大数据平台价值调研

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 版本 | 修订者 | 修订描述 |
| 2018.11.08 | v1 | 林 清 | 初稿 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

[1. 平台安装部署 3](#_Toc529951196)

[1.1 工作内容 3](#_Toc529951197)

[1.2 当前状态 3](#_Toc529951198)

[1.3 业界状态 3](#_Toc529951199)

[1.4 未来发展趋势 3](#_Toc529951200)

[2. 组件管理 4](#_Toc529951201)

[2.1 工作内容 4](#_Toc529951202)

[2.2 当前状态：组件管理复杂： 4](#_Toc529951203)

[2.3 业界状态：组件管理自动化 4](#_Toc529951204)

[2.4 未来发展趋势 5](#_Toc529951205)

[3. 故障恢复 5](#_Toc529951206)

[3.1 工作内容 5](#_Toc529951207)

[3.2 当前状态 5](#_Toc529951208)

[3.3 业界状态 5](#_Toc529951209)

[3.4 未来发展趋势 5](#_Toc529951210)

[4. API服务 5](#_Toc529951211)

[4.1 工作内容 5](#_Toc529951212)

[4.2 当前状态 6](#_Toc529951213)

[4.3 业界状态 6](#_Toc529951214)

[4.4 未来发展趋势 6](#_Toc529951215)

[5. 组件选型 6](#_Toc529951216)

[5.1 工作内容 6](#_Toc529951217)

[5.2 当前状态 6](#_Toc529951218)

[5.3 业界状态 6](#_Toc529951219)

[5.4 未来发展趋势 6](#_Toc529951220)

[6. 结论 7](#_Toc529951221)

# 平台安装部署

## 工作内容

提供快速部署的安装包，如IData项目中的Ansible安装包、IBNS项目中的Helm安装。

## 当前状态

无论IData还是IBNS均做到了一键部署，区别在于：

|  |  |  |
| --- | --- | --- |
|  | IData Ansible安装包 | IBNS Helm安装包 |
| 安装方式 | 一键安装 | 一键安装 |
| 维护成本 | 所有组件的安装逻辑在一个包中，非常复杂，需要专人维护，修改任意一个微小配置都要经过桂亮童鞋（容易造成“单点故障”！！）。 | 每个组件各自维护一个Helm包，组件负责人自己维护！ |
| 安装速度 | 40min | Kubernetes集群部署完成后，秒级部署/卸载 |
| 安装包大小 | 3GB | Helm包非常小，数据存放在镜像中。分发时只需配置Docker Hub地址，无需配置镜像。 |
| 可视化安装 | 自研web服务实现 | 支持 |

## 业界状态

IBNS项目中，我们用Helm包来发布组件。当前容器化的潮流下Ansible脚本安装部署的模式已经被取代了，而在开源社区下Helm基本是一家独大（没有找类似对手），隐隐要成为标准！

Helm当前发展状态如下：

* 目前已经从Kubernetes中分离出来，[成为CNCF的独立项目](https://www.helm.sh/helm-enters-the-cncf/index.html)（今年6月1日）。
* 官方Helm仓库发布了近百个组件的安装包，并且社区非常活跃。官方发布的信息，Helm社区有3,500个贡献者和566个贡献机构。
* 有一个Web UI项目[monocular](https://github.com/helm/monocular) ，可以将Helm仓库打造的像一个应用商店。
* Helm仓库除了可以本地部署以外，还支持在云部署（[Chart Museum](https://github.com/helm/chartmuseum)）。当前支持S3、Oss等几乎所有对象存储。

Helm使用过程中的感受：

* 官方Helm仓库中有状态组件不多。
* 个人开发一个组件Helm包有一定的成本（比ansible 脚本要复杂）。

## 未来发展趋势

官方Helm仓库提供的包越来越多（肯定的！！）。Helm当前已经从Kubernetes项目中独立，后续是否会有可能对Mesos、Swarm等编排工具有所支持？

# 组件管理

## 工作内容

启用HA、变更组网模式、副本策略、配置管理等。

## 当前状态：组件管理复杂

IBNS项目中，我们完成了对Spark、MongoDB、Kafka、ZK的容器化，但是当前只是将服务从裸机移动到了容器中，运维相比原来没有太多优势，Kubernetes的编排能力并有发挥。进行扩容，服务迁移等操作时依然需要手工介入。

