计算平台事业部 COMPUTING PLATFORM

阿里云开发者社区 ALIBABA CLOUD DEVELOPER COMMUNITY

“后红海”时代
独家揭秘当下大数据体系

分析 大数据体系4大热点

详解 9大领域架构分布

预测 未来演进4大技术趋势

概述 有待探索3大疑问

藏经阁

阿里云开发者电子书系列

钉钉扫码加入

飞天大数据平台交流群

钉钉扫一扫加入

DataWorks 产品交流群

钉钉扫一扫加入

MaxCompute 产品交流群

扫一扫加入 MaxCompute

交互式分析（Hologres）交流群

钉钉扫一扫加入

实时计算 Flink 产品交流群

钉钉扫一扫加入

E-MapReduce 产品交流群

钉钉扫一扫加入

机器学习 PAI 产品交流群

钉钉扫一扫加入

搜索推荐技术交流群

阿里云开发者“藏经阁”

海量电子书免费领

目录

00 编者按

任何一种技术都会经历从阳春白雪到下里巴人的过程，就像我们对计算机的理解从“戴着鞋套才能进的机房”变成了随处可见的智能手机。在前面 20 年中，大数据技术也经历了这样的过程，从曾经高高在上的 “火箭科技（rocket science）”，成为了人人普惠的技术。

回首来看，大数据发展初期涌现了非常多开源和自研系统，并在同一个领域展开了相当长的一段“红海”竞争期，例如 Yarn VS Mesos、Hive VS Spark、Flink VS SparkStreaming VS Apex、Impala VS Presto VS Clickhouse 等等。经历激烈竞争和淘汰后，胜出的产品逐渐规模化，并开始占领市场和开发者。

事实上，近几年，大数据领域已经没有再诞生新的明星开源引擎（Clickhouse@2016 年开源，PyTorch@2018 年开源），以 Apache Mesos 等项目停止维护为代表，大数据领域进入**“后红海”**时代：技术开始逐步收敛，进入技术普惠和业务大规模应用的阶段。

阿里云计算平台事业部 著

01 当下的大数据体系的热点

BigData 概念在上世纪 90 年代被提出，随 Google 的 3 篇经典论文（GFS，BigTable，MapReduce）奠基，已经发展了将近 20 年。这 20 年中，诞生了包括 Google 大数据体系，微软 Cosmos 体系，阿里云的飞天系统，开源 Hadoop 体系等优秀的系统。这些系统一步步推动业界进入“数字化”和之后的“AI 化”的时代。

海量的数据以及其蕴含的价值，吸引了大量投入，极大的推动大数据领域技术。云（Cloud）的兴起又使得大数据技术对于中小企业唾手可得。可以说，大数据技术发展正当时。笔者有幸在微软（互联网/Azure 云事业群）和阿里巴巴（阿里云）经历了大数据发展 20 年过程中的后 15 年。受到编辑的邀请，并给了一个很大的题目来讨论“当下的大数据体系和发展”。笔者目前负责阿里巴巴和阿里云大数据领域的部分工作，也接触很多内部和外部的客户/业务。本文试从系统架构的角度，就**大数据架构热点，每条技术线的发展脉络，以及技术趋势和未解问题**等方面做一概述。

目前，大数据领域仍然处于发展期，部分技术收敛，但新方向和新领域层出不穷。本文内容和个人经历相关，是个人的视角，难免有缺失或者偏颇，同时限于篇幅，也很难全面。仅作抛砖引玉，希望和同业共同探讨。

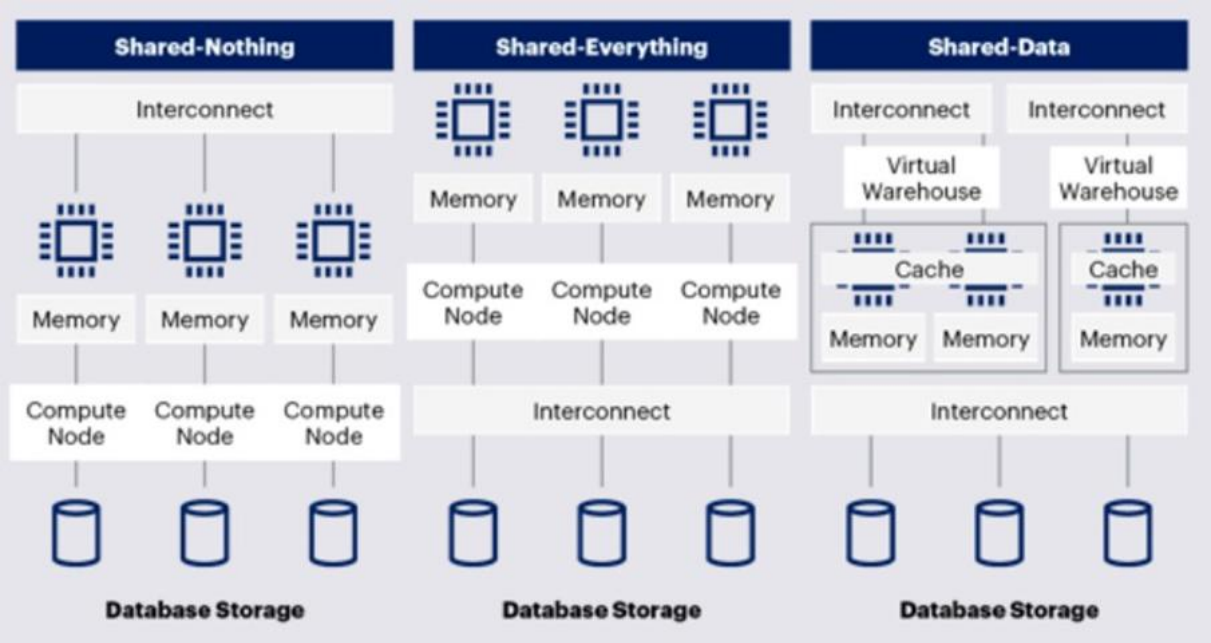
从体系架构的角度看，“Shared-Everything”架构演进、湖仓技术的一体化融合、云原生带来的基础设计升级、以及更好的 AI 支持，是当下平台技术的四个热点。

1.1 系统架构角度，平台整体向 Shared-Everything 架构演进

泛数据领域的系统架构，从传统数据库的 Scale-up 向大数据的 Scale-out 发展。从分布式系统的角度，整体架构可以按照 Shared-Nothing（也称 MPP），Shared-Data，Shared-Everything 三种架构。

大数据平台的数仓体系最初由数据库发展而来，Shared-Nothing（也称 MPP）架构在很长一段时间成为主流。随云原生能力增强，Snowflake 为代表的 Shared-Data 逐渐发展起来。而基于 DFS 和 MapReduce 原理的大数据体系，设计之初就是 Shared-Everything 架构。

Shared-Everything 架构代表是 GoogleBigQuery 和阿里云 MaxCompute。从架构角度，Shared-Everything 架构具备更好的灵活性和潜力，会是未来发展的方向。



（图：三种大数据体系架构）

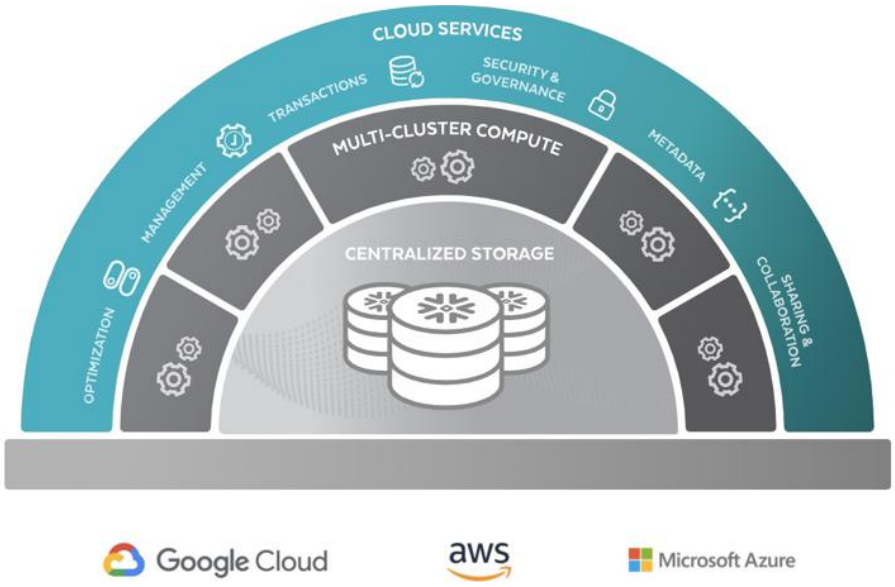
1.2 数据管理角度，数据湖与数据仓库融合，形成湖仓一体

数据仓库的高性能与管理能力，与数据湖的灵活性，仓和湖的两套体系在相互借鉴与融合。在 2020 年各个厂商分别提出湖仓一体架构，成为当下架构演进最热的趋势。但湖仓一体架构有多种形态，不同形态尚在演进和争论中。

(图：数据湖与数据仓库借鉴融合)

1.3 云架构角度，云原生与托管化成为主流

随着大数据平台技术进入深水区，用户也开始分流，越来越多的中小用户不再自研或自建数据平台，开始拥抱全托管型（通常也是云原生）的数据产品。Snowflake 作为这一领域的典型产品，得到普遍认可。面向未来，后续仅会有少量超大规模头部公司采用自建（开源+改进）的模式。



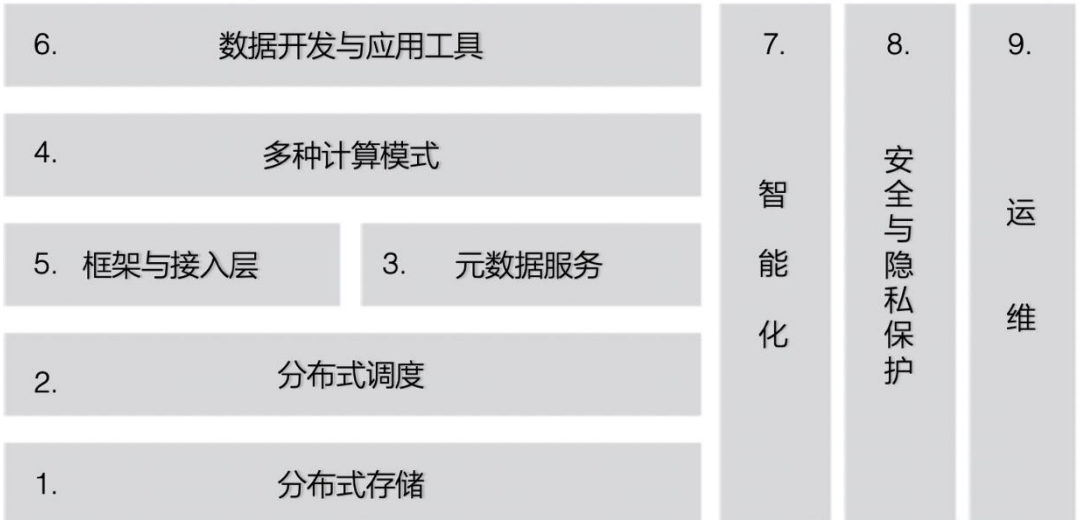
(图：snowflake 的云原生架构)

1.4 计算模式角度，AI 逐渐成为主流，形成 BI+AI 双模式

BI 作为统计分析类计算，主要是面向过去的总结；AI 类计算则具备越来越好的预测未来的能力。在过去五年中，算法类的负载从不到数据中心总容量的 5%，提升到 30%。AI 已经成为大数据领域的一等公民。

02 大数据体系的领域架构

在前文(#1.1)介绍的 Shared-Nothing、Shared-Data、Shared-Everything 三种架构中，笔者经历过的两套体系（微软 Cosmos/Scope 体系，和阿里云 MaxCompute）均为 Shared-Everything 架构，因此笔者主要从 Shared-Everything 架构角度，将大数据领域分成 6 个叠加的子领域、3 个横向领域，共 9 个领域，具体如下图。



(图：基于 Shared-Everything 大数据体系下的领域架构)

