2019.7

项目汇报

报告人：周　超

目 录

PART ONE

自我介绍

Self-introduction

自 我 介 绍
Self-introduction

01

周超，2017.6毕业于河海大学,通信工程专业。

02

2017.7就职于华苏科技，职位通信工程师。

03

2018.2就职于中软国际，职位软件测试。

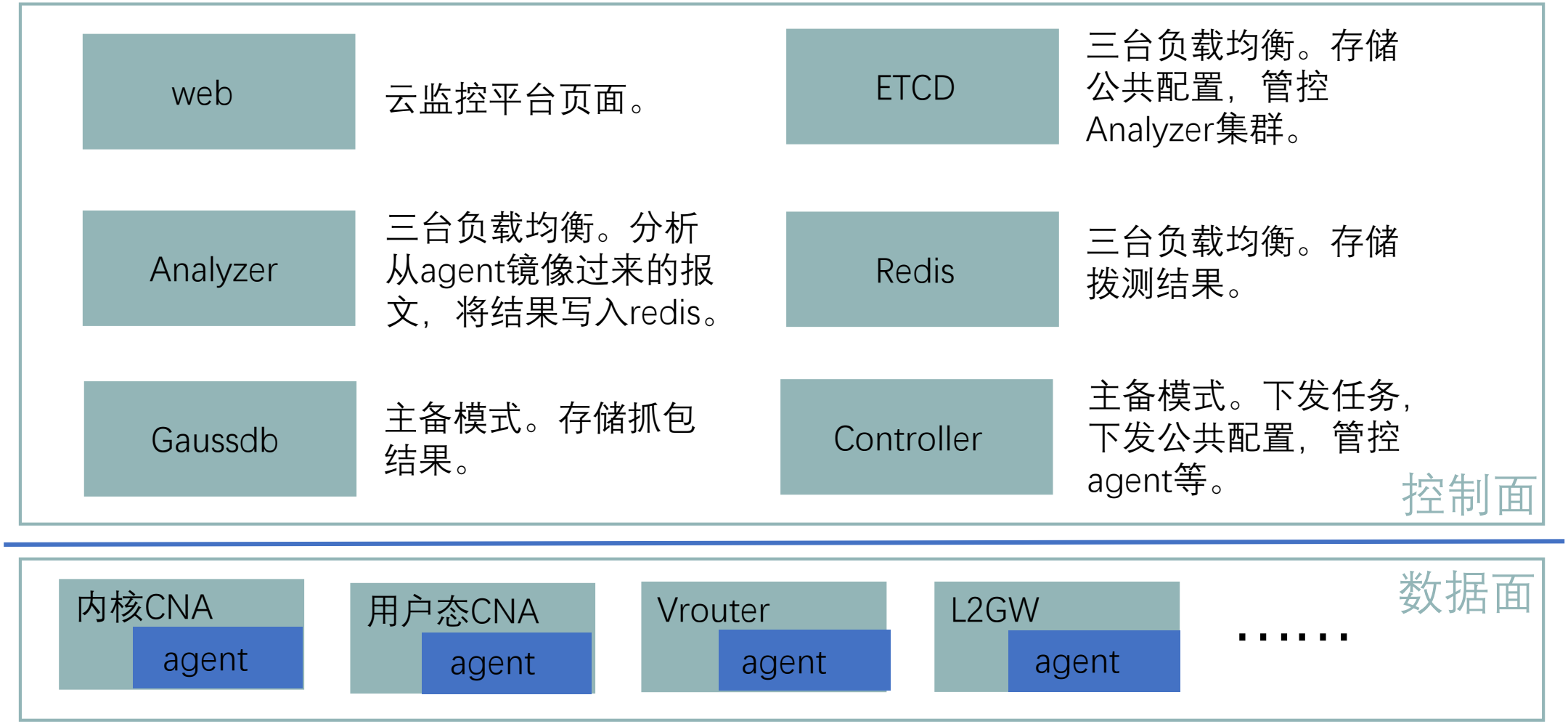
PART TWO

项目简介

Project Brief

项 目 简 介
Project Brief

项目名称：云监控系统
项目描述：实现抓包及拨测功能，实时稳定地监控云中的数据流信息、高效地定位各种云网络场景的异常。



项 目 简 介
Project Brief

拨测：

数据面不同类型的节点组合成各种不同的网络场景，由controller通过post方式下发任务，起点节点的agent构造报文，并根据controler下发的配置将报文传递给场景的下一节点，途经报文节点的agent将报文镜像给analyzer进行分析。

