \*\*本章小结 10

第五章 总结与展望 11

5.1 总结 11

5.2 展望 11

参考文献 12

致谢 13

第一章 引言

1.1 项目背景

随着工作流建模技术的不断发展和进步，工作流建模在工业界受到越来越多的重视。与此同时，越来越多的工作流建模工具不断涌现。常见的工作流建模和管理工具和技术包括：BPMN[[1]](#footnote-1)、BPEL[[2]](#footnote-2)、UML[[3]](#footnote-3)、Aris tools[[4]](#footnote-4)、ProM[[5]](#footnote-5)、YAWL[[6]](#footnote-6)等，它们采用的语言或是形式化能力不足（如BPMN、BPEL、UML和Aris tools中的EPC语言，这些符号建模语言有助于图形化清楚地描述工作流模型，但无形式化分析能力），或是有一定的形式化能力，但支持的力度不够（如ProM、YAWL二者均基于Petri网，但只支持控制流维度的描述）。因此，亟需一种能够形式化地对工作流进行统一建模的语言。着色Petri网，在基本的Petri网基础上引入数据类型，使得基于着色Petri网，提出一种支持控制流维度，数据维度和资源维度的工作流建模语言成为一种可能[1]。

为了便于对工作流模型进行建模和分析，我们需要将现有模型统一到PNML的文件类型，将EPC等现有业务模型转化到Petri 网模型，以实现统一的建模和分析。

如今，基于经典Petri 网的建模工具正在被越来越多的人接受，也被越来越多的人使用，我们不能忽视经典Petri网在工作流建模中的强大作用，但是，经典Petri网也具有很大的局限性：

1. 没有测试库所中零令牌的能力
2. 模型容易变得很庞大
3. 模型不能反映时间方面的内容
4. 不支持构造大规模模型，如自顶向下或自底向上

为了解决这些问题，国内外很多研究人员都在研究着色Petri网(Colored Petri Net)以及基于时间的Petri 网(Timed Petri Net)。目前，在着色Petri网的研究中，实现最好最完整的工具是CPN Tools，但这个工具不是开源的，这就造成了我们在扩展上的困难。

为了满足工作和研究需要，我们需要建立一个以PNML为文件存储格式，以着色Petri 网为底层模型，开源的工作流建模工具。

1.2 国内(外)工作流建模技术研究现状

SDN 起源于 2006 年斯坦福大学的 Clean Slate 研究课题[2].2009 年,Mckeown 教授正式提出了 SDN 概念[3]. 利用分层的思想,SDN 将数据与控制相分离. 控制层负责业务逻辑处理，数据层负责报文的收发。此时 SDN 只是一种概念，SDN 要求控制层和数据层共同遵守一个接口标准，但是这个接口的设计并未详细给出一个实现。

2013年，ONF(Open Networking Foundation) 开放网络基金会在提出了SDN架构，以Openflow作为接口标准。它强调控制与数据的分离，但其应用并不广泛，被用于校园网、企业网和数据中心。

NFV架构是由由运营商提出的更适用于运营商业务的SDN架构。

基于Petri网的工作流建模技术在国内外研究机构及工业界中，都得到了广泛的应用。在基于Petri网的工作流建模、分析和管理工具中，比较常见的有：YAWL、PIPE[[7]](#footnote-7)、ProM等，它们基于不同的模型，侧重于工作流建模的不同方面。虽然它们各自解决了工作流建模不同方面的问题，但它们在开源性、可扩展性以及分析算法支持方面存在一定的问题。

当前国内外基于Petri网的工作流建模相关的技术包括底层文件存储类型、仿真技术、数据分析技术等方面，它们的研究现状如下：

…………………

1.X可补充二级标题

1.X.1 可补充三级标题

1.X.2 可补充三级标题

1.Z 论文的主要工作和组织结构

本文主要介绍了Petri网和着色Petri网的相关理论知识，以及一个基于PNML的工作流建模工具的设计和实现。主要是对使用Petri网进行工作流建模的理论知识进行描述，同时描述一个基于PNML的工作流建模系统的设计和实现过程以及这个过程中出现的一些问题。

第一章：概述和前言部分，主要介绍了项目背景，当前工作流建模技术的研究及现状 ，并描述了该论文的主要工作。

第二章：主要介绍基于PNML的工作流建模工具的开发和实现过程中用到的相关理论知识和理论研究。

第三章：从项目概述和需求分析两个方面，描述基于PNML的工作流建模工具的提出背景和开发者信息，同时分析和总结出功能性和非功能性需求。

第四章：从项目概要设计方面，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。

第五章：对系统不同模块的详细设计和实现进行描述，其中包括支持着色Petri网的PNML标准的制定和各个模块的类设计图。同时描述了本系统与BeehiveZ2集成的实现以及相应数据库的设计。