经过多年的发展，每个领域都有一定的进展和沉淀，下面各个章节将概述每个子领域的演进历史、背后驱动力以及发展方向。

2.1 分布式存储向多层智能化演进

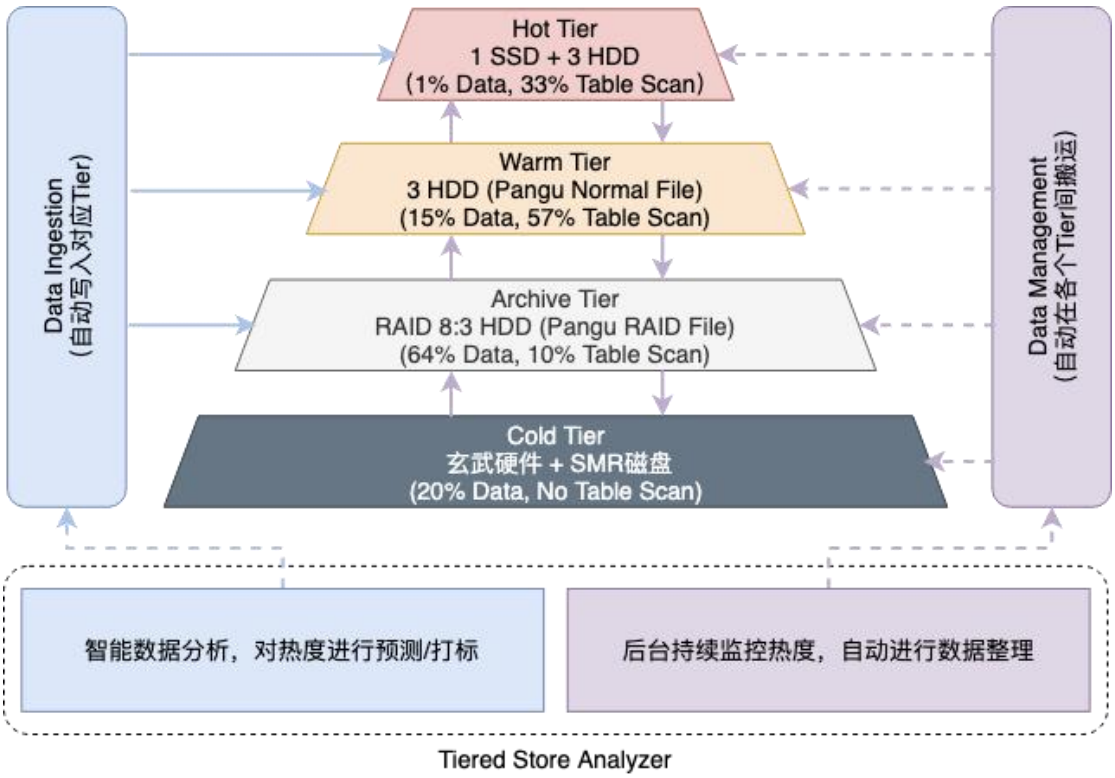
分布式存储，本文特指通用大数据海量分布式存储，是个典型的带状态（Stateful）分布式系统，高吞吐、低成本、容灾、高可用是核心优化方向。**（注：下述分代，仅仅为了阐述方便，不代表严格的架构演进。）**

第一代，分布式存储的典型代表是谷歌的 GFS 和 Apache Hadoop 的 HDFS，均为支持多备份的 Append-only 文件系统。因 HDFS 早期 NameNode 在扩展性和容灾方面的短板不能充分满足用户对数据高可用的要求，很多大型公司都有自研的存储系统，如微软的 Cosmos（后来演进成 Azure Blob Storage），以及阿里巴巴的 Pangu 系统。HDFS 作为开源存储的奠基，其接口成为事实标准，同时 HDFS 又具备支持其他系统作为背后存储系统的插件化能力。

第二代，基于上述底盘，随海量对象存储需求激增（例如海量的照片），通用的 Append-only 文件系统之上，封装一层支持海量小对象的元数据服务层，**形成对象存储（Object-based Storage）**，典型的代表包括 AWS S3，阿里云 OSS。值得一提的是，S3 与 OSS 均可作为标准插件，成为 HDFS 的实时存储后端。

第三代，以数据湖为代表。随云计算技术的发展，以及（2015 年之后）网络技术的进步，**存储计算一体的架构逐渐被云原生存储（存储托管化）+ 存储计算分离的新架构取代。**这也是数据湖体系的起点。同时因存储计算分离带来的带宽性能问题并未完全解决，在这个细分领域诞生了 Alluxio 等缓存服务。

第四代，也是当下的趋势，随存储云托管化，底层实现对用户透明，因此存储系统有机会向更复杂的设计方向发展，从而开始**向多层一体化存储系统演进。**由单一的基于 SATA 磁盘的系统，向 Mem/SSD+SATA（3X 备份)+SATA（1.375X 为代表的 EC 备份)+冰存储（典型代表 AWS Glacier）等多层系统演进。如何智能/透明的将数据存储分层，找到成本与性能的 Trade-off，是多层存储系统的关键挑战。这领域起步不久，开源领域没有显著好的产品，最好的水平由几个大厂的自研数仓存储系统引领。



(图：阿里巴巴 MaxCompute 的多层一体化存储体系)

在上述系统之上，有一层文件存储格式层（File Format layer），与存储系统本身正交。

存储格式第一代，包含文件格式、压缩和编码技术以及 Index 支持等。目前主流两类的存储格式是 Apache Parquet 和 Apache ORC，分别来自 Spark 和 Hive 生态。**两者均为适应大数据的列式存储格式**，ORC 在压缩编码上有特长，Parquet 在半结构支持上更优。此外另有一种内存格式 Apache Arrow，设计体系也属于 format，但主要为内存交换优化。

存储格式第二代，以 Apache Hudi/Delta Lake 为代表的近实时化存储格式。存储格式早期，是大文件列存储模式，面向吞吐率优化（而非 latency）。随实时化的趋势，上述主流的两个存储模式均向支持实时化演进，Databricks 推出了 Delta Lake，支持 Apache Spark 进行近实时的数据 ACID 操作；Uber 推出了 Apache Hudi，支持近实时的数据 Upsert 能力。尽管二者在细节处理上稍有不同（例如 Merge on Read or Write），

但整体方式都是通过支持增量文件的方式，将数据更新的周期降低到更短（避免传统 Parquet/ORC 上的针对更新的无差别 FullMerge 操作），进而实现近实时化存储。因为近实时方向，通常涉及更频繁的文件 Merge 以及细粒度元数据支持，接口也更复杂，Delta/Hudi 均不是单纯的 format、而是一套服务。存储格式再向实时更新支持方向演进，会与实时索引结合，不再单单作为文件存储格式，而是与内存结构融合形成整体方案。主流的是实时更新实现是基于 LogStructuredMergeTree（几乎所有的实时数仓）或者 Lucene Index（Elastic Search 的格式）的方式。

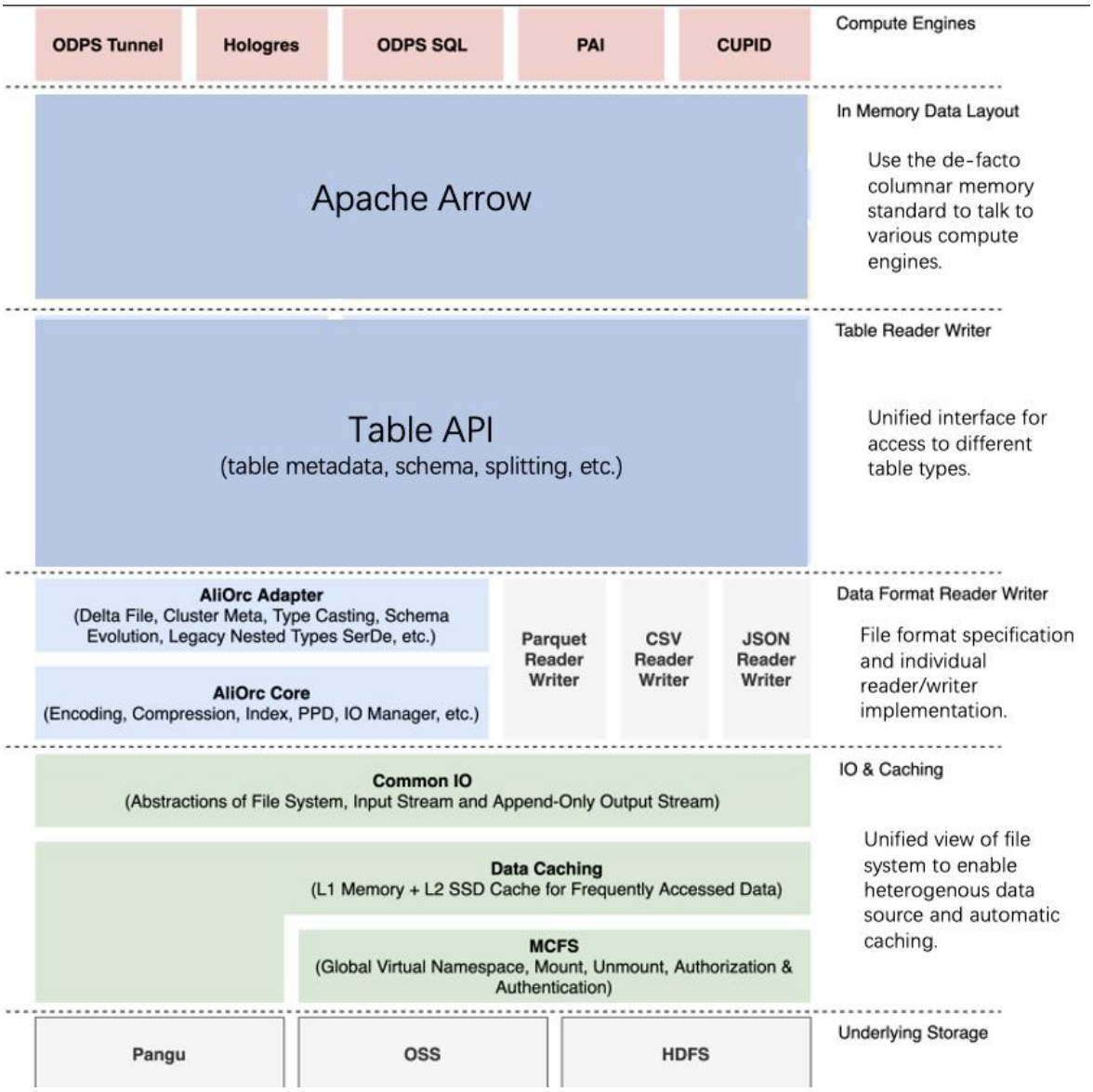
从存储系统的接口/内部功能看，越简单的接口和功能对应更开放的能力（例如 GFS/HDFS），更复杂更高效的功能通常意味着更封闭，并逐步退化成存算一体的系统（例如 AWS 当家数仓产品 RedShift）。两个方向的技术在融合。

以阿里巴巴大数据体系为例：

阿里巴巴的大数据存储系统，目前数万台存算一体/存算分离的服务器，总存储容量超过 5EB，支持阿里集团和阿里云的主线大数据业务。过去 5 年演进脉络如下：

1. 2017 年文件格式**全面升级为基于 Apache Orc 的 AliOrc 格式**：并将完整的 C++ ORC writer 实现和多个读写性能优化贡献开源社区。团队成员在开源社区影响力包括 1 位 PMC、**1 位 committer 和 2 位 contributor**，共计 40+提交，2w+行代码。
2. 2018 年，AliOrc 重点发展了**智能编码**：通过异步预读、高效的字典编码实现、动态内存管理、zero-copy 内存优化、浮点数和 decimal 类型编码优化、自适应编码等优化和改进，每个版本之间性能提升 20%。
3. 2018 年，开始**智能化分层存储**的探索。基于数据智能分析，将数据划分为热数据、温数据、归档数据和冷数据。通过 SSD、HDD、EC 编码和冷存机器等技术和硬件，将数据分布在不同的机器上。采用了智能分层存储后，降低了 15%+的存储成本，热数据的 TableScan 性能提升 50%+。

4. 2019 年，**开始近实时化的演进**。基于 AliOrc 的 block 级别 Zone Map，以及一种可以高效地将几个不同字段的相邻的数据联合排布在一起的 Z-Order Index，通过谓词下推显著提升过滤能力。
5. 2020 年，**全面升级为 AliOrc v2.0**。通过并行化编码技术、异步化并行 IO，以及高效的支持随机读的 IO Manager，进一步提升了读写性能。同时，AliOrc 与引擎深度定制，支持 row group 对齐、lazy read、lazy decoding，统一了实时和离线系统的列存文件存储。



(图：阿里大数据体系 - MaxCompute 存储架构)

展望未来，我们看到可能的发展方向/趋势主要有：

1. 平台层面，存储计算分离会在两三年内成为标准，平台向托管化和云原生的方向发展。平台内部，精细化的分层成为平衡性能和成本的关键手段（这方面，当前数据湖产品还做得远远不够），AI 在分层算法上发挥更大的作用。
2. Format 层面，会继续演进，但大的突破和换代很可能取决于新硬件的演进（编码和压缩在通用处理器上的优化空间有限）。
3. 数据湖和数仓进一步融合，使得存储不仅仅是文件系统。**存储层做的多厚，与计算的边界是什么**，仍然是个关键问题。阿里云给了一个答案，但仍然需要时间验证。

2.2 分布式调度，基于云原生，向统一框架和算法多元化发展

计算资源管理是分布式计算的核心能力，本质是解决不同种类的负载与资源最优匹配的问题。在“后红海时代”，Google 的 Borg 系统，开源 Apache Yarn 依旧是这个领域的关键产品，K8S 在大数据计算调度方向上仍在起步追赶。阿里巴巴大数据体系中，有“伏羲”作为自研的高性能分布式度系统，在全区域数据排布、去中心化调度、在线离线混合部署、动态计算等方面满足后红海时代业务场景下的数据/资源/计算调度需求。