抓包：

由controller下发抓包任务，相关agent节点执行抓包任务，并将结果写入到gaussdb。

流检测：

数据流在网络中走不同的流向，如果遇到数据传输遇阻，自动拉起相关拨测任务。

PART THREE

主要职责

Main Responsibilities

主要职责
Main Responsibilities

1

物理环境及虚拟环境的手动及自动化搭建

熟悉了安装操作系统的流程，配置相关网卡，挂载磁盘等。熟悉了虚拟机相关操作命令及网桥流表的相关配置。熟悉了交换机的简单命令。

主要职责
Main Responsibilities

a) 网桥

网桥工作在数据链路层，将两个LAN连起来，根据MAC地址来转发帧，可以看作一个“低层的路由器”（路由器工作在网络层，根据网络地址如IP地址进行转发）。

常用命令：

ovs-vsctl show	查看网桥和端口
ovs-vsctl add-br br0(网桥名)	创建网桥
ovs-vsctl add/del-port br0 eth1	添加删除端口
ovs-ofctl dump-flows br0	查看br0网桥当前流表
ovs-appctl dpif/show	查看接口id等

主要职责
Main Responsibilities

b) 虚机

虚机经常在物理机用xml文件创建。先获取系统镜像，再修改xml模板文件。

常用命令：

virsh list –all	查看虚机列表
virsh create xx.xml	创建并启动虚机
virsh destroy 虚机名	彻底删除虚机
virsh start 虚机名	开启虚机
virsh shutdown 虚机名	关闭虚机
virsh reboot 虚机名	重启虚机

主要职责
Main Responsibilities

c) 交换机

交换机是一个扩大网络的器材，能为子网络中提供更多的连接端口，以便连接更多的计算机。

把端口配到一个vlan后，连线相应端口的主机之间就可以网络互通了。

常用命令：

system-view	进入系统视图
quit	退出
disp vlan	显示vlan
vlan xx	进入vlanxx，如果不存在则创建
int e1/0/10	进入端口10（交换机id/交换机插槽位/第10个口）
port access vlan xx	把当前端口放入vlanxx
undo port e1/0/10	删除当前VLAN端口10
commit	提交

主要职责
Main Responsibilities

2

写私有云产品组件的自动化部署脚本，包括心跳，部署，升级，回退，入参等。

组件本来是手动装，后来需要对接到某些自动化部署平台进行自动化安装等一系列操作，控制面主要用python实现，数据面用shell和ansible实现。这一过程熟悉了python的基本用法、类与对象的关系、类的继承原理等；还熟悉了rpm的打包流程及打包配置文件spec的配置原理、ansible语言的基本语法及各模块的用法。

主要职责
Main Responsibilities

d)rpm

Rpm包用rpmbuild构建。

常用命令：

rpm –ivh xxx.rpm	安装rpm包
rpm -qa	查看已安装的rpm包
rpmbuild -bb xx.spec	构建rpm包
rpm -e xxx	卸载xxx组件

主要职责
Main Responsibilities

目录名	说明
BUILD	编译rpm包的临时目录
BUILDROOT	编译后生成的软件临时安装目录
RPMS	最终生成的可安装rpm包的所在目录
SOURCES	所有源代码和补丁文件的存放目录
SPECS	存放SPEC文件的目录(重要)
SRPMS	软件最终的rpm源码格式存放路径(暂时忽略掉，别挂在心上)

