中国科学院软件研究所

面向大数据的多源异构云计算平台
详细设计说明书

文档状态	[ ] 初稿 [ ] 评审通过 [ ] 修改 [ ] 发布 [ ] 作废	**作者**	网驰项目组
		部门	
		完成日期	

文档信息：

文档名称	架构设计说明书	文档编号	
文档存放			
发行者	中国科学院软件研究所	发行日期	2016-6-1

文档变更记录：

修订日期	修订说明	版本号	修订人	审核人
20160601	面向大数据的多源异构云计算平台	V0.1	吴恒	吴恒

目 录

1. 前言

1.1. 项目背景

云计算是创新 2.0 下的信息化系统发展的新业态，是知识社会创新 2.0 推动下的经济社会发展新形态，已成为促进信息技术产业发展和应用创新的主要推手之一，具有重要的国家战略意义。2015 年 1 月，国务院发布了《关于促进云计算创新发展培育信息产业新业态的意见》，国家明确云计算技术接近国际先进水平，构建自主标准体系目标；利用公共云计算服务资源开展百项互联网和大数据应用示范工程；引导传统企业的转型升级，鼓励积极拓展国际市场。

当前，云计算产业主要采用“以数据中心为基础、各自运营为手段、巨资竞争为策略”的实践模式，核心是集中式资源在线服务，通过资源规模化运营、广域共享和公用化服务实现 IT 总体实施成本降低和利用率提升，技术特征主要包括资源虚拟化、分配动态化、服务自动化等。例如 Amazon EC2、Microsoft Azure、Google Cloud Platform、阿里云、UCloud、金山云等。 随着以大数据为基础的智能计算技术深入发展，应用场景更加复杂，云计算强边界已经制约云应用的持续创新。为此，以多源异构云协作为基本特征、面向大数据的新型云计算模式逐渐成为学术界关注的热点和产业界关注的重点。

本项目为面向大数据的多源异构云计算平台，旨在通过多方云资源深度融合，以“软件定义”方式按需管理多源异构云；旨在通过软件优化、硬件加速等手段，满足大数据高效运行，便捷易用的目标。

1.2. 参考文献

1.http://www.cisco.com/c/en/us/products/cloud-systems-management/intercloud-fabric/index.html
2. http://dl.acm.org/citation.cfm?id=2904111&CFID=793855473&CFTOKEN=73457219

1.3. 术语与缩写

缩写、术语	解 释
SDN	软件定义网络，Software Defined Network
SDS	软件定义存储，Software Defined Storage
NFV	网络功能虚拟化、Network Function Virtualization
CI	持续集成、Continuous Integration
AC	应用编排，Application Compose

2. 需求分析

随着 IT 技术的快速发展和应用深入，信息化系统呈现出交付移动化、访问全域化、峰值极限化、增值精细化发展趋势，迫切需要企业 IT 架构的转型升级。企业 IT 架构是指机房硬件管理及信息系统开发交付模式，涉及信息系统如何运维，如何交付，如何开发三大核心问题。云计算是解决 IT 架构转型的主要技术途径，目前主要关注运维和交付问题，其中，虚拟机技术主要解决企业 IT 架构的高效能运维转型，核心是便捷、提高资源利用；容器技术主要解决企业 IT 架构的互联网交付转型，核心是敏捷、开发运维一体化。未来，以大数据为核心、具备按需分析、精准增值的应用将成为主流，如何应对大数据多源异构，协同复用、高效运行等特征，实现 IT 架构转型达到简化开发的目的面临挑战。

本项目基于云计算已有成果，通过整合多源异构云计算平台满足大数据多源异构的采集需要；通过引入数据湖满足大数据场景协同分析需求（http://www.vldb.org/pvldb/vol8/p1916-bhardwaj.pdf）；通过软件优化和硬件加速手段，实现云平台高效运行目标。具体而言，本项目主要解决企业 IT 架构的大数据开发转型，核心是快捷、简化开发复杂度。如图 1 所示，本项目是在整合虚拟机、容器平台基础上，重点支持数据湖场景。