造成上述问题的最主要原因是：大部分组件是有状态的，Kubernetes原生接口无法将运维流程描述清晰。为了容器化能够实现我们在设计的时候做了很多妥协，比如，使用主机网络、绑定容器的运行节点等等。这些妥协与Kubernetes的理念是冲突的，造成平台一些功能的发挥受到了限制。

## 业界状态：组件管理自动化

为了解决有状态容器的问题，CoreOs公司提出了[Operator Framework](https://coreos.com/operators/)，核心思想是：“[进行Kubernetes二次开发，为每个有状态组件打造专门API，将运维知识固化到软件！](https://coreos.com/blog/introducing-operators.html)”。

自2016年CoreOS公司提出Operator框架后，相当多的[数据服务公司](https://commons.openshift.org/sig/operators.html)基于这个框架（或者这个理念）实现对应组件在Kubernetes层面的自动化管理！

同样的红帽的[OpenShift平台](https://docs.okd.io/3.11/operators/osdk-getting-started.html)、VMWare的Cloud Foundry，也提供了Operator接口进行二次开发。

当前以下组件能够找到的Operator（不完全统计）：

|  |  |  |
| --- | --- | --- |
| 组件名称 | Operator提供者 | 备注 |
| [MongoDB](https://docs.opsmanager.mongodb.com/current/tutorial/install-k8s-operator/) | Mongo官方 | 集成在企业版中 |
| [Kafka](https://www.confluent.io/confluent-operator/) | Confluent公司 | [Confluent](https://www.confluent.io/confluent-operator/)公司是一家数据平台公司，该公司以Kafka为核心组件打造数据平台。目前Kafka Operator是开源的 |
| [Redis](https://redislabs.com/blog/redis-enterprise-operator-kubernetes/) | RedisLab | Redis企业版的功能，但是也找到了[开源方案](https://github.com/spotahome/redis-operator)。 |
| [Mysql](https://github.com/oracle/mysql-operator) | Oracle官方 | 开源，似乎是个实验版本 |
| [PostgreSQL](https://github.com/CrunchyData/postgres-operator) | CrunchyData公司 | CrunchyData专门提供PostgreSQL的商业解决方案 |
| [Cassandra](https://github.com/instaclustr/cassandra-operator) | Instaclustr | Instaclustr也是一家大数据公司el |
| [Spark](https://github.com/GoogleCloudPlatform/spark-on-k8s-operator) | GoogleCloudPlatform |  |

## 未来发展趋势

[Operator Framework](https://coreos.com/operators/)将运维的工作编码到K8S的API接口中，实际上是一种DevOps的落地方式，将一些运维动作自动化。

当前AI大热，DevOps进化出了AiOps的新概念，AIOps的概念更广泛，包括：常规运维、服务监控、故障发现等范围。

# 故障诊断、恢复

## 工作内容

解决组件在使用过程中的故障，恢复组件服务。

## 当前状态

组件发生故障的原因：绝大部分是网路、存储、使用方式等问题。组件代码Bug造成故障的场景非常少。

为了减少故障、或者及早发现问题。WIS项目进行了：指标监控（Influx+ Granfana）、异常告警（微信告警）、日志收集（ELK），使故障发生后运维能够快速介入。

当前无法做到自动恢复故障（这里指的是一些突发故障）！

## 业界状态

在传统IT工程中，故障诊断、恢复、监控等概念涵盖在APM系统中。APM称为应用性能管理（Application Performance Management），是通过**即时监控以实现对应用程序性能管理和故障管理的系统化解决方案**。可以在网上百度到大量提供APM解决方案的公司，如[Elastic APM](https://www.elastic.co/cn/solutions/apm)。