Spec文件
%description
#关于软件包的详细说明
%prep
#主要是对源代码包的解压和打补丁，而解压常用的指令就是
%setup -q
%build
#该阶段执行常见的configure和make操作，常见指令如:
#%configure
#make %{?_smp_flags}
%install
#该阶段执行make install操作，它会在%_buildrootdir目录里建好目录结构，然后将需要打包到rpm软件包里的文件从%_builddir里拷贝到%_buildrootdir里对应的目录里。最常见到的指令就是：
#rm -rf %{buildroot}
%clean
#编译完成后一些清理工作，主要包括对%{buildroot}目录的清空(当然这不是必须的)，通常执行诸如make clean之类的命令。
#rm -rf %{buildroot}
%files
#主要用来说明会将%{buildroot}目录下的哪些文件和目录最终打包到rpm包里，需要打包的文件在此列出。
#另外，defattr(文件权限，用户名，组名，目录权限)用来指定权限，如:
%defattr(-,root,root,-)，这条指令设置缺省权限。
%pre
#安装前的准备工作
%post
#安装后的工作，如设置服务的开机启动，定时任务的设定，启动服务等等
%preun
#卸载前的工作，例如停止服务，关闭服务的开机启动，删除定时任务

主要职责
Main Responsibilities

e) ansible

ansible <host-pattern> [-m module_name] [-a args] [options]
<host-pattern> 自己定义的主机组
[-m module_name] 指定模块
[-a args] 指定要执行的动作
例子：ansible test -m script -a "/sh/test.sh“

```
tasks:
  - name: Copy ansible inventory file to client
    copy:
      src: /etc/ansible/hosts
      dest: /etc/ansible/hosts
      owner: root
      group: root
      mode: 0644
```