常见的集群调度架构有：

- **中心化调度架构：**早期的 Hadoop1.0 的 MapReduce、后续发展的 Borg、和 Kubernetes 都是中心化设计的调度框架，由单一的调度器负责将任务指派给集群内的机器。特别是中心调度器中，大多数系统采用两级调度框架通过将资源调度和作业调度分开的方式，允许根据特定的应用来定做不同的作业调度逻辑，并同时保留了不同作业之间共享集群资源的特性。Yarn、Mesos 都是这种架构。

- **共享状态调度架构：**半分布式的模式。 应用层的每个调度器都拥有一份集群状态的副本，并且调度器会独立地对集群状态副本进行更新。如 Google 的 Omega、Microsoft 的 Apollo，都是这种架构。
- **全分布式调度架构：**从 Sparrow 论文开始提出的全分布式架构则更加去中心化。调度器之间没有任何的协调，并且使用很多各自独立的调度器来处理不同的负载。
- **混合式调度架构：**这种架构结合了中心化调度和共享状态的设计。一般有两条调度路径，分别为部分负载设计的分布式调度，和来处理剩下的负载的中心式作业调度。

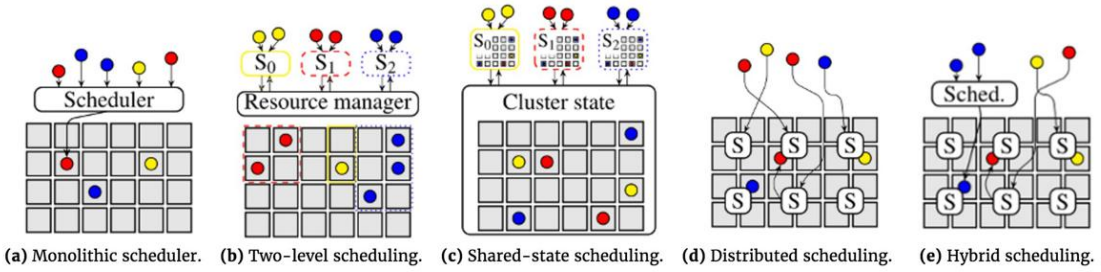


Figure 1: Different cluster scheduler architectures. Gray boxes represent cluster machines, circles correspond to tasks and S_i denotes scheduler i.

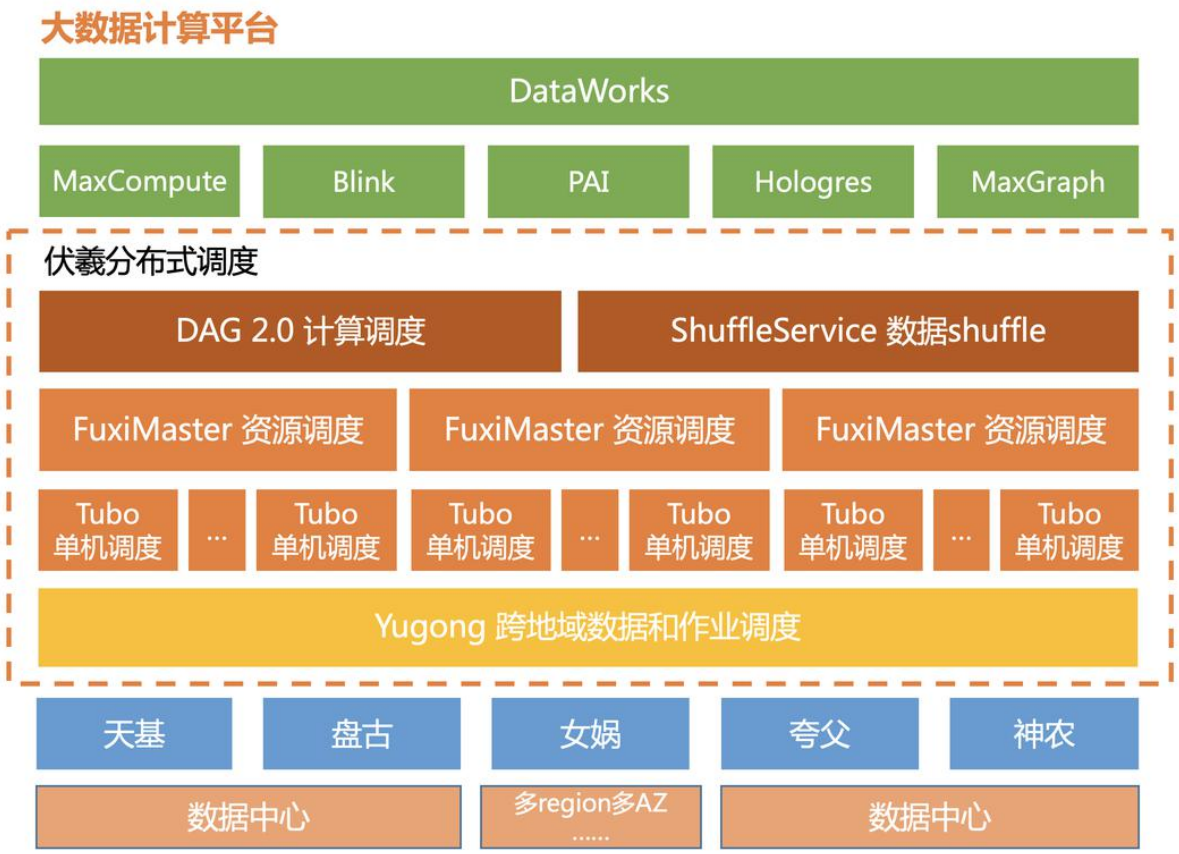
(图 ： The evolution of cluster scheduler architectures by Malte Schwarzkopf)

无论大数据系统的调度系统是基于哪种架构，在海量数据处理流程中，都需要具备以下几个维度的调度能力：

- **数据调度：**多机房跨区域的系统服务带来全域数据排布问题，需要最优化使用存储空间与网络带宽。
- **资源调度：**IT 基础设施整体云化的趋势，对资源的调度和隔离都带来更大的技术挑战；同时物理集群规模的进一步扩大，去中心化的调度架构成为趋势。
- **计算调度：**经典的 MapReduce 计算框架逐渐演化到支持动态调整、数据 Shuffle 的全局优化、充分利用内存网络等硬件资源的精细化调度时代。
- **单机调度：**资源高压力下的 SLA 保障一直以来是学术界和工业界发力的方向。Borg 等开源探索都假设在资源冲突时无条件向在线业务倾斜，但是离线业务也有强 SLA 需求，不能随意牺牲。

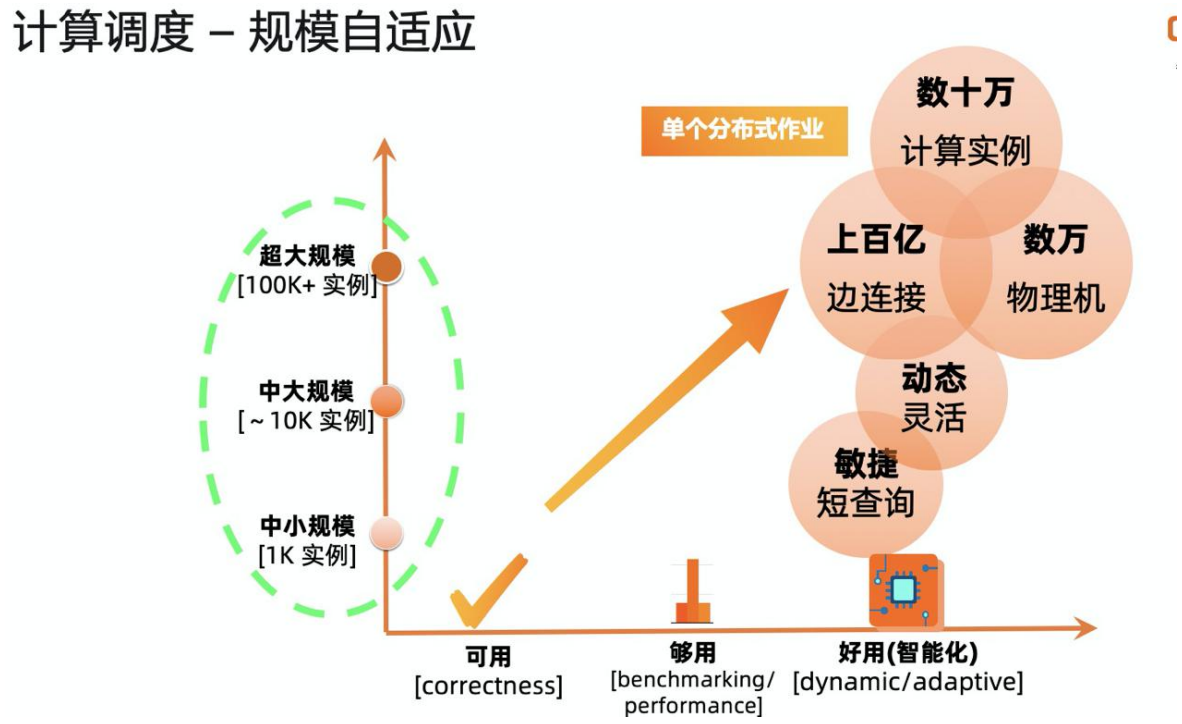
以阿里巴巴大数据体系为例：

阿里自研的调度框架就是 Fuxi(伏羲)。 Fuxi 系统主要负责飞天里的调度服务，在系统设计上是一个通用的调度系统，上层业务既包括偏在线的用户，也包括偏大数据计算的 MaxCompute（ODPS），Blink，Hologres，PAI，ADS 等用户。



- **数据调度：**Fuxi 在业内第一次提出了**多集群计算和数据调度**的概念，并基于跨集群数据缓存策略、跨集群计算调度、多集群业务排布等技术实现了跨地域维度上存储冗余／计算均衡／长传带宽／性能最优之间的最佳平衡。
- **资源调度：**Fuxi 近年来已升级到了**去中心化资源调度架构**，单集群支持 10 万服务器* 10 万并发 job 的高频调度，具备动态弹性扩展能力。基于 Fuxi 的资源调度能力，此前在阿里已经大规模落地应用了混部技术，随着 Fuxi 对 Kubernetes 等开源生态的进一步支持，**统一资源池&统一调度**的能力也使全局资源利用率进一步提升。

- **计算调度：**随着大数据业务的飞速增长和新计算模型的持续迭代，计算调度框架需要融合 AI 能力，以更好的动态自适应性应支持千万量级甚至更高量级的超大规模计算。



- **单机调度：**Fuxi 早年解决了作业快速启动和结束、资源调度策略、资源利用率提升等基本需求；近年来又提出了基于优先级的精细化资源隔离策略，解决了资源保障和利用率提升的难题。基于精细化的资源管控能力，混部技术支撑了双十一 0 点峰值的75%交易流量，在双十一超大规模数据量的压力下，保障了高优先级业务无一延时。

阿里巴巴大数据平台调度系统的发展，持续不变的一个重点是超大规模下对性能和成本的极致追求，如混部、跨集群调度等；同时，系统 AI 能力的融合，通过提升系统自适应能力来大幅提升调度系统的性能。阿里巴巴调度系统，基于开源体系也在进一步加强发展托管化与云原生的能力，如基于 Kubernetes 对全局资源池的统一调度研发，对各类异构资源池做统一管理，提升全局资源池的利用率，降低成本。

展望未来，我们看到可能的发展方向/趋势主要有：

1. K8S 统一调度框架：Google Borg 很早就证明了统一的资源管理有利于最优匹配和削峰填谷，尽管 K8S 在“非在线服务”调度上仍然有挑战，K8S 准确的定位和灵活的插件式设计应该可以成为最终的赢家。大数据调度器（比如 KubeBatch）是目前投资的一个热点。
2. 调度算法多元化和智能化：随各种资源的解耦（例如，存储计算分离），调度算法可以在单一维度做更深度的优化，AI 优化是关键方向（实际上，很多年前 Google Borg 就已经采用蒙特卡洛 Simulation 做新任务资源需求的预测了）。
3. 面向异构硬件的调度支持：众核架构的 ARM 成为通用计算领域的热点，GPU/TPU 等 AI 加速芯片也成为主流，调度系统需要更好支持多种异构硬件，并抽象简单的接口，这方面 K8S 插件式设计有明显的优势。

2.3 元数据服务统一化

元数据服务支撑了大数据平台及其之上的各个计算引擎及框架的运行，元数据服务是在线服务，具有高频、高吞吐的特性，需要具备提供高可用性、高稳定性的服务能力，需要具备持续兼容、热升级、多集群（副本）管理等能力。主要包括以下三方面的功能：

- DDL/DML 的业务逻辑，保障 ACID 特性，保障数据完整性和一致性。
- 授权与鉴权能力，保证数据访问的安全性。
- Meta(元数据) 的高可用存储和查询能力，保障作业的稳定性。

