盘点2018年晋升为Apache TLP的大数据相关项目

去年，我整理了2017年成功晋升为Apache TLP的大数据相关项目进行了整理，具体可以参见《盘点2017年晋升为Apache TLP的大数据相关项目》。现在已经进入了2019年了，我在这里给大家整理了2018年成功晋升为Apache TLP 的大数据相关项目。2018年晋升成 TLP 的项目不多，总共四个，按照项目晋升的时间进行排序的。

Apache Trafodion：基于 Hadoop 平台的事务数据库引擎

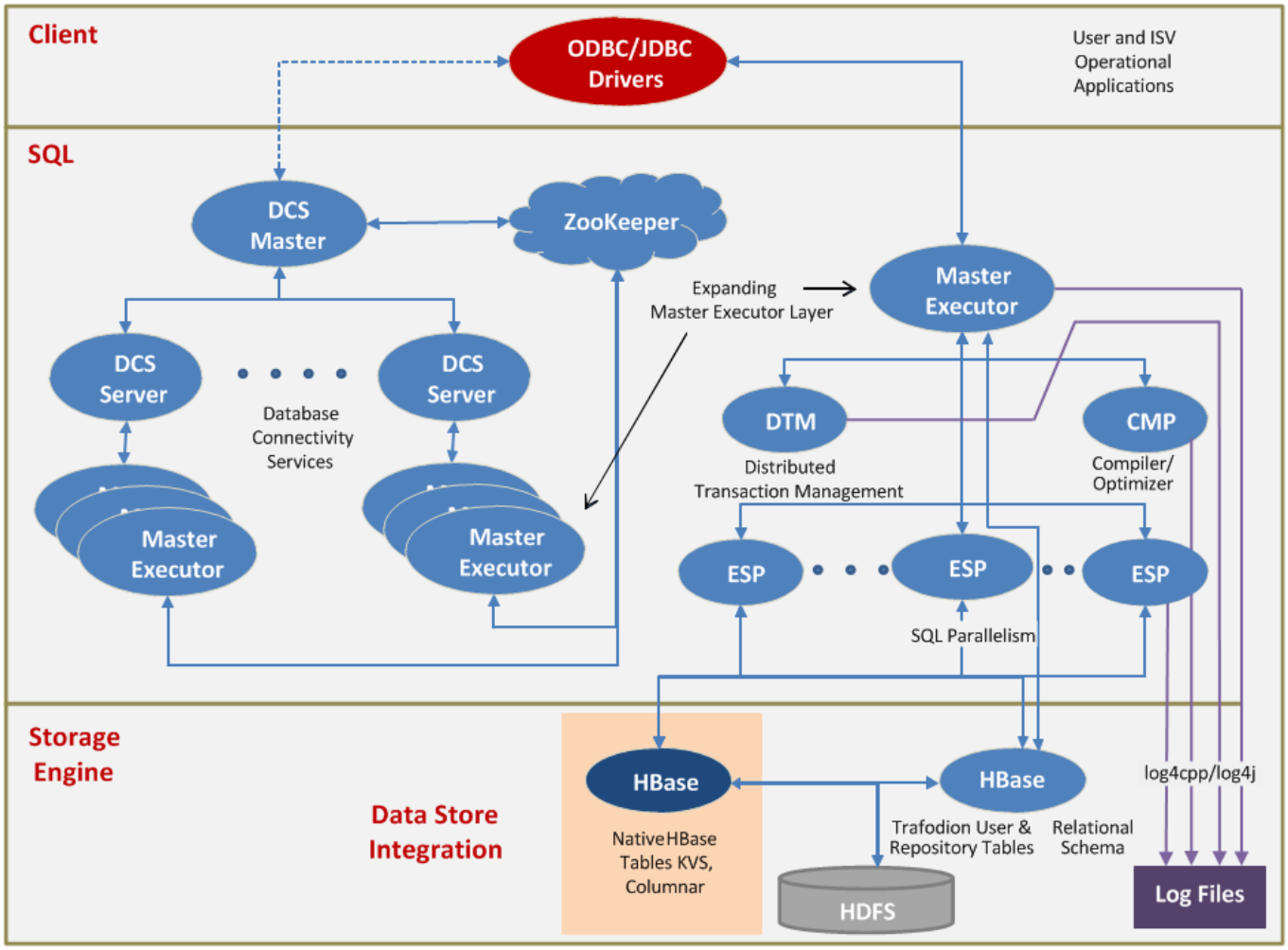
2018年01月10日，Apache Trafodion 成功晋升成 TLP 的，参见这里。Apache Trafodion 最初由 Hewlett-Packard Company 和 HP Labs 的信息技术部门开发，用于在Apache HBase上提供针对大数据事务或操作工作负载的SQL查询语言是由惠普开发并开源的基于 Hadoop 平台的事务数据库引擎。提供了一个基于Hadoop平台的交易型SQL引擎。它是一个擅长处理交易型负载的Hadoop大数据解决方案。其主要特性包括：