主要职责
Main Responsibilities

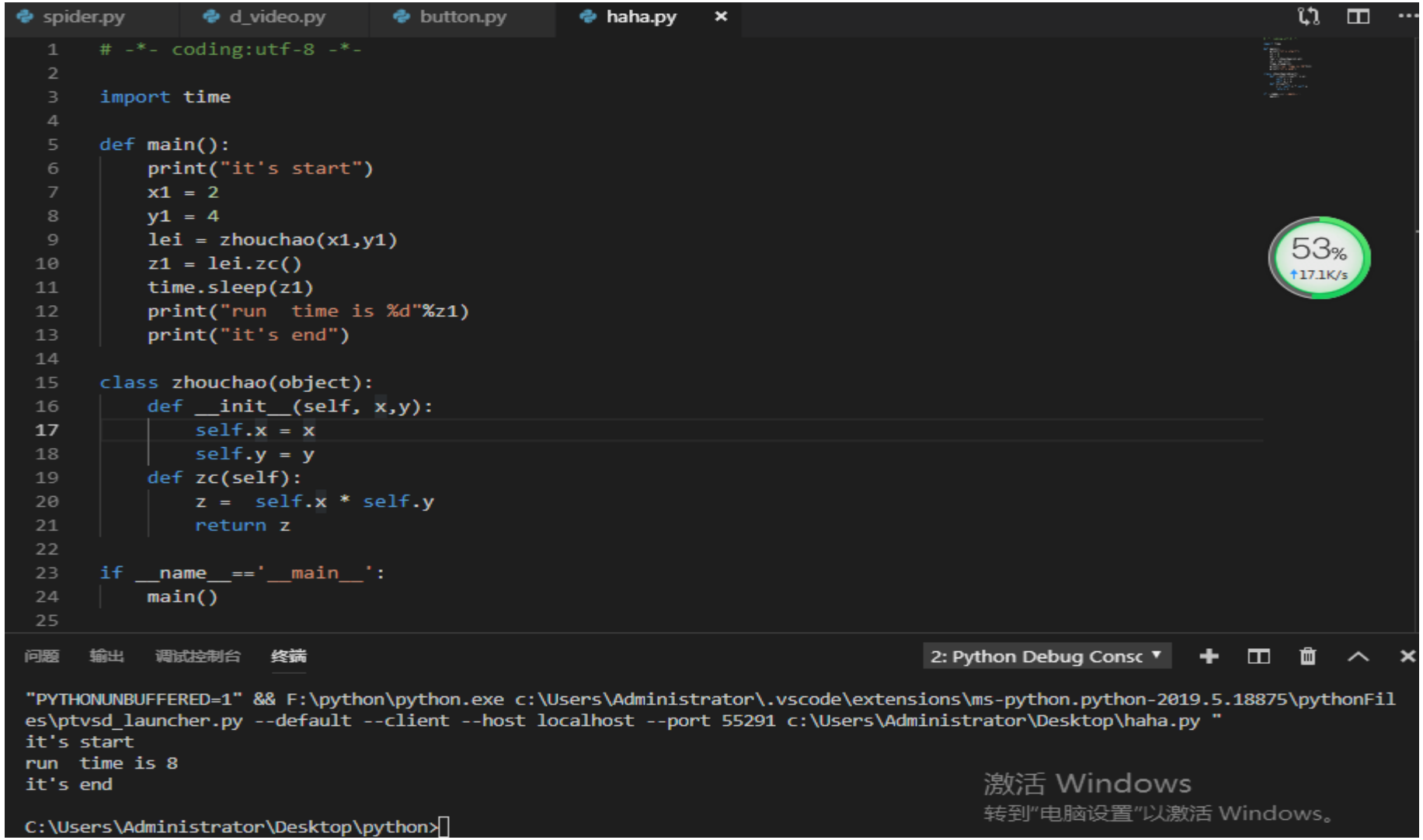
f) python基本语法及继承关系

Python是面向对象语言，很多功能都已经写好在库里了，用的时候只需要调用相应的包就行。

如果自己写的代码重复度很高，可以把这段代码封装在类里，然后用的时候调用类里的具体方法。

Python里的类可以相互一直调用，比如我要用类A里的方法实现某个功能，A的父类即是类B又是类C，但是B和C又有同一个父类D，所以如果我调用类A，类D可能执行两次，为了避免这种情况发生，可以使用super（）来继承。

主要职责
Main Responsibilities



主要职责
Main Responsibilities

3

CI的维护、产品组件自动化出包。

熟悉了Jenkins的基本操作方法；熟悉了从库上pull及向库push代码等Git操作命令、自动化出包用shell实现，熟悉了sed，cut，mkdir，tar，chmod，chown，scp，mv等shell操作命令。

主要职责
Main Responsibilities

g) git

常用命令：

git clone url	下载代码
git add .	添加当前目录下所有文件
git status	查看提交文件缓存区文件列表
git pull	拉取当前分支代码
git commit -m [message]	提交暂存区到仓库区
git branch	查看分支
git checkout 分支名	切换分支
git branch -b 分支名	新建分支并切换到此分支
git push origin 分支名	提交当前分支代码到库上分支名

主要职责
Main Responsibilities

4

编写产品的测试设计，文本用例，自动化用例并进行测试。

熟悉了测试的一整套流程及测试所要涵盖的点。熟悉了xmind，tmss等工具的用法。熟悉了python的unittest，selenium包。熟悉了jmeter的基本操作、正则表达式。

主要职责
Main Responsibilities

f) unittest测试套

```
import unittest
class MyTest(unittest.TestCase):
    def setUp(self):
        # 每个测试用例执行之前做操作
    def test_xxx(self):
        #具体测试操作
    def tearDown(self):
        # 每个测试用例执行之后做操作
if __name__=='__main__':
    test_suite = unittest.TestSuite()
    test_suite.addTest(unittest.makeSuite(MyTest))
    runner = xmlrunner.XMLTestRunner(output='report')#指定报告放的目录
    runner.run(test_suite)
```

主要职责
Main Responsibilities

h) web自动化测试包selenium

```
1   # -*- coding:utf-8 -*-
2   import requests
3   import webbrowser
4   import re
5   #import click
6
7   def main():
8       url = input("请输入url:")
9       #webbrowser.open(url)
10      url = requests.get(url)
11      '''
12      f = open('F:\\python\\zc.txt','wb')
13      f.write(url.content)
14      f.close()
15      '''
16      imglist = get_img(url.text)
17      d_img(imglist)
18
19
20  def get_img(text):
21
22      mode = re.compile('<img src="(.+?).jpg">')
23      imglist = mode.findall(text)
24      if imglist == []:
25          print("not picture")
26          exit()
27      else:
28          return imglist
```