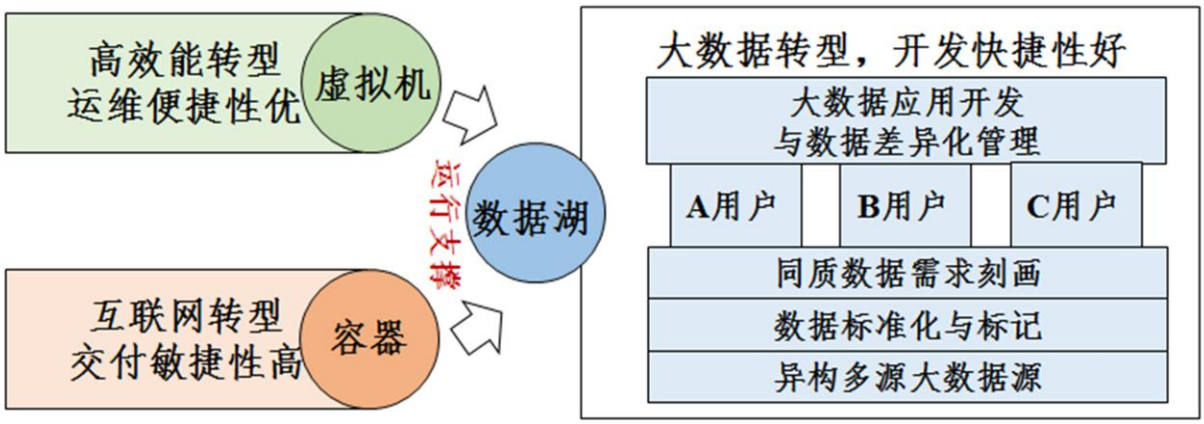


图 1 项目目标

数据湖场景是指不同人员以相同数据作为输入，差异化进行分析处理，得出个性化结果的过程。例如，某商业银行具有大量涉及消费的数据，通过对其标准化和分类，提供按需大数据供给服务。不同业务部门可能关心相同数据集(消费记录)，根据不同目标定制数据分析处理，得出消费年纪分布、或消费商品分布等多维结果。为应对该需求，开发通常采用数据拷贝，再验证

的方式，一般低效耗时。因此，迫切需要数据湖服务化和数据优化管理技术，如图 2 所示。

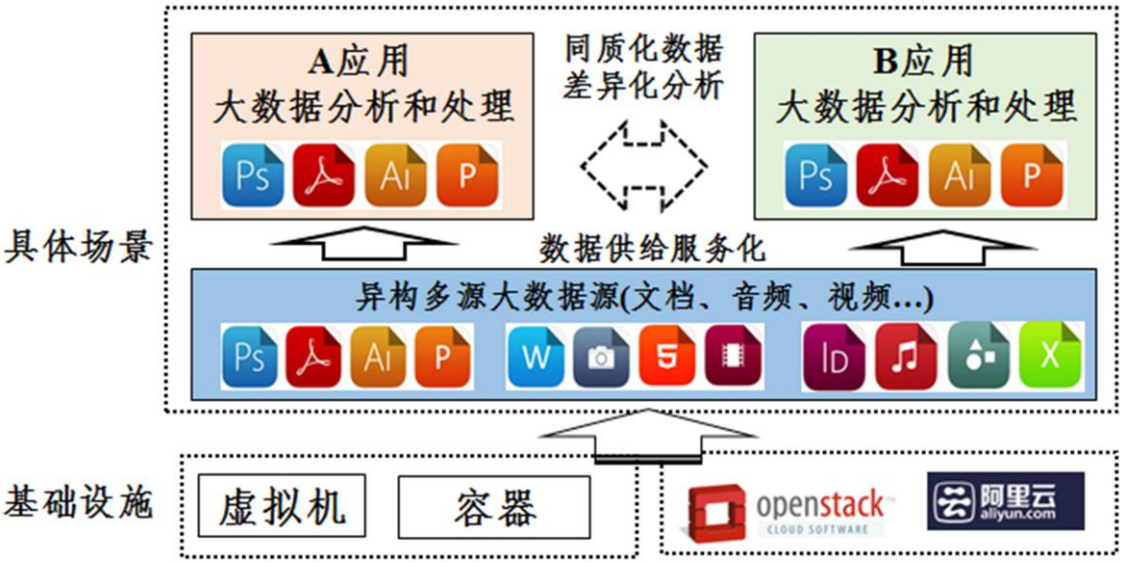


图 2 场景描述

3. 设计目标

3.1. 重要设计

1. 系统层面：集成多源异构云平台管理设计，至少可管理 Xen 和 Docker 两种异构虚拟化技术、至少具备 Aliyun、OpenStack 两种管理能力，提供服务化接口
2. 系统层面：集成配置中心设计，具备配置收集、变更通知、可靠运行等能力，提供配置组织模型、变更推送机制文档
3. 系统层面：研发物理机管理设计、具备生命周期管理、信息收集、核心软件安装和抽象能力，提供服务化接口
4. 系统层面：适配虚拟机/容器生命周期设计，提供服务化接口
5. 系统层面：基于虚拟机，实现 SDN、SDS、NFV 高级功能，提供服务化接口
6. 系统层面：基于容器，实现 CI、AC 高级功能，提供服务化接口
7. 研究层面：细粒度分布式应用追踪技术
8. 研究层面：自适应云应用性能预测技术
9. 研究层面：低开销性能干扰评估技术
10. 研究层面：多目标资源优化调度技术