第一代元数据系统

第一代大数据平台的元数据系统是 Hive 的 Hive MetaStore（HMS），在早期版本中 HMS 元数据服务是 Hive 的内置服务，元数据更新（DDL)以及 DML 作业数据读写的一致性

和 Hive 的引擎强耦合，元数据的存储通常托管在 MySQL 等关系数据库引擎。

随着客户对数据加工处理的一致性（ACID）、开放性（多引擎，多数据源）、实时性，以及大规模扩展能力的要求越来越高，传统的 HMS 逐步局限于单集群，单租户，Hive 为主的单个企业内部使用，为保障数据的安全可靠，运维成本居高不下。这些缺点在大规模生产环境逐步暴露出来。

第二代元数据系统

第二代元数据系统的代表，有开源体系的 Apache IceBerg，和云原生体系的阿里巴巴大数据平台 MaxCompute 的元数据系统。

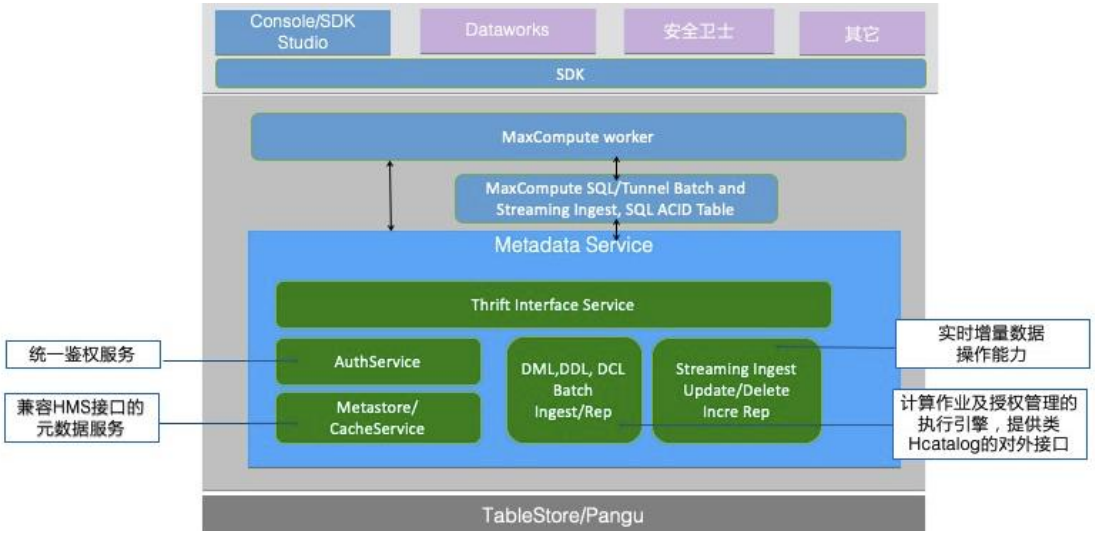
IceBerg 是开源大数据平台最近两年出现的独立于引擎和存储的“元数据系统”，其要解决的核心问题是大数据处理的 ACID，以及表和分区的元数据的规模化之后性能瓶颈。在实现方法上 IceBerg 的 ACID 依托了文件系统 POSIX 的语义，分区的元数据采用了文件方式存储，同时，IceBerg 的 Table Format 独立于 Hive MetaStore 的元数据接口，因此在引擎的 adoption 上成本很高，需要各个引擎改造。

基于未来的热点和趋势的分析，开放的，托管的统一元数据服务越来越重要，多家云厂商，都开始提供了 DataCatalog 服务，支持多引擎对湖和仓数据存储层的访问。

以阿里巴巴大数据体系为例：

阿里巴巴的大数据体系，提供了统一的元数据服务：

	HMS（第一代）	IceBerg（第二代）	MaxCompute（阿里巴巴大数据平台，同属第二代）
架构设计	早期作为 Hive 的内置服务，元数据存储使用数据库。	元数据存储使用 Table Format，独立于 HMS 接口。	元数据系统采用了微服务化 scale out 架构，元数据存储使用云原生 KV 存储引擎，同时服务化的接口封装了 KV 存储的细节，较好的兼容了 HMS 的接口。
多计算模式支持(统一元数据)	基于 HMS+HCatalog，支持 Hadoop 开源引擎生态	与 HMS 不兼容，引擎的 adoption 上成本较高，需要各个引擎改造。支持 Spark、Flink、Presto、Hive 等引擎。	兼容 HMS 接口，低成本支持 MapReduce, MaxCompute, PAI, Spark 等云原生和开源计算引擎
大规模 Schema 管理能力	单集群，单租户，没有线性扩展能力	Schema Evolution 定义清晰	基于元数据存储及读写的线性可扩展性，具备多集群支持能力。
ACID	仅 ACIDTable 类型支持事务保证，依赖悲观锁方式，其它数据表类型在异常情况下可能有脏数据。	依赖文件系统的原子性操作，版本切换依赖数据库系统，乐观锁，支持批量和增量更新，适用于“slow changing”的场景。	具备多租户，多引擎，大规模数仓平台的 ACID 能力。基于乐观锁并发控制方式统一支持批量&增量数据更新、支持流批接口、支持备份恢复、Versioning，以及以上操作的数据强一致性。
数据访问和权限控制的统一入口	基础的管控能力，缺少企业级能力	未包含	在元数据访问的入口做收口，支持统一的账号系统，以及统一的鉴权服务，具备高 qps，低 latency，高可用的在线鉴权服务的能力。
高可靠性&高稳定性	并发和性能以及规模处理能力表现一般	在 qps 和 latency 以及微服务化的 scale out 表现不足。	基于分布式多节点架构、大规模任务调度和流控能力，提供高稳定性保障。基于跨集群多副本和在线多版本备份恢复能力提供数据高可靠性保障。



（图：阿里巴巴大数据体系下的统一元数据系统）

Maxcompute 的元数据服务过去一年来，重点开发了湖上元数据服务，以及湖仓一体的统一的元数据服务能力。

展望未来，我们看到可能的发展方向/趋势主要有：

1. 湖仓一体进一步发展下，**元数据的统一化**，以及对湖上元数据和数据的访问能力建设。如基于一套账号体系的统一的元数据接口，支持湖和仓的元数据的访问能力。多种表格式的 ACID 的能力的融合，这个在湖上数据写入场景越来越丰富时，支持 Delta、Hudi、IceBerg 表格式会是平台型产品的一个挑战。
2. 元数据的权限体系转向企业租户身份及权限体系，不再局限于单个引擎的限制。
3. 元数据模型开始突破关系范式的结构化模型，提供更丰富的元数据模型，支持标签，分类以及自定义类型和元数据格式的表达能力，支持 AI 计算引擎等等。

2.4 多种计算引擎并存

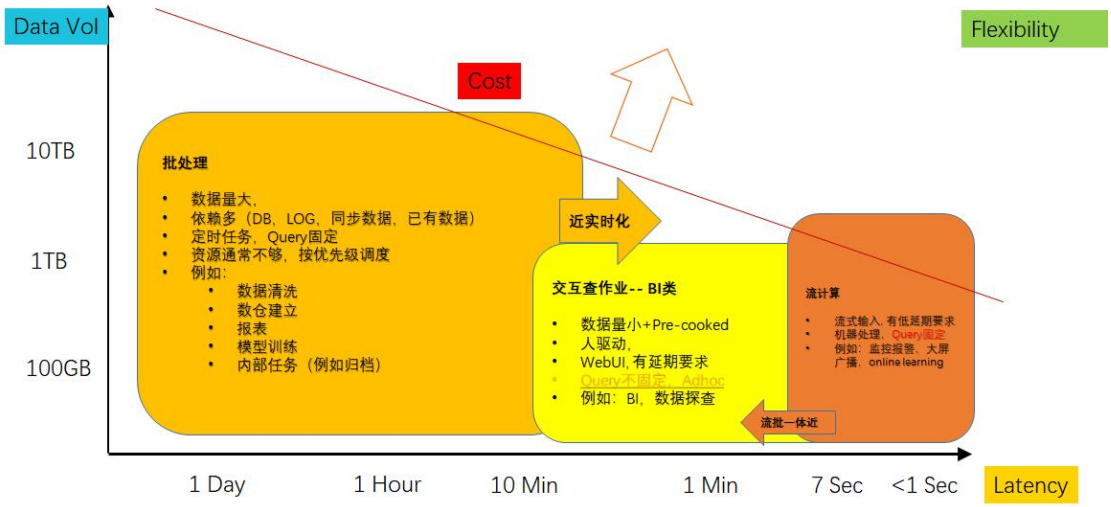
计算层是整个大数据计算生态的核心，是数据到价值转换的关键。 大数据场景中有各类计算形态，如批、流、交互式、多模、图、搜索、等多种计算模式。

大数据领域发展了 20 年，在“后红海”时代，**主流计算模式已经基本固定，形成批处理、流处理、交互式、机器学习四个核心方向**，以及一些小众/专门场景的计算模式。在开源社区领域，经过百舸争流式的竞争和沉淀，也基本形成了主流的社区形态。

除了机器学习，前三个方向有一定的 overlapping，例如 Spark 同时支持流、批和部分交互能力。但最终形成广泛影响力的引擎，都是在某一方向建立显著的竞争门槛。

整体看，计算引擎的发展将会在存储计算分离架构基础上，以一套数据支持多种计算模式：

- **存储计算分离，以及随后的 1+N 架构**（即一套数据之上支持多种计算模式）



- **批处理** - 是大数据处理的基础形态，以 Bulk Synchronous Parallelism（BSP）为基础原理，从 Map-Reduce（MR）模式开始发展起来，所谓“批”指的就是 Bulk（也译作 Batch）。Map-Reduce 的运算框架逐步发展成 Direct Acyclic Graph（DAG），上层语言也开始从 MR 的 Java 代码向 SQL 转型，第一版本集大成的批处理开源系统是 Hive+Tez。因为 Hive 2.0 是严格 BSP 模式，每次数据交互均需要落盘，牺牲了延迟和性能。Spark 抓住内存增长的趋势，推出基于 Resilient Distributed Dataset(

RDD)的运算框架，展开与 Hive 的竞争。当前在开源领域，Hive/Spark 是主流引擎，随 Spark 稳定性和内存控制逐步完善，Spark 逐步占领开源市场。目前批处理仍然是最主流的计算形态，整体的优化方向是更高吞吐/更低成本。最近两年，随**近实时方向**的兴起（以开源 Apache Delta/Hudi 为代表），批处理数据从接入到计算的延迟得的显著的降低，给用户提供了一种成本/延迟的另一个平衡点。

- **交互式分析** - 通常是面向分析场景（人驱动，中小规模输入数据/小规模输出数据），在中小规模 cook 好的数据上（通常是批处理之后的数据），基于更快的存储、更多的内存（bufferpool）、更实时的数据更新（通常是基于 LSMTree 的方案），也采用更多的 OLAP 优化技术（例如 Plan Cache）。优化方向更偏延迟（而非成本和吞吐率）。技术栈发展上，有两个脉络，一个是从分布式数据库角度发展起来，采用 MPP 架构，例如开源领域的 Apache Impala 和 Clickhouse，自研领域的 AWS Redshift。另一个是更偏云原生和大数据的架构，例如 Apache Presto。

批处理和交互分析，有天然的统一需求，因此很多自研的分析引擎也包括一定的批处理能力，形成一体化，例如当前如日中天的 snowflake。而 Google BigQuery 采用附加交互引擎（内置一个更快的 BI Engine）的方式形成一体化。从细节看，交互分析的引擎优化更偏数据库类优化方向，更强调用好 Memory 和 Index，Plan 相对简单对 QueryOptimizer 要求低，不需要支持丰富的 UDF，也不需要做 Query 中间的 failover。批处理引擎更面向 throughput 优化，核心是更优化的 Plan 以及尽量降低 IO，同时对 failover 要求高（因此部分数据要落盘）。这也是为什么 BigQuery 选择双引擎的原因。