主要职责
Main Responsibilities

```
19
20  def get_img(text):
21
22      mode = re.compile('<img src="(.+?).jpg">')
23      imglist = mode.findall(text)
24      if imglist == []:
25          print("not picture")
26          exit()
27      else:
28          return imglist
29
30
31  def d_img(imglist):
32      image_name = 0
33      for img_url in imglist:
34          try:
35              img_url = img_url+".jpg"
36              print(img_url)
37              f = open('F:\\python\\zc_' + str(image_name) + ".jpg",'wb')
38              img = requests.get(img_url).content
39              f.write(img)
40              f.close()
41          except Exception as e:
42              print(str(img)+" error",e)
43          image_name += 1
44      print("end! num:%d"%image_name)
45
46
47  if __name__=='__main__':
48      main()
```

主要职责
Main Responsibilities

5

处理上下游客户遇到的问题。

锻炼沟通能力。

主要职责
Main Responsibilities

6

开发添加报文标志位，日志防爆功能。

熟悉了go语言的基本语法，channel原理、并发。

主要职责
Main Responsibilities

i) golang

所有可执行的go程序都必须包含一个main函数，是程序的入口，mian函数包含在mian包中。

- go run：go run 编译并直接运行程序，它会产生一个临时文件（但不会生成 .exe 文件），直接在命令行输出程序执行结果，方便用户调试。
- go build：go build 用于测试编译包，主要检查是否会有编译错误，如果是一个可执行文件的源码（即是 main 包），就会直接生成一个可执行文件。
- go install：go install 的作用有两步：第一步是编译导入的包文件，所有导入的包文件编译完才会编译主程序；第二步是将编译后生成的可执行文件放到 bin 目录下（$GOPATH/bin），编译后的包文件放到 pkg 目录下

```
package main
import "fmt"
func main() {
   /* 哈哈哈哈 */
   fmt.Println("Hello, World!")
}
```

主要职责
Main Responsibilities

```
<proj>
  |--<src>
     |--<a>
        |--<a1>
           |--al.go
        |--<a2>
           |--a2.go
     |--<b>
        |--b1.go
        |--b2.go
     |--<c>
        |--c.go
  |--<pkg>
  |--<bin>
```

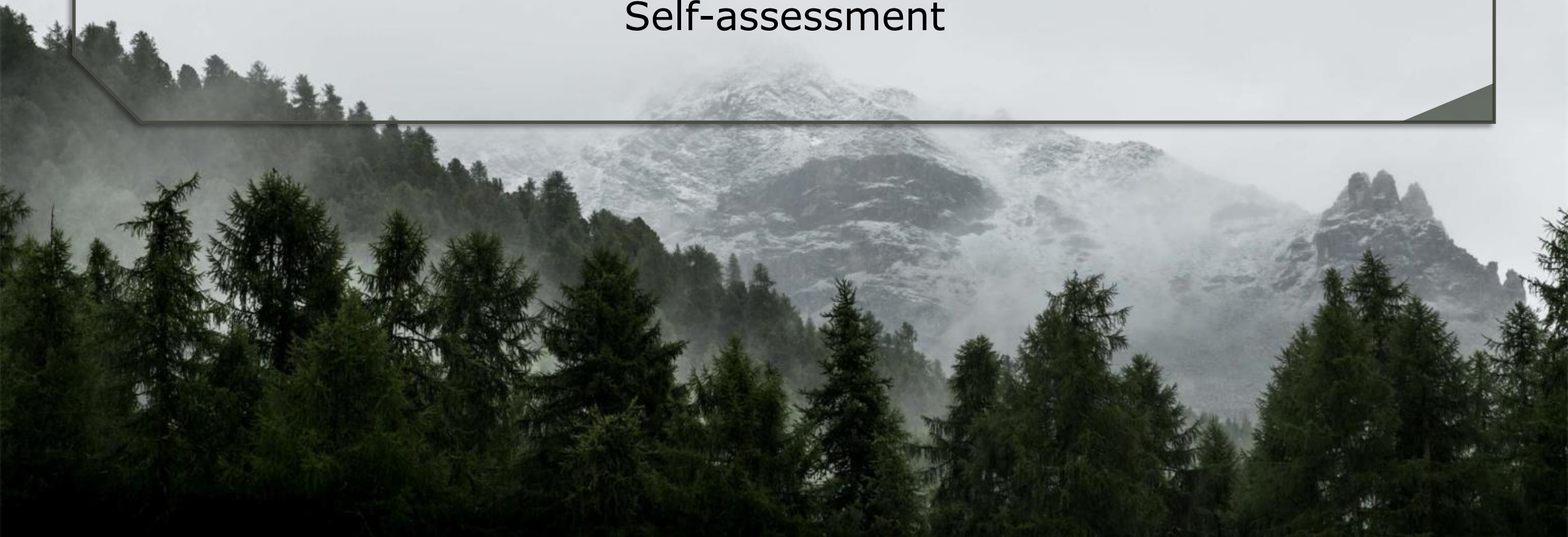
c.go:

```
package main
import(
   "a/a1"
   "a/a2"
   "b"
)
func main(){
   a1.PrintA1()
   a2.PrintA2()
   b.PrintB()
}
```