第六章：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。

第二章 \*\*技术概述

2.1 \*\*技术



图2.1 \*\*图

注：需要文字说明。

2.1.1 \*\*



图2.2 \*\*图

注：需要文字说明。

2.1.2 \*\*

表2.1 \*\*表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |

注：每章开始另起一页，不是一章的末尾页，页内不要有大幅“留白”。

2.2 \*\*技术

2.X \*\*本章小结

第三章 \*\*系统需求分析与概要设计

3.1 \*\*项目整体概述

3.1.1 \*\*可补充三级标题

3.1.2 \*\*可补充三级标题

3.2 \*\*系统的需求分析

3.2.1 业务流程

设备接入模块承担所有需要控制器与设备交互的业务。对众多的业务进行抽象后，我们可以将控制器与设备交互的业务分为2类：控制业务(如：设备上线、设备下线等)和数据传输业务(如：上传升级软件包、上传补丁文件等)。这2种业务皆由控制器和设备之间达成的通信协议所承载。协议内部包含一次业务下发需要的所有内容，包括业务的类型、设备的信息、所传输的信息等。整个设备接入层的业务就是将控制器业务转换为协议并下发给设备的过程。图A展示了设备接入模块的活动流程泳道图。

3.2.2 \*\*可补充三级标题

在每个用例描述后加系统顺序图

写非功能需求

总结展望

致谢

3.3 \*\*系统的概要设计

3.3.1 \*\*可补充三级标题

3.3.2 \*\*可补充三级标题

3.X \*\*本章小结

第四章 \*\*项目\*\*模块的详细设计与实现

4.1 \*\*模块概述

4.2\*\*模块的详细设计

4.2.1 \*\*可补充三级标题

4.2.2 \*\*可补充三级标题

4.3 \*\*模块的实现

4.3.1 \*\*可补充三级标题

在实现部分，可以有少量关键性的代码，代码的排版形式，代码贴入表格框(1行\*1列)，设置为可以跨页的形式，且代码中的字体使用“Arial/五号/单倍行距”，对该表格框标记为图x.y，并给出以该段代码的用途给出图名。示例如下：

|  |
| --- |
| public String signAndUnsignList() throws Exception {  String resultStr = SIGN\_UNSIGN\_LIST\_RESULT;  Loginer loginer = this.getLoginer();  Page page = null;  CriteriaSetup criStp = new CriteriaSetup();  try {  //设置界面不分页  this.ecLimit = ExtremeTablePage.getLimit(this.getHttpServletRequest(), 999999);  page = this.myWorkextSvc.FindTasksOfActorId(loginer, new Long(ProcessVariable.JBPM\_FLOW\_TASK\_ONLINE),  this.ecLimit, criStp);  } catch (Exception ex) {  Log.debug(ex);  throw new SysException(SysExcepType.BUG\_UNKNOWN\_RUNTIME\_EXCEPTION, ex);}  List list = page.getResult();  //排序  Comparator comp = new Comparator(){  public int compare(Object o1,Object o2) {  TaskBillVO v1=(TaskBillVO)o1;  TaskBillVO v2=(TaskBillVO)o2;  if((v1.getDuration()==null || v1.getDuration().equals("")  || v1.getDuration().equals("null")) && (v2.getDuration()!=null  && !v2.getDuration().equals("") && !v2.getDuration().equals("null")))  return 1;  else if((v2.getDuration()==null || v2.getDuration().equals("")  || v2.getDuration().equals("null")) && (v1.getDuration()!=null  && !v1.getDuration().equals("") && !v1.getDuration().equals("null")))  return 0;  else  return 1;  }  };  Collections.sort(list,comp);  this.setTaskbillVOs(list);  getCaseNum(list);  this.getHttpServletRequest().setAttribute("totalRows", page.getTotalCount());  return resultStr;} |

图4.1 MyWorkController类代码

4.3.2 \*\*可补充三级标题

4.X \*\*本章小结

第五章 总结与展望

5.1 总结

5.2 展望

参考文献

[1] 作者,译者,书名(版本),出版地:出版社,出版时间,引用部分起止页.

[2] 作者,译者,文章题目,期刊名,年份,卷号(期数):引用部分起止页.

[3] 作者,学位论文名,本科/硕士/博士论文,大学/机构名,年份.

[4] 网页的主题,URL.

致谢

感谢指导老师……

感谢参与本项目的同组的其他同学……

感谢其他……

1. 关于BPMN的详细信息，参见：http://www.bpmn.org/ [↑](#footnote-ref-1)
2. 关于BPEL的详细信息，参见：http://en.wikipedia.org/wiki/Business\_Process\_Execution\_Language [↑](#footnote-ref-2)
3. 关于UML的详细信息，参见：http://en.wikipedia.org/wiki/Unified\_Modeling\_Languag [↑](#footnote-ref-3)
4. 关于Aris的详细信息，参见：http://www.ids-scheer.com/en/ARIS\_ARIS\_Platform/3730.html [↑](#footnote-ref-4)
5. 关于ProM的详细信息，参见：http://prom.sf.net/ [↑](#footnote-ref-5)
6. 关于YAWL的详细信息，参见：http://www.yawlfoundation.org/ [↑](#footnote-ref-6)
7. 关于PIPE的详细信息和介绍，参见：http://pipe2.sourceforge.net/ [↑](#footnote-ref-7)