11. 研究层面：高性能硬件加速技术
12. 研究层面：虚拟化操作系统构造技术
13. 研究层面：高可靠数据备份加强技术
14. 研究层面：数据协同管理技术

3.2. 设计原则

1. 多源异构云平台采用模块化设计
2. 服务化接口采用高安全设计
3. 设计异常和容错处理机制，参考异常处理文档

4. 架构设计

4.1. 总体架构

整体架构如图 3 所示，以服务注册与配置管理中心为核心，涉及物理机管理与软件安装、虚拟机管理与迁移优化、容器管理与安全加强、网络虚拟化及功能虚拟化、持续继与应用编排、权限管理与资源管控、界面框架与组装等模块。

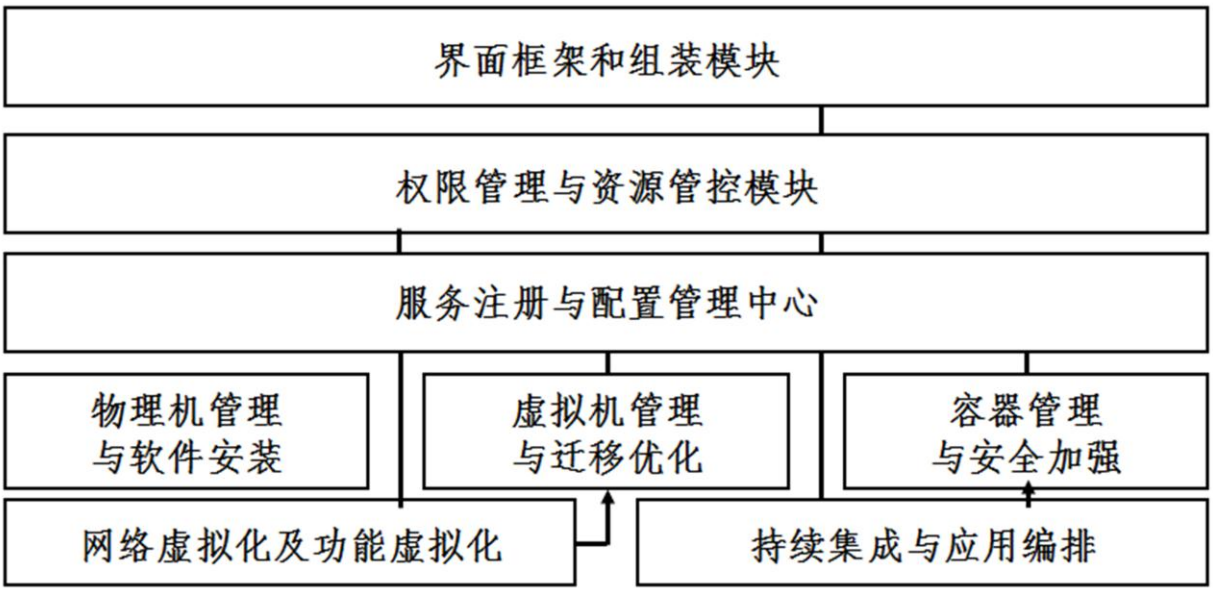


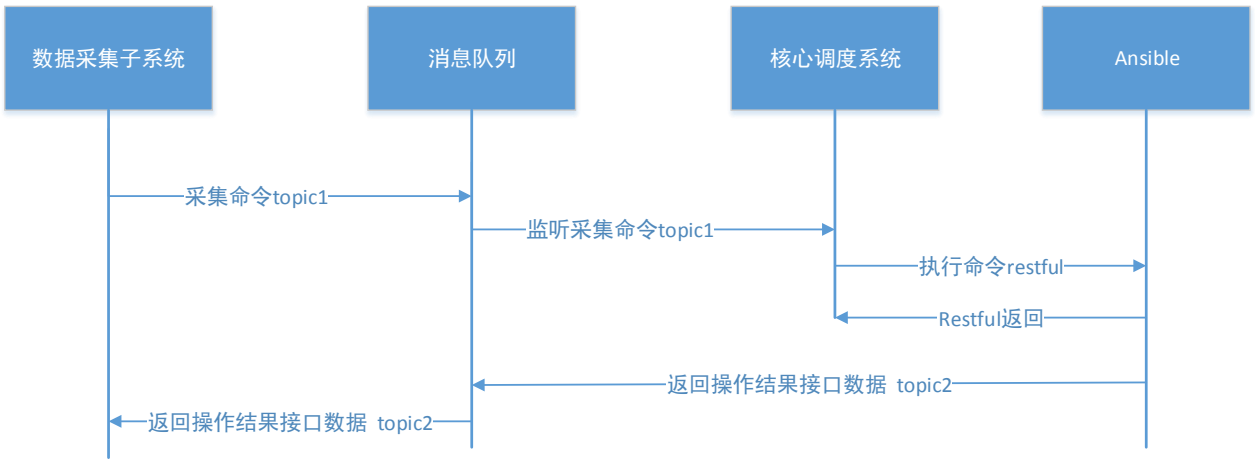
图 3 总体架构

- 服务注册与配置管理中心：用于统一记录其它模块的配置信息，维护配置变更及影响模块，保证其它各模块配置信息的一致性；

- 物理机管理与软件安装：具备物理机安装虚拟机/容器、配置网络、存储的能力，研发集群管理技术；