PART FOUR

自我评价

Self-assessment



自 我 评 价
Self-assessment

- 工作积极上进，力求在保持质量的同时最高效的完成任务；
- 和领导同事相处融洽，一起学习讨论，一起娱乐八卦；
- 乐于学习，工作中遇到的新知识点及时去充电赋能。
- 有责任心，上下游客户遇到的问题尽量当时当天解决，问题太多当天没时间解决的分清紧急程度及时和客户沟通。
- 深度不够，有时光顾着做任务而忽略了技能点的深挖，对有些不太重要的点含糊不清。

PART FOUR

新项目学习

新项目学习

hadoop

HDFS	分布式文件系统
Hbase	分布式列式数据库
MapReduce	编程模型，大规模数据集并行运算
Hive	数据仓库工具
Zookeeper	协同工作系统
Ambari	安装Hadoop
YARN	资源调度
Kafka	分布式发布订阅消息系统

新项目学习

cypher

增

```
create  (p:Person{name:"zhouchao",age:24})
create (a:Person{name:'小明'})-[r:同学]->(b:Person{name:'小强'})
```

删

```
match (p:Person)   where p.name='小强' delete  p
match (p:Person)   where p.name='小强' remove  p.age return p
```

改

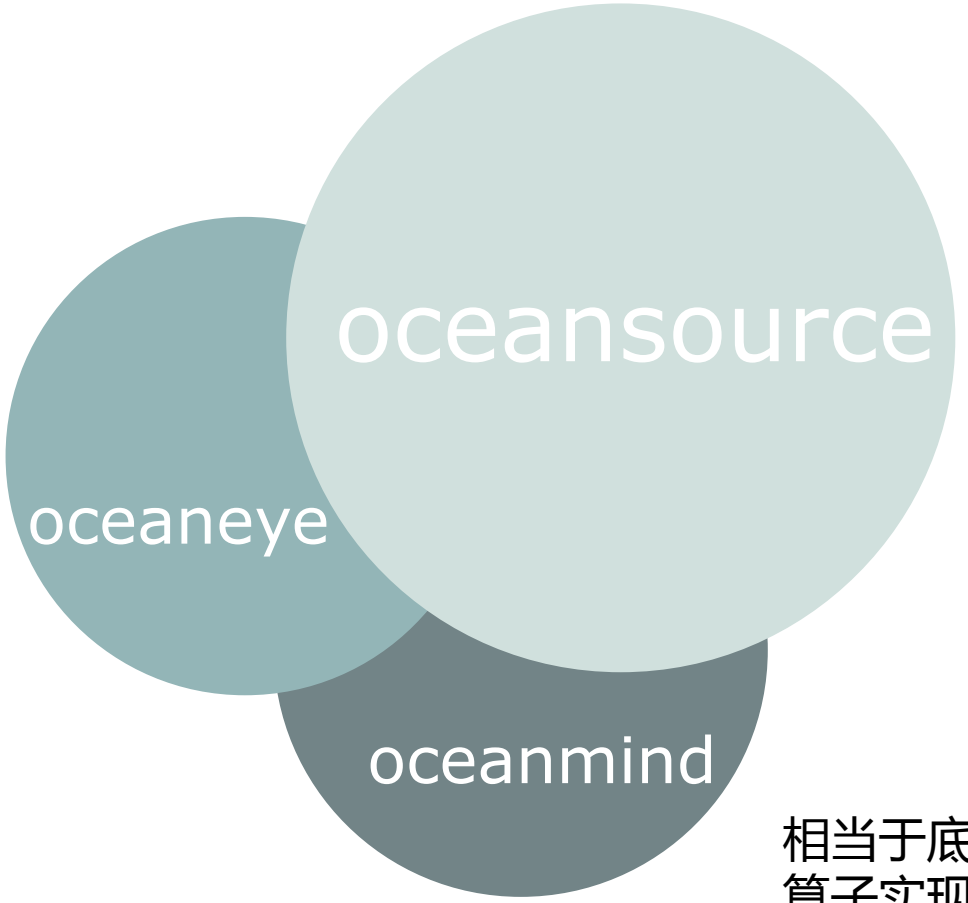
```
match (p:Person)   where p.name='小强' set  p.age = 24 return p
```