当前AIOps兴起，APM的概念被涵盖在其中，主要包括以下工作内容：

* 指标检测：取代传统静态阈值，实现自动学习阈值、自动调参。
* 文本检测：日志分析，当前许多云服务商都提供日志分析服务。
* 调用链监控：监控系统的整个接口调用过程，当前网络上有了比较多的调用链监控工具，如[OpenZipkin](https://zipkin.io/)、[SkyWalking](https://github.com/apache/incubator-skywalking)、[Pinpoint](https://naver.github.io/pinpoint/)、[Dapper](http://bigbully.github.io/Dapper-translation/)。
* 故障预测：主要指硬件的故障预测如磁盘、网络、内存泄露等。 ------ 网络上能够找到很多关于磁盘故障预测，以及基于流量识别DDos攻击的案例。

## 未来发展趋势

见AIOps。

# API服务

## 工作内容

* 为新组件提供基础API的使用说明，通过PPT文档、说明文档等方式输出。
* 当业务有代码优化需求时，提供相关建议、深度使用API的一些技巧。

## 当前状态

API服务实际上被当做调研工作的一部分，主要提供的简单的Demo + 组件的基本概念，并且这一部分内容在网络上是可以比较容易获取的。

## 业界状态

实际上公司目前大部分大数据业务的模式是：“采集 🡪 处理 🡪 存储 🡪 展示”的架构，每个步骤串行运。

当前一些平台产品提供拖拽式开发的功能，将数据加载、数据保存、数据处理等功能封装成独立的功能模块，用户通过拖拽模块的方式构建业务，完全不需要了解底层平台的架构、API。

此类平台包括：

* Kettle：拖拽式的ETL平台。
* [FusionInsight Universe](https://developer.huawei.com/cn/ict/Products/BigData)：华为的数据分析平台，其中数据采集、数据清洗、ETL、数据挖掘、报表等模块均提供了拖拽编程的的功能。

上述平台的主要用户包括：业务外包、数据分析师等非编码人员。**公司内部大部分数据清洗、数据挖掘的逻辑均是以编码方式实现的，此类平台暂时似乎没有强需求**。

## 未来发展趋势

当前可视化编程的功能主要应用在报表类软件上，使用用户可以快速生成各种样式的报表。专门为数据处理开发拖拽编程的功能，并非当前开源社区的趋势，而且短时间内不会成为发展的主流方向。

# 组件选型

## 工作内容

结合业务场景给出合理的架构方案，为业务开发组提供一些建议。

## 当前状态

需要选型的主要是数据库类组件。大数据领域的数据库/存储组件非常多，在不同的应用场景差别非常大。

组件选型的相关工作实际上不多，但在一些场景下可能会遇到一些困难，如：

* 对业务场景、数据规模等信息掌握不充分；
* 对一些组件的认识并不深入、没有实践经验，造成一些选择上的错误；

## 业界状态

[DB-ENGINES](https://db-engines.com/en/ranking)会更新DB相关的热度排名，这些排名信息有助于用户进行选型。

## 未来发展趋势

略

# 结论

在**安装部署**、**API服务**、**组件选型**三方面没有找到有价值的、能够形成竞争力的课题。其中，**安装部署**工作在不久的将来可能80%的工作会被自动化。**API服务**、**组件选型**这两部分工作很难被自动化，只要我们的技术储备领先于业务组，那么我们的工作对业务来说依然有价值。

在**组件管理**方面，开源社区已经有比较好的方案，但是方案实施门槛比较高。未来不排除需要IBNS引入这些架构时，需要我们提供支持：

* **DevOps实践**：通过Operators框架完善组件的容器化程度，提升其运维管理的自动化程度，并积累DevOps经验 —— IBNS项目未来可能会有这个需求；

在**故障恢复**方面，目前是业界的难点，开源社区很难找到很好解决方案，并且该课题涉及内容很广，包括：软硬件、监控、诊断、决策、建立知识库等。目前，业界的期望是在自动化运维的基础上，引入AI实现智能运维。

* **AIOps实践**：LOG日志挖掘、基于AI的系统故障诊断、磁盘故障预测、问答机器人—— 是当前的发展方向，并可能在之后公司的其他项目中有需要；