- 虚拟机管理与迁移优化：支持虚拟机生命周期管理，具备内存和硬盘在线迁移能力；
- 容器管理与应用加强：支持容器生命周期管理，强化其安全不足；
- 网络虚拟化与功能虚拟化：该模块有效地支持 vxlan、VPN 等能力；
- 持续集成与应用编排：具备应用从源码到容器的自动化生成，具备可视化应用组件刻画和一键部署能力；
- 界面框架与组装：所有模块的可视化操作。

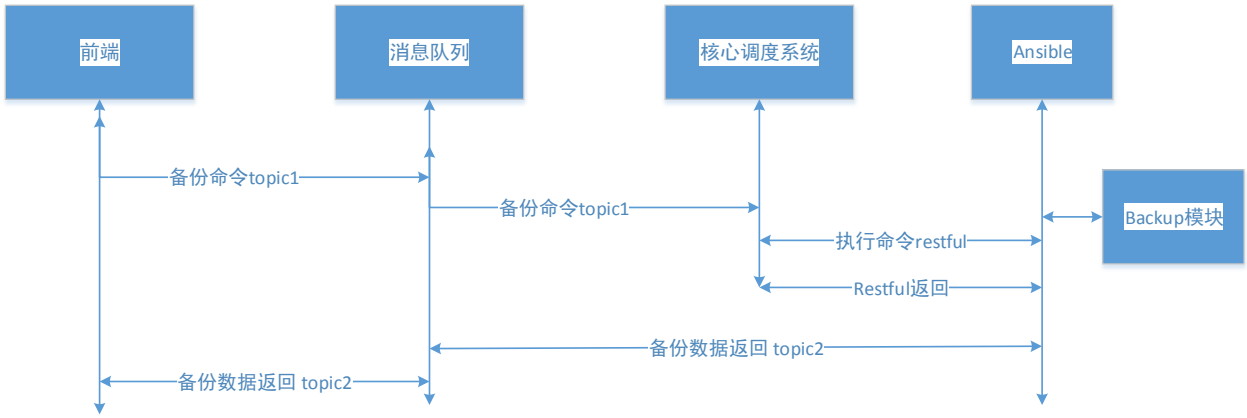
4.1.1. 服务注册与配置管理中心

数据采集子系统通过将物理机配置信息采集参数通过 topic 的方式将数据写入到消息子系统，核心调度子系统监听到数据采集模块的信息然后通过核心调度算法将数据分发到 Ansible+restful 接口模块，操作数据通过 ansible 的 result 模块返回数据给消息队列。



4.1.2. 数据备份模块设计

前端界面下发备份 topic 命令给消息队列，消息队列将消息发送给核心调度子系统，通过给 ansible 添加 backup 模块来实现数据备份和拷贝到远程 sftp 服务器上。