查

```
match (p:Person)   where p.name='小强' return p
```

新项目学习

相当于Web的个性定制化平台，可以通过oceanming提供的接口显示数据。

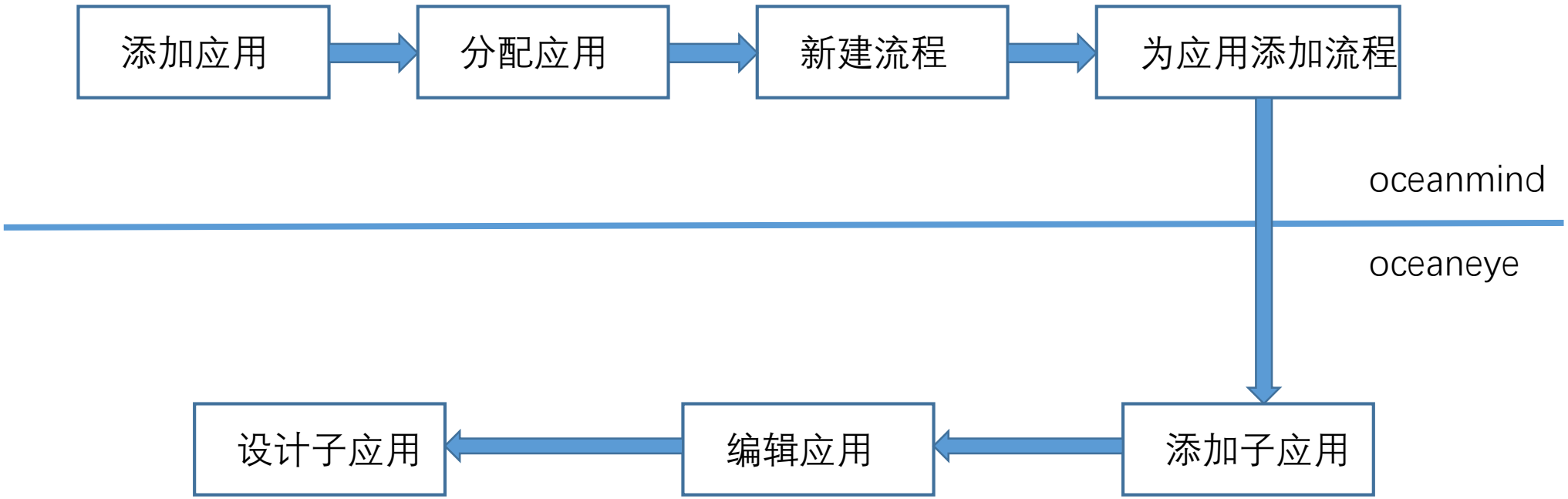


相当于Oceanmind的流程可视化平台，新建库后，分为原始层、标准层、结果层。在它们之间建立流程，实现数据清洗，转换，监控等功能。

相当于底层核心平台，可以通过算子实现各种想要的大数据的功能。可以新建应用，定义设计流程。

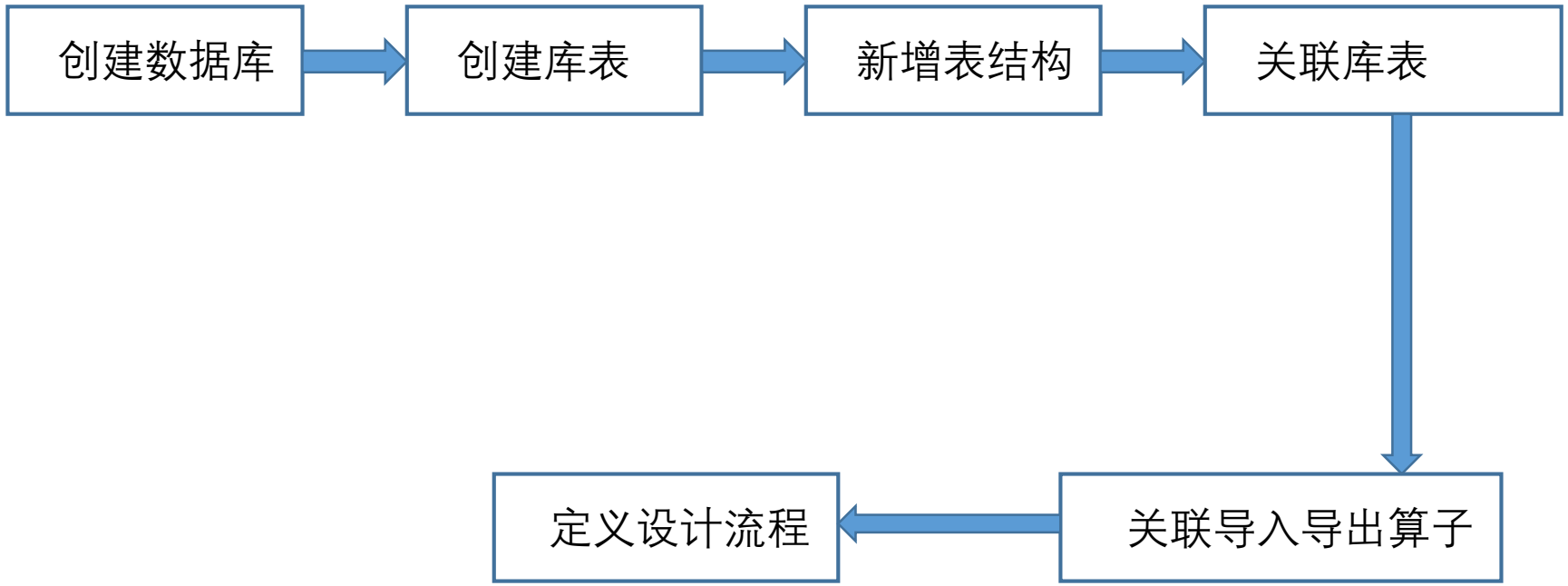
新项目学习

操作流程1


添加应用
分配应用
新建流程
为应用添加流程
oceanmind
oceaneye
添加子应用
编辑应用
设计子应用

新项目学习

操作流程2



新项目学习情况

Json简单架构

```
{
    "properties":{
        id,类型等控件的基本属性，控件所要显示的内容。
    },
    "events":{
            事件:{
                动作:{
                    临时对象:{
                    },
                    临时变量,
                    method,
                    return,
                }
            }
    }
}
```

2019.7

感 谢 您 的 聆 听