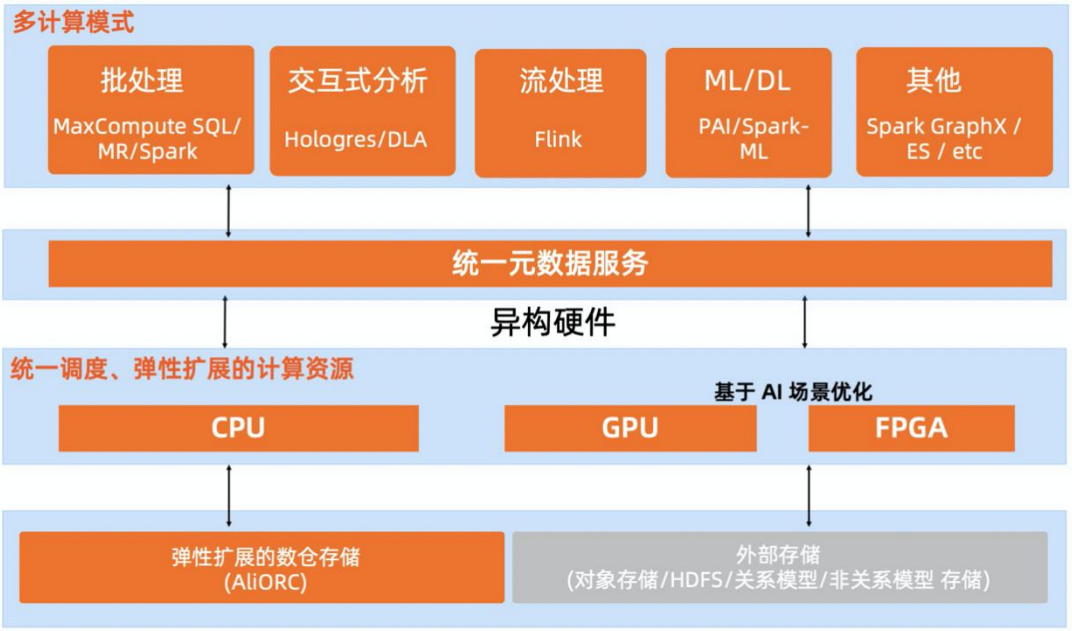
- **流处理** - 采用 Continuous Processing 的计算模式，通过本地状态保存（State）和 CheckPoint（CP，用来做 failover），形成分布式增量计算引擎。这种计算模式与 BSP 架构不同。主打的场景是实时大屏，监控报警以及最近流行的实时机器学习。系统面向低延迟优化，处理的是流式写入的数据，一致性模型（Exact-Once VS Atleast-Once）、LateEvent 处理方式、以及 Window 函数支持是不同流计算引擎设计的取舍。开源领域 Spark Streaming、Apex、Heron 和 Flink 经过竞争，Flink 因

具备完整 Exact-Once 语义保证和完善的 LateEvent 处理能力，最终获得社区广泛的关注。

- **MachineLearning(ML)/DeepLearning(DL)** - 以统计为基础理论的传统机器学习有丰富的生态，包括 Python 系、Matlab 等等。大数据领域 Spark MLib 以一套数据多种计算的优势，一度成为大数据传统算法的主流。随 AlphaGo 引爆深度学习领域，TensorFlow 和 Pytorch 成为业界标杆。目前 DL 领域仍然处于红海期，模型并行化以及超大模型是近期的热点。特别是随 DL 兴起，Python 作为标准语言开始流行，Spark 推出 Koalas 用于连接大数据与 AI 开发，Python 有取代 Java 成为命令式编程类（Imperitive）大数据开发语言的潜力（Decleritive 类编程标准一直是 SQL）。
- **其他小众计算模式** - 因满足不同细分场景，还有包括图计算，文本检索等引擎。图领域细分成三个子场景：图计算、图分析和图学习。分布式图分析场景目前仍未有完善的方案，图语言也在发展期，未形成统一标准。文本检索领域，主要基于倒排索引技术，开源生态 Elastic Search 成为生态主流。限于篇幅，这部分不再做更细节的介绍。

以阿里巴巴大数据体系为例：

阿里巴巴大数据体系支持多类引擎，除了大数据计算平台的引擎产品如 MaxCompute SQL Engine、EMR Jindo、Blink/Flink、 Hologres，也支持其他开源和阿里自研的引擎，如图计算引擎 MaxGraph、搜索引擎 Elastic/OpenSearch、Hive、Spark、Tensorflow、Numpy 等多种异构引擎。



- **批处理引擎 - MaxCompute SQL Engine**

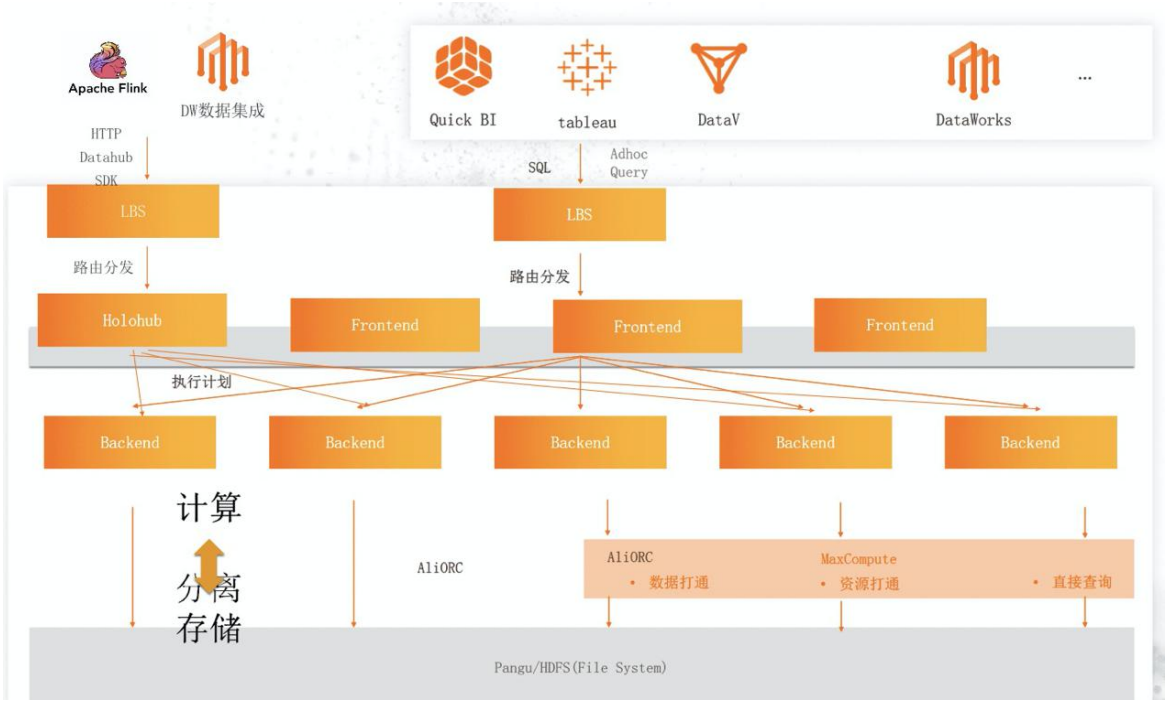
在 MaxCompute 的作业中，有 90%以上的作业是 SQL 作业，SQL 引擎的能力是 MaxCompute 的核心竞争力之一。 SQL 引擎的 3 个模块 - 编译器、运行时和优化器，在设计实现上做的优化主要有：

- 编译器：对 SQL 标准有友好支持，支持 100% TPC-DS 语法；具备强大的错误恢复能力。
- 运行时：基于 LLVM 优化代码生产，支持列式处理与丰富的关系算符；基于 CPP 的运行时具有更高效率。
- 优化器：支持 HBO 和基于 Calcite 的 CBO，通过多种优化手段不断提升 MaxCompute 性能。

(图：MaxCompute SQL 引擎发展历程)

- **交互式引擎 - MaxCompute 交互式分析 (Hologres)**

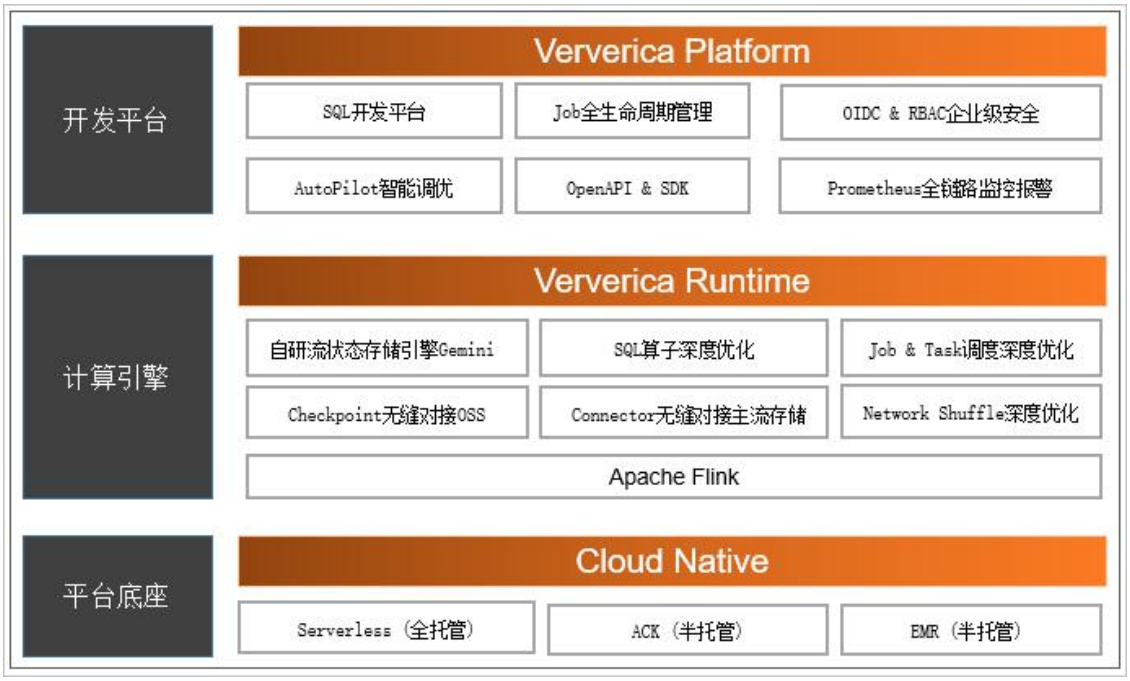
Hologres 与 MaxCompute 天然无缝融合，全兼容访问各种 MaxCompute 文件格式，充分利用异步模型和各种分析型查询引擎的优化技术，实现对 PB 级离线数据的毫秒级交互式分析。



(图：阿里巴巴大数据体系 - Hologres)

- **流处理引擎 - 实时计算 Flink 版**

Apache Flink 是目前业界最为流行和成熟的开源流计算技术，阿里云选择围绕 Apache Flink 技术打造了实时计算 Flink 版，并在集团内部有超大规模的验证，支持阿里集团所有实时计算数据业务。Flink 社区在 2020 年持续繁荣，蝉联最活跃的 Apache 项目；Flink 也成为了事实上的国内外实时计算标准。



(图：阿里巴巴大数据体系 - 实时计算 Flink 版产品架构)

- **MachineLearning/DeepLearning - PAI Tensorflow 等**

阿里巴巴大数据体系下，有多个自研的机器学习和深度学习引擎， 其中主要有 **PAI Tensorflow**：是 PAI 机器学习平台深度定制的面向全集团的深度学习计算引擎。在数据处理、算力如全局异构计算、模型优化层面做了大量创新和优化。

其他的机器学习和深度学习框架和引擎还有：

- **X-DeepLearning：**工业级深度学习框架，主要面向大规模广告/推荐/搜索场景。
- **MNN：**一款轻量级深度学习端侧推理引擎，主攻解决深度神经网络模型在端侧推理运行问题，目前已开源。
- **xNN：**蚂蚁金服研发的端侧机器学习引擎，致力于解决机器学习模型在以支付宝为代表的超级 App 以及 IOT 设备中部署所面临的技术难题。
- **ALPS：**蚂蚁的机器学习算法框架，目标是为蚂蚁的算法及业务同学提供一个高性能、易用的机器学习算法研发框架，并在此基础上沉淀蚂蚁具有金融特色的算法库（涵盖 CV/NLP/ML 等方向）和解决方案。

- **图计算引擎 - Graph Compute**

图计算服务（Graph Compute）支持图数据建模、导入和修改、支持 Apache TinkerPop 标准 Gremlin 语言进行图查询，并支持常见图分析算法，具有数据加载快、规模可扩展、查询延时低（毫秒级）和离在线混合引擎与共享存储等优势，帮助用户构建海量关系数据的图应用服务。

(图：阿里巴巴大数据体系 - Graph Compute)

展望未来，我们看到可能的发展方向/趋势主要有：

1. **近实时化成为主流 - 近实时化方案**，是在分钟级的延迟上做到数据的一致性，几乎不用依赖大量内存的 BTree 系统和常驻的服务，将成本降低到几乎和离线一致，在延迟和成本间找到一个新的平衡，会逐步取代部分离线的作业。
2. **IoT 领域兴起** - 随设备的智能化和 5/6G 网络兴起，面向 IoT 的分析会逐渐火热。计算形态可能会发生变化，从云为中心演进到云边端一体。
3. **大数据平台/引擎整体云原生化** - 新兴的引擎均基于云的架构重新设计，充分利用云的优势，降低复杂度的同时提供更多价值。随云原生化，Shared-Everything 架构成为未来的演进趋势。
4. **Learned based 优化** - 机器学习技术会充分融入大数据系统（甚至任何系统）的设计，优化器、调度系统、存储格式、Index/MV 设计等多个领域均会大量使用 AI 的技术来做优化。例如 Cost based Optimization 中的基于 Statistics 的 Cost 推导，会大量被 Learn based Statistics 取代。