4.2. 设计策略

1. 备份调度策略： 备份数据在备份的时候，如果是没有网络可能会造成拥塞，所以需要对数据备份进行分割备份。 备份数据极限值 backup_max,一次备份一台备份服务器最大的数值，如果超过这个值就分配给定时器，过 5 分钟再进行备份。
2. 调度测试设计：假设 Ansible Slave 节点操作服务器的容量为 N。从接口获取执行 Ansible 命令的服务器数量 NC，计算需要 Ansible Slave 节点的数量 AH=NC/N,然后选取可用的 AH 台 Ansible Slave 节点，将任务发送 Ansible Slave 节点上执行。每一次选择 Ansible Slave 节点的时候优先考虑上次没有选择过的节点。
3. Ansible slave 节点高可用策略：采用 ha+keepalive 来保证节点节点的可以高可用性，这样做的原因是因为 Ansible 向外面提供的 restful 接口，restful 接口保证有效运行的情况最好是用高可用的方案
4. 多任务处理：ansible 本身可用支持多任务同时处理，在包装了 restful 接口的时候，resftul 接口本身是多线程运行的，即下发一个命令就运行一个线程。

4.2.1. 数据安全性设计

在通信的过程中，数据都通过 base64 加密。提高通信的安全性。

4.2.2. 扩展性设计

可以在数据库中添加 ansible-slave 节点来增加核心调度子系统的容量。每次增加一个 slave 节点就可以增加 400 个节点的容量

4.2.3. 性能设计

【如果有性能设计考虑，这里补充性能的设计思路

如果没有，标注“无”并说明理由】

4.2.4. 运行安全设计

【如果运行安全方面的设计考虑，这里运行安全、系统容灾和系统容错的设计思路
如果没有，标注“无”并说明理由】

4.2.5. 容量设计

Ansible slave 节点容量是 400 个 host 节点，核心调度层的容量和这个相关

4.2.6. 可维护性设计

【如果有容量设计考虑，这里补充容量的设计思路
如果没有，标注“无”并说明理由】

5. 功能大类列表

5.1. 物理机管理与软件安装

- 支持通过 IPMI 管理物理机
- 提供 rpm 源和配置管理
- 安装和配置 Xen、Docker
- 网络配置，支持 OVS，可配置成 vxlan 和 dpdk 两种
- 存储配置，支持本地存储、OCFS2 和 Ceph 三种
- 获取物理机信息，包括系统信息，进程信息、服务等
- 监测物理机资源，包括 CPU、内存、网络和存储

5.2. 虚拟机生命周期管理与迁移优化

- 通过 ISO 安装虚拟机
- 虚拟机创建、启动、关机、删除，支持通过拷贝和模板两种策略创建虚拟机
- 虚拟机转变成模板、模板转换成虚拟机
- 虚拟机挂起、恢复
- 虚拟机内存快照和硬盘快照

- 虚拟机内存在线迁移和硬盘在线迁移
- 虚拟机登陆密码的定制化
- 虚拟机安全策略的个性化配置
- 虚拟机 CPU、内存、网卡和硬盘的在线变更
- 支持 VNC、Spice 两种协议交付访问虚拟机
- 虚拟机的集群管理
- 虚拟机的高可用管理

5.3. 容器生命周期管理与安全加强

- 容器仓库的安全配置管理
- 容器镜像的导入和导出
- 通过 Dockerfile 生成容器镜像
- 镜像的安全上传与下载
- 镜像的创建、停止和删除
- 容器的网络交付访问
- 容器 CPU、内存资源的在线调整
- 容器集群管理
- 容器高可用管理

5.4. 网络虚拟化和网络功能虚拟化

- 支持 vxlan 的广播机制
- 支持 dpdk 加速机制
- 支持虚拟防火墙管理
- 支持虚拟路由
- 支持虚拟 VPN
- 支持内存模拟硬盘
- 支持 SSD 加速

5.5. 持续集成与应用编排

- 支持 git
- 支持 svn
- 支持持续集成工具
- 支持 Compose
- 支持 Nginx、Mysql、Redis、Sprak、HDFS 的容器化

5.6. 权限管理和资源管控

- 支持角色访问控制，控制用户能执行的操作
- 支持流程管理，满足个性化流程需求
- 支持资源隔离，控制用户可使用的资源

6. 外部接口需求

6.1. 硬件接口

【（1）系统是否存在需要调用外部设备 。】

【（2）如果没有需要注明“无”。】

6.2. 软件接口

【（1）与其它系统是否存在接口，具体描述。】

【（2）如果没有需要注明“无”。】

6.3. 其他接口

【（1）与其它系统是否存在接口，具体描述。】

【（2）如果没有需要注明“无”。】