2.5 框架与接入层

接入层和管控是一个子系统，主要用于服务背后的主系统。从架构和功能角度上看，与大多数服务的接入层差异不大，也不存在明确的演进和代差，因此简要概述。唯一值得一提的是，随越来越多的大数据平台走向“托管化”或者说“服务化”，框架管控层越来越厚，大多数企业级能力增强来自管控部分。例如，计算引擎是数仓类产品的核心，但最终用户需要的远不止引擎本身而是好用的数据仓库产品，就像发动机是汽车的核心部件，但用户所需的是完整、好开的汽车。

接入和管控层，抽象下来，主要功能包括：

1. 前端 API 接入-是系统对外的接口，通常提供 HTTP 协议接入，并具有认证、流控能力。部分系统提供 Web 接入能力。
2. 服务层 - 包括更多的业务逻辑，例如用户/租户管理，提供服务层面的访问控制，以及服务级别的流控。对于引擎来说，服务层很可能包括编译与作业分发能力，异常作业的检测与隔离等等。有些系统为了简化将 API 接入与服务层合二为一个服务进程。

设计考虑：

1. 服务于背后的主系统，功能随后台变化而变化，并没有“一定之规”。
2. 管控层直接决定系统的可用性，因此也需要完善的容灾能力，无状态的服务组件通常依托部署系统实现容灾，对于有状态服务，通常将状态存储在元数据系统或者底层存储系统中。
3. 很多独立引擎，特别是开源类，接入和管控层通常比较简单。对于企业级服务来说，很多额外的功能都在本服务扩展，也体现企业级服务的价值。例如：监控/运维能力、审计日志、计量计费、对计算系统热切换的控制等。甚至包括自动化作业调优等高级功能（例如 SparkCruise，来自微软 Azure 托管版的基于历史信息的自动作业调优子系统）。

4. 当用户选用多个系统组合搭建一套大数据平台，不同系统都会有自己的管控层，造成服务的冗余和各系统的割裂。因此很多云平台提供商，会致力于抽象统一规范和公共子模块，例如统一认证协议/服务（Kerberos 等）、统一权限管理，Terraform API 标准等。

以阿里巴巴大数据体系为例：

阿里巴巴的大数据平台（MaxCompute 为主）完整的实现了上图中所有的功能。限于篇幅展开两个特色功能：

1. **热升级能力** - 是全托管服务必备的能力，但不同系统实现的难度不同，离线系统相对简单，在线系统更难些。但基本的能力均包括：1）多版本平行部署执行能力，2）多版本间的切流控制。基于热升级的多版本能力，可以扩展出更多的功能灰度能力，甚至做到对单一客户/租户的特别版本。
2. **元仓系统** - 元仓系统有别于元数据系统（元数据系统是在线的核心元数据服务），它是扩展后更丰富的系统数据，包括数据访问统计信息、历史作业运行情况、数据血缘等等，大多数服务数据中台的数据均来自元仓系统。元仓系统的数据呈现形式是数据表，部分可以开放给客户，数据库系统中的 Information Schema 是一种元仓系统对用户的呈现。

展望未来，我们看到可能的发展方向/趋势主要有：

1. **托管化** - 框架与接入层是企业级能力增强的关键一层，随托管化成为大趋势，这一层会有大量的企业级能力加入，会逐步成为关键层。

2.6 数据开发与治理平台

随着大数据技术在众多领域的广泛应用，大量数据源需要接入大数据平台，多种数据处理引擎和开发语言被各类技术/非技术人员人员使用，复杂业务催生了规模庞大、逻辑复杂的工作流程，数据成为业务的生命线需要重点保护，数据作为业务的原动力需要更加方便快捷的被分析和应用。

让大数据计算平台真正能够服务于业务，还需要一系列数据研发和数据治理利器，以帮助数据工作者低成本和高效地获取数据洞察，实现业务价值：

- **数据集成：**支持关系型数据库、大型数仓、NoSQL 数据库、非结构化数据、消息队列等数据源之间的数据同步，包含批量数据同步、实时数据同步、整库数据迁移等，解决云上、跨云、跨地域以及本地 IDC 机房等复杂网络环境之间的数据同步问题。
- **元数据服务：**支持不同数据源的元数据发现与元数据归集，并构建数据目录和数据血缘服务，同时为上层数据开发与数据治理提供元数据服务。
- **数据开发：**提供在线数据开发 IDE，支持多种计算存储引擎，提供批计算、实时计算、交互式分析、以及机器学习等一体化数据开发服务，为各种技术/非技术人员提供高效极致的 ETL/ELT 研发体验。
- **调度系统：**支持大规模、高并发、高稳定性的细粒度周期性任务调度，支持流处理、批处理与 AI 一体化数据任务编排，保障数据生产的时效性、稳定性。
- **数据治理：**提供数据资产管理、数据质量控制、数据安全管理、监控告警、数据标准、成本优化等服务，保障数据仓库能够规范、健康、合规和可持续发展。
- **数据服务：**提供快速将数据表生成数据 API 服务的能力，连接数据仓库与数据应用的“最后一公里”，实现灵活可控的数据共享交换服务。

以阿里巴巴大数据体系为例：

在阿里大数据体系中， DataWorks 作为大数据平台操作系统，历经十二年发展，形成了涵盖数据集成、数据开发、数据治理、数据服务的一站式大数据开发与治理平台，每天稳定调度上千万数据任务，支撑了阿里集团 99%的数据业务。

展望未来，我们看到可能的发展方向/趋势主要有：

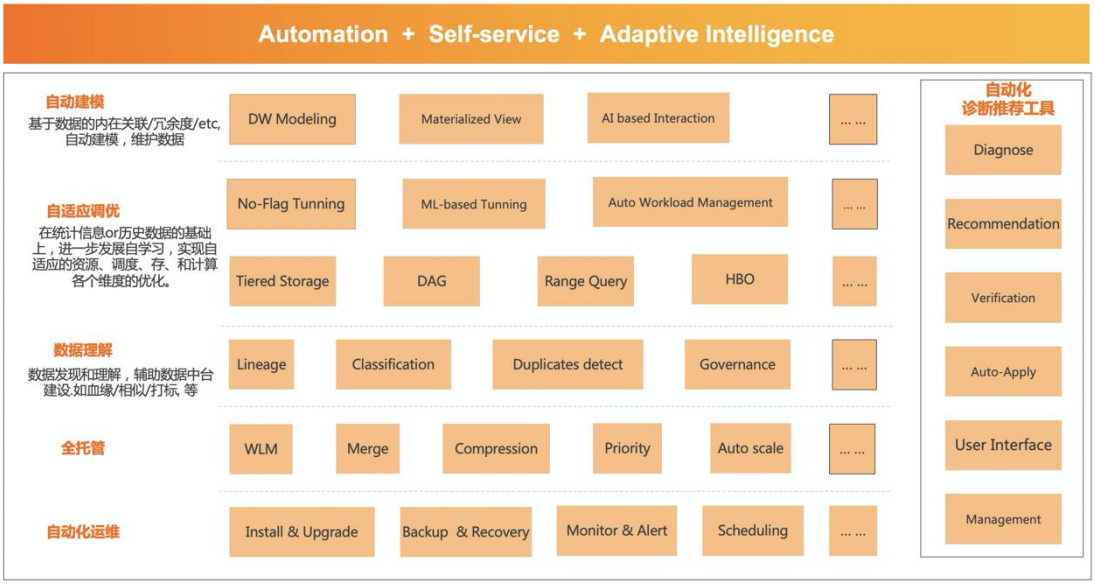
1. **低代码开发与分析：**数据的获取、分析与应用将逐步从专业开发人员覆盖到更多的分析师和业务人员，因此数据开发与分析工具将逐步从专业代码开发工具向低代码化、可视化工具演进。甚至是基于 NLP 和知识图谱技术，实现通过自然语言执行数据查询。低代码化开发与分析工具让非技术人员也能轻松获取数据洞察，实现数据价值的普惠，实现“人人都是分析师”。
2. **智能编程：**在传统的 ETL 开发过程中，存在着大量重复的或相似的编码工作，未来将在 AI 的加持下，通过语义分析、数据血缘探测、输入意图预测等技术，以智能编程助手的形式帮助开发人员实现更高效的编程。
3. **开发即治理：**过去我们大多习惯于先开发后治理，最终则让数据治理成为了负担。随着数据规模的不断增长、数据安全与隐私保护越来越受关注、数据业务化的持续发展，将不再允许数据治理仅仅是事后行为，数据治理将会融合到覆盖事前、事中和事后的大数据生产与应用的全链路中，数据开发与治理将协同并进。

2.7 智能化

随着大数据平台及其所承载业务的发展，我们也面临着新的挑战：

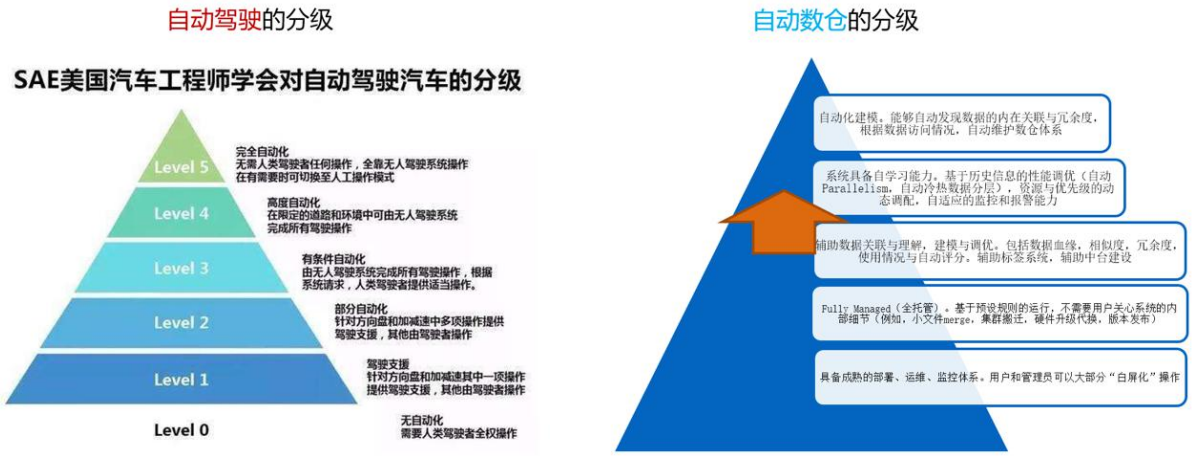
- 10PB 到 EB 级数据和百万级别作业规模，已经成为主流，海量数据和作业靠人很难管理和。传统的 DBA 模式或数据中台团队不再胜任。
- 多种数据融在一起，人无法在海量规模上理解数据的所有价值。
- 大数据系统经过多年发展，如果需要实现“跃迁”式的进步，需要体系结构层面的改造。

因此 AI for System 的概念兴起，基于 AI 的能力做系统的优化，在数仓领域我们可以称之为自动数仓（Auto Data Warehouse）。数据湖领域也可以有更多的自动化，但因为数仓方向的数据管理/调优能力发展更早，这个领域更领先。下图是一个基本的自动数仓能力分类。



自动化本身并不太可衡量，我们定义了一个“自动数仓”的能力分级，类比“自动驾驶”。分为 L1-L5.

- **L1 级：**运维能力白屏化和工具化。目前绝大多数系统都可以做到这个层次。
- **L2 级：**更好的系统托管化，底层系统对用户透明。例如小文件 Merge 自动化、软硬件升级透明、自动 loadbalance 等。很多全托管系统都可以做到这个层次（例如 Snowflake、MaxCompute）。
- **L3 级：**中台能力的自动化，辅助数据关联与理解，建模与调优。包括数据血缘，相似度，冗余度，使用情况与自动评分。辅助标签系统，辅助中台建设。市面上的很多数据中台产品聚焦在这一层。
- **L4 级：**系统具备自学习能力。基于历史信息的性能调优（自动 Parallelism，自动冷热数据分层等等），资源与优先级的动态调配，自适应的监控和报警能力，自动数据异常识别。目前大多数系统的能力边界在此，有巨大的发挥空间。
- **L5 级：**自动化建模。包含更高阶的数据理解，能够自动发现数据的内在关联与冗余度，根据数据访问情况，自动维护数仓体系。



最近 1-2 年，**自动化资源管理和自动化作业调优成为热点**，有非常多的研究性论文。几个核心元产品也推出新能力，例如 AWS Redshift 的自动 workload mgmt、自动 key sorting 和 table sorting，微软 Azure 的 SparkCruise（@VLDB2021）用来抽象公共子查询做 Materialized View。

以阿里巴巴大数据体系为例，整个领域都是新兴方向，限于篇幅，只展开介绍下面三个方面:

1. **History Based Optimization (HBO)** - 根据作业历史执行情况做调优，目前阿里的生产系统主要优化资源使用的方向，例如 memory、cpu 以及 worker 并行度等等，自动 Index 和自动 MV 抽取等能力在实验中。HBO 体系是一个典型 Learn based 系统，具备“任务执行历史 + 集群状态信息 + 优化规则 → 更优的执行”的特点，系统基于一套信息收集和优化的框架，具有很好的扩展性，方便添加多种新优化能力。从架构看，HBO 包含两部分：

a. **Online：**是个在线系统，用于查找是否存在相应的 hbo optimization 规则，如果命中，则按照规则进行。这个规则可以是资源分配优化，可以是物理执行计划的并发度，也可以是优化器里面的 Optimization；

b. **Offline：**获取任务的历史执行记录，按照一定的策略生成 hbo optimization 规则。

2. **基于学习的智能跨域调度** - 本质上也是基于历史信息的的优化过程，但更聚焦于跨数据中心的数据排布和调度带宽优化。通过精细化的数据访问统计，做更优的数据规划，并持续根据访问调优。具体算法和架构可以参考 Yugong@VLDB 2019

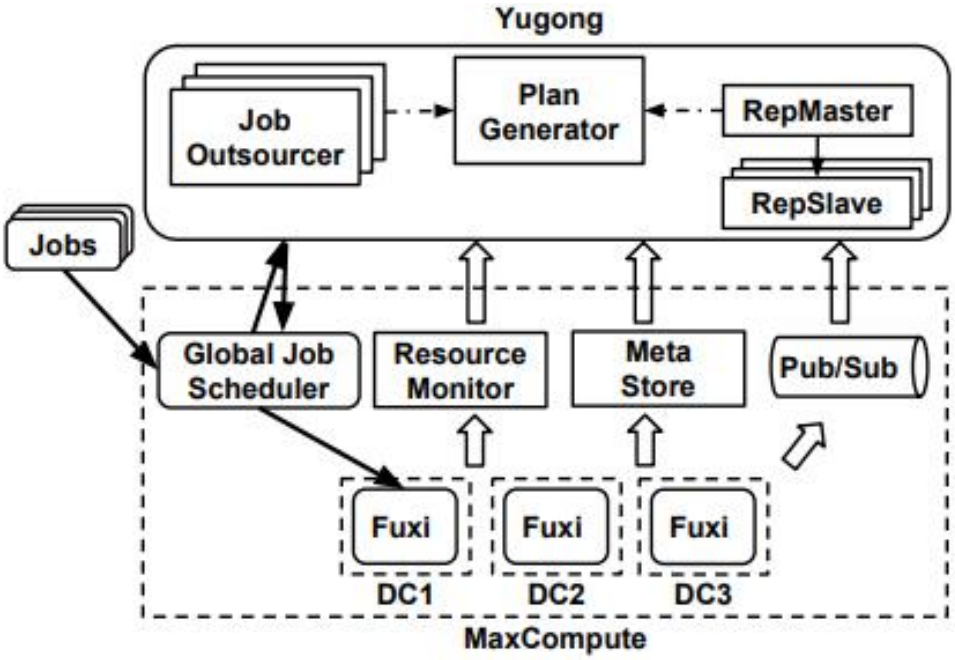


Figure 7: Yugong Components

3. 数据异常的自动发现和报警 - 大数据的数据异常发现是个关键问题，当前主流的做法是基于规则的质量检测，这些检测通常覆盖在作业工作流的各个地方。当前阿里大数据平台有超过 2000 万作业，数百万的质量检测规则，因此面临规则很难匹配动态变化的数据的问题。因此我们特别开发了动态阈值智能预测与算法匹配周期性波动的能力。具体算法见 RobustPeriod@Sigmo2020。

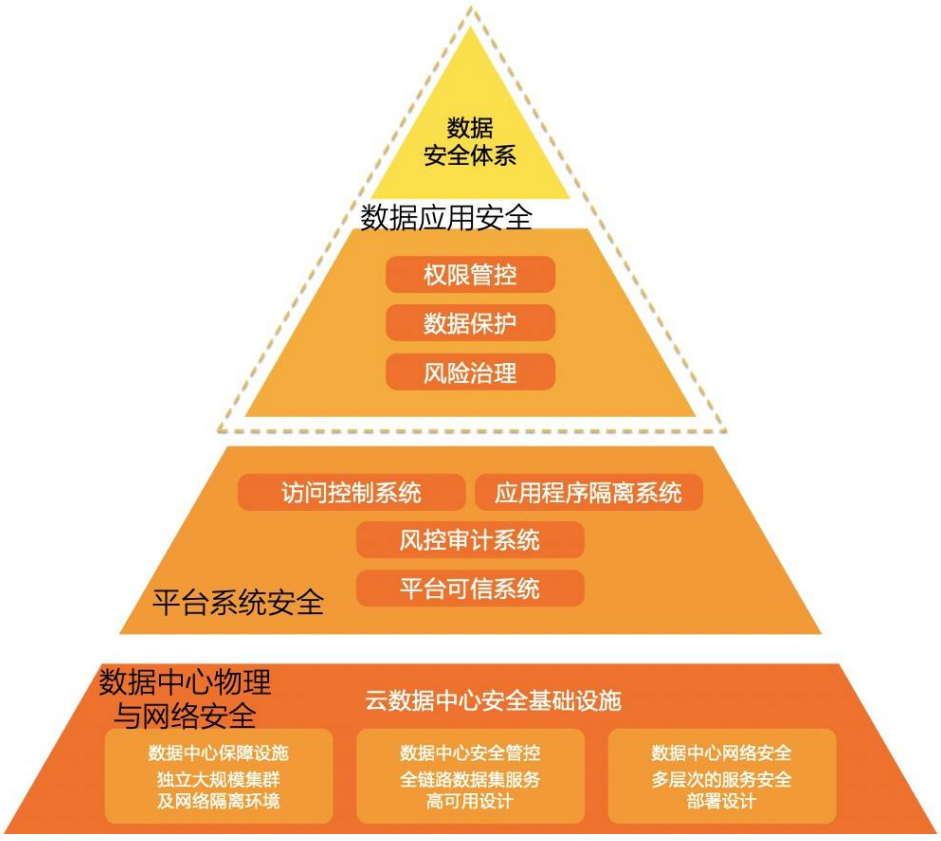
展望未来，我们看到可能的发展方向/趋势主要有：

2.8 安全与隐私保护

随着大数据的发展，数据在多方数据融合场景下能发挥更大的价值。然而在这种场景下用户的隐私保护以及数据的合规问题成为了严重的问题。**问题的本质是数据的开放性与使用安全性的平衡。**安全能力，包括数据安全/隐私保护能力，是大数据体系中的重要能力基线之一。

信息的可用性、信息的完整性、以及信息的保密性是信息安全的三个基本要素。我们将企业级大数据中台要面临的安全风险，根据其所涉及的系统及技术领域的不同，分为三个层次。

1. 最基础的层次是**数据中心的物理安全与网络安全**，数据中心是构建大数据中台的基础，数据中心自身安全性和网络接入安全性直接影响到企业大数据中台的可用性。主要包括数据中心保障设施、数据中心安全管控、数据中心的网络安全等几个维度的建设。**当云架构成为主流，物理安全方面通常由云厂商承担。**
2. 在这之上是**企业大数据平台的系统安全**，由大数据平台内部的多个安全子系统构成，如访问控制、应用程序隔离、平台可信、风控和审计等子系统。这些子系统共同保障大数据平台的完整性。
3. 最上层是**数据应用安全**，贴近于用户的应用场景。通过在这一层提供丰富多样的数据安全产品，大数据中台为用户应用数据的各类业务场景提供切实可靠的数据安全能力。

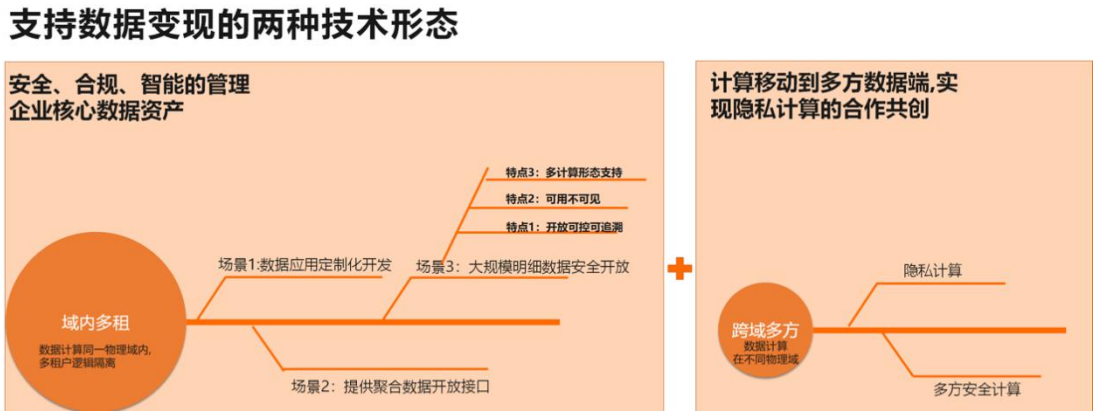


以阿里巴巴大数据体系为例：

阿里巴巴的大数据体系，依托阿里云底座完成数据中心物理安全，平台和数据安全方面也做了非常多的工作，下图给出了一个基于数据生命周期的数据安全管理体系。里面有非常多的子领域，比较存储加密、敏感数据发现和保护、数字水印等等。

展望未来，我们看到可能的发展方向/趋势主要有：数据安全共享

随数据被认知为资产，数据变现成为一个热门话题，它背后的技术：数据安全共享和多方安全计算也成为热点方向。总体看，数据变现（也称为数据安全共享），有两种典型场景：一方数据对外售卖，多方数据交互计算。



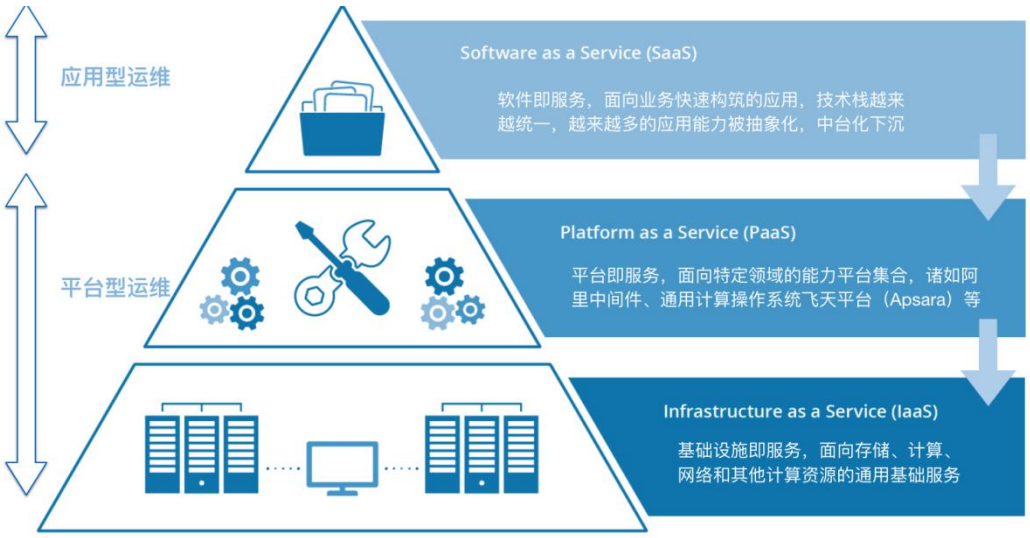
域内多租的方案，通常需要细粒度的接入/访问控制、计算隔离、下载保护等技术配合。主流数仓产品均提供这类方案（比如 Snowflake 的 DataSharing）。另外，这个领域的一个研究方向是基于差分隐私（Differential Privacy）加密的密态计算（例如 DPSAaS@Sigmod2019）。

多方数据交互计算方案，通常基于联邦学习技术（Federated Learning）。

2.9 运维

运维是伴随着任何一家公司的产品，在整个产品生命周期中一直是存在的。运维负责公司产品业务的正常运转，处理紧急故障响应，保障业务连续性，产品可用性改进，性能&效率优化，变更管理，监控，容量规划与管理等相关工作：这些是运维核心的定义所在。

随着大数据行业的发展，运维走过了传统 Ops 到专业化、工具平台化、再到云化数据化、甚至是到了智能化的阶段，从云计算 SaaS/PaaS/IaaS 三层的演进趋势我们可以看到，IaaS 与 PaaS 之间的分界线越来越往上走，基础设施越来越统一，云化虚拟化的趋势抹平了基础设施层；同时 SaaS 与 PaaS 层的分界线也没有那么清晰，在 SaaS 层快速构筑应用的成本越来越低，越来越多的 SaaS 层能力抽象后下沉到 PaaS 层，拿来即用，也可以说是 PaaS 层在往上层演进。结合云计算 SPI 三层的发展，我们也可以将运维粗略划分为面向 SaaS 层的应用型运维、面向 PaaS 和 IaaS 层的平台型运维。



大数据运维业务围绕稳定性（质量）、成本及效率主要关注如下：

- 针对日常业务稳定性可以分为日常事件管理、问题管理、变更管理及发布管理的标准化 ITIL 流程；
- 针对成本管理包含了从资源预算、资源采购、预算执行、财务账单、过保替换等围绕资源生命周期管理的相关事项；
- 针对效率包含了如何开发一体化的运维平台以高效支撑业务监控、服务管理、系统管理、应急/安全管理等。

以阿里巴巴大数据体系为例：

阿里大数据运维属于既包含应用型运维，又包含平台型运维的运维体系。 为支撑阿里大数据海量规模产品及业务的运维，我们需要建设一套分层的运维体系，借用 SPI 三层思想来划分，可以将这套运维体系称之为运维垂直 PaaS 体系：

- 运维 IaaS 层：映射基础设施的运维层，为上层运维平台及业务提供管控能力；
- 运维 PaaS 层：面向运维领域功能的服务层，比如流程编排、作业调度、配置管理、运维分析服务、审批/通知、脚本/命令通道服务等；

- 运维 SaaS 层：面向具体大数据平台业务的定制化智能运维运营应用，跟随所运维对象系统一起向前迭代发展的运营站点。

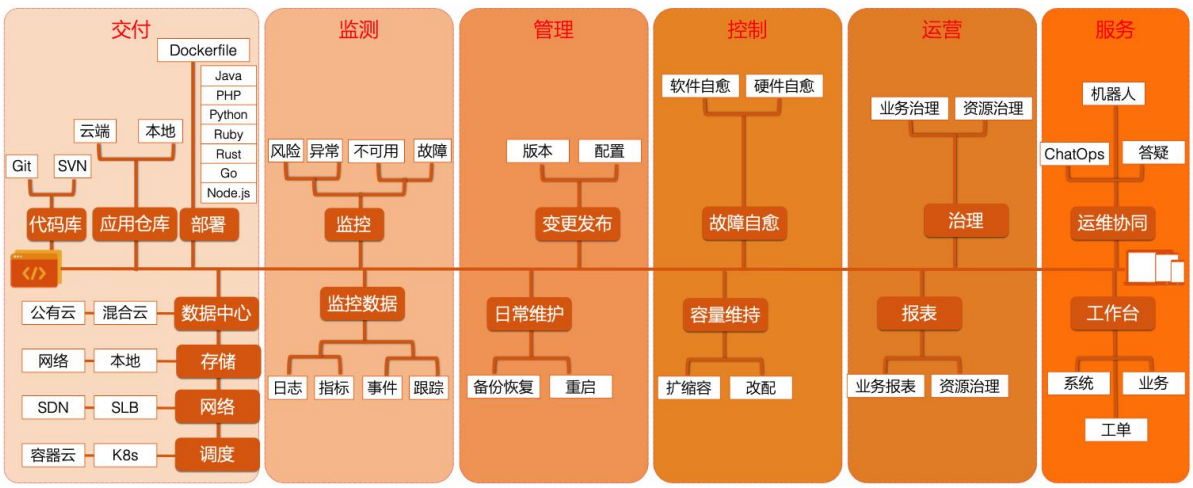
在垂直运维体系之上，运维业务也可以划分为 SRE 大中台及 SRE 小前台：“大中台”强调运维资源整合、运维能力沉淀的平台体系，为前台运维业务开展提供可快速复用的底层技术、数据等资源和能力；“小前台”就是贴近最终运维所服务对象的相关运维业务，基于中台整合的数据和产品技术能力，对各前台业务形成强力支撑，快速试错，快速行动。

大数据体系发展至今，服务器规模已发展到数万、数十万、甚至百万台规模。随着基础设施规模的发展，运维系统也经历了一个从人工、到平台化、再到智能化的自然的演进过程，运维的演进需要能支撑起大数据基础设施的高复杂、高安全、高可靠、高效率要求。

展望未来，我们看到可能的发展方向/趋势主要有：

1. 产品化

运维的产品化，本周上是指让各类运维动作的更标准，更可控。产品化是把对本身服务客户的产品的运维需求、和运维产品本身的需求、集成到产品功能上，并迭代传承。同时通过这些标准化的动作，逐步把底层的一些运维数据统一起来。

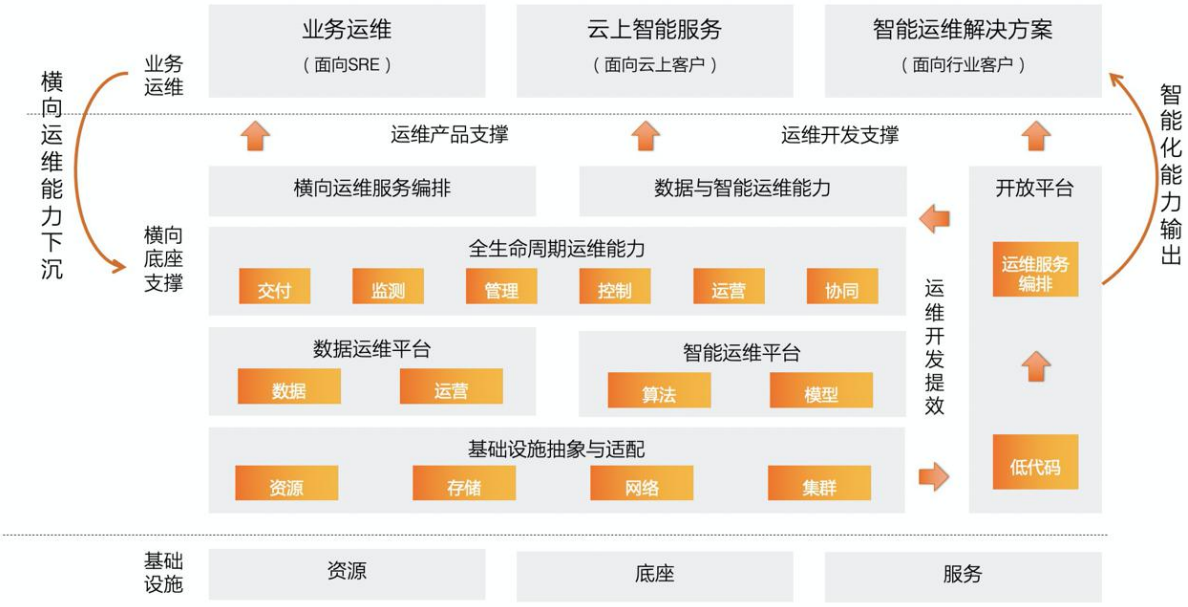


2. 数据化

随着产品规模的扩大以及系统的复杂多产品依赖，在整个过程中会产生大量的系统数据，随之数据化能力会凸显的非常重要。数据的收集存储，计算处理，运维数仓建设，数据分层，实时性，运营分析等相关数据化能力会逐步成为必须基本能力。重点需要关注运维数据的工程，集成，安全和质量。

3. 智能化

在超大业务规模下，逐步按以前的传统的运维操作方式不能更高效的支持好业务，一个例子对线上复杂问题的快速定位和修复。但这类人脑的规则级随着复杂度增长以及产品周期发展不会是一个线性提升，这个时候是需要通过一些智能化算法能力去帮助处理这些海量数据，从中找到一定的结果规则，辅助加快专业人士的判断。但这块的技术挑战非常巨大，同时对资源也有一定的消耗。但这个方向是应该持续发展。


横向运维能力下沉
业务运维
横向底座支撑
基础设施
业务运维
（面向SRE）
云上智能服务
（面向云上客户）
智能运维解决方案
（面向行业客户）
智能化能力输出
运维产品支撑
运维开发支撑
横向运维服务编排
数据与智能运维能力
开放平台
全生命周期运维能力
交付
监测
管理
控制
运营
协同
运维服务编排
运维开发提效
数据运维平台
数据
运营
智能运维平台
算法
模型
基础设施抽象与适配
资源
存储
网络
集群
低代码
资源
底座
服务

03 大数据体系未来演进的 4 大技术趋势

趋势 1：近实时架构兴起

在离线 batch 计算和纯流式实时计算之间，以开源 Apache Delta/Hudi 为代表的近实时架构成为热点。近实时架构避免了流计算庞大的状态存储与管理，在成本和延迟上找到了另一个平衡。随近实时架构的形成，计算架构最终完成从离线到实时全频谱支持。

趋势 2：数据共享与隐私保护成为热点

数据成为资产，开始具备可变现和可交易的能力。可保护隐私的数据交换/共享能力成为强劲的需求。基于 Differential Privacy 的数据编码交易，以及基于 Federated Learning 的多方面安全计算是该领域的热点技术。

趋势 3：IoT 成为新热点

目前人的行为数据（日志）是大数据计算的主要来源，超过 80%的数据都来源于行为日志（例如浏览、点击）。随 5G+智能化设备的兴起，设备日志会成为更大的数据源增长点，面向海量低价值设备数据的处理和优化，需要得到更多的关注。

趋势 4：AI for System

AI for System，即上文中提到的大数据自动驾驶。AI 作为工具，成为优化的常用手段。在大数据领域，随数据量/系统复杂度的增长，DBA 模式已经不再试用。利用算法优化系统成为主流方向，大数据的“自动驾驶”会越来越自动。

04 大数据体系内待探索的 3 个疑问

大数据技术收敛，并进入普惠和业务大规模应用的阶段，渗透到各行各业。超大规模数据计算和基于数据的智能决策，已经是企业业务数据化运营的重要基础。不过，在后红海时代，大数据体系发展有 3 个疑问值得我们关注：

疑问 1：引擎发展呈现跨界的趋势，但最终是否能够诞生一套引擎满足多样的计算需求，并兼顾通用性和效率？

随大数据系统整体架构的稳定，各种引擎的发展逐渐进入收敛期，批计算、流计算、交互分析、机器学习收敛成为四个核心计算模式，每个模式均有主线开源引擎成为事实标准。

过去 3 年没有再诞生主流的开源计算引擎（每个模式中，引擎的发展脉络详见第二章节）。同时，引擎边界开始变得模糊，HTAP 等 Hybrid 模式成为探索的新趋势，计算模式是否进一步收敛，收敛的终态会是什么样子，是个热点话题。

疑问 2：关系模型之外，是否会发展出其他主流计算范式？

大数据领域整体还是以二维关系表达和计算为基础（Relational DB 的理论基础），是否有新的计算范式在数据库领域也持续讨论了多年，尽管有包括图计算在内的其他计算范式，但过去的 40 年，关系运算持续成为主流。

其中核心原因，笔者个人的判断是二维关系表达更贴近人的理解能力，或者说高维表达和处理很难被人理解和处理。但关系表达有显著的短板，它无法处理半结构化和非结构化的数据（比如音视图类的数据）。

近几年兴起的深度学习技术，带来了一种全新的处理方式，海量正交化的高维特征作为输入，由深度神经网络理解数据，以模型作为产出的引擎计算出结果。这种方式避免人脑对数据处理的局限性，可以在更高维度更复杂数据上做处理，给未来提供了一种新的处理方式的可能性。

但深度学习核心仍然在寻找“最好”的 co-relation，可解释性，推导逻辑以及对结果正确性保证都不够好。

疑问 3：基于开源自建与直接选购企业级产品，谁更能获得用户的认可？

开源软件是大数据发展的关键推手，助力大数据系统的普及化。但面临如下挑战：开源系统的软件交付模式，也给很多客户带来高维护成本。

以一个典型的腰部互联网企业为例，一个 100 台规模的大数据平台硬件投入大约 200 万/年，同时需要维持一个 3-5 人的研发/运维团队，年成本 200-300 万/年。综合 TCO 高达 450 万/年。

这也是为什么像 Snowflake 这样的自研企业级产品流行的原因，大多数不具备深度研发能力的公司，愿意为更丰富的企业级能力和更低的综合 TCO 买单；大数据系统开发进入深水区，投资巨大，需要高商业利润才能支持。

事实上，云计算四巨头均有自己的自研产品提升利润率的同时也提升差异化竞争力（例如 AWS Redshift，Google BigQuery，阿里云飞天 MaxCompute）。

而每个开源社区背后无一例外均有商业公司推出企业版（例如 Databricks 之于 Spark，VVP 之于 Flink、Elastic 之于 Elasticsearch）。

因此，长期看，大多数用户（特别是中小型）进入“技术冷静期”后，开始审慎考虑综合投资收益，考虑上云、以及直接采购企业级产品+服务（放弃自建平台）。

本文试从系统架构的角度，就**大数据架构热点，每条技术线的发展脉络，以及技术趋势和未解问题**等方面做一概述。特别的，大数据领域仍然处于发展期，部分技术收敛，但新方向和新领域层出不穷。本文内容和个人经历相关，是个人的视角，难免有缺失或者偏颇，同时限于篇幅，也很难全面。仅作抛砖引玉，希望和同业共同探讨。

钉钉扫码加入

飞天大数据平台交流群

钉钉扫一扫加入

DataWorks 产品交流群

钉钉扫一扫加入

MaxCompute 产品交流群

扫一扫加入 MaxCompute

交互式分析（Hologres）交流群

钉钉扫一扫加入

实时计算 Flink 产品交流群

钉钉扫一扫加入

E-MapReduce 产品交流群

钉钉扫一扫加入

机器学习 PAI 产品交流群

钉钉扫一扫加入

搜索推荐技术交流群

阿里云开发者“藏经阁”

海量电子书免费领