- 完整的ANSI SQL语言支持
- 完整的ACID事务支持。对于读、写查询，Trafodion支持跨行，跨表和跨语句的事务保护
- 支持多种异构存储引擎的直接访问
- 为应用程序提供极佳的高可用性保证
- 采用了查询间(intra-query)并发执行模式。轻松支持大数据应用
- 同时应用编译时和运行时优化技术，优化了OLTP工作负载的性能

事务管理特性包括

- 事务串行化基于开源项目HBase-Trx的实现原理，采用多版本并发控制(MVCC)
- 增强的故障恢复机制保证了数据库中用户数据的一致性
- 事务管理器支持多线程的SQL客户端应用
- 支持非事务型数据访问，即直接访问底层HBase表

Apache Trafodion的进程构架



如果想及时了
解Spark、Hadoop或者Hbase相关的文章，欢迎关注微信公共帐号：iteblog_hadoop

上图描述了Trafodion的进程构架。主要进程包括：

- 客户端应用通过JDBC或者ODBC访问Trafodion。Trafodion的ODBC驱动采用了优化的wire protocol，高效地同Master Executor进程进行网络交互。上图演示了一个Type 4的JDBC配置。.
- Master Executor是负责执行用户SQL语句的主进程。它内部包含了一份SQL compiler代码的拷贝，因此多数SQL语句可以在Master Executor进程内部进行编译而无需和单独的编译进程进行通信。此外，所有执行计划中的root节点都在Master Executor进程中执行。
- 少部分SQL语句（比如,DDL和一些应用工具）需要启动第二个独立的编译器进程对SQL语句进行处理；即上图中的CMP进程
- Trafodion 支持多种不同形式的并发执行方式。当系统生成了并发查询计划时，系统会动态地启动多个ESP进程，即Executor Server Processes。每一个ESP负责执行查询计划中的一个分段(fragment)
- DTM进程负责分布式事务。DTM的职责包括日志管理和事务协调。
- Trafodion支持访问原生HBase表，为此，SQL引擎将读取HBase的元数据。为了提供更好的OLTP访问性能，Trafodion还提供了定制的Trafodion表结构，用HBase

Table进行存储。Trafodion表拥有自己的元数据，同样存储在HBase中。

Apache Trafodion 官方网址：https://trafodion.apache.org/

Apache HAWQ：Hadoop 原生 SQL 查询引擎

2018年08月23日，Apache HAWQ 成功晋升成 TLP 的，参见这里。

Apache HAWQ 是一个高级 SQL-on-Hadoop 弹性查询引擎和分析型数据库。它结合了 MPP 数据库的关键技术优势和 Hadoop 的可扩展性。HAWQ 原生支持访问 HDFS 上的数据，并提供业界领先的性能和线性可扩展性。支持 PB 级数据交互式查询，并为用户提供了完整的、符合标准的 SQL 接口。

HAWQ 提供业界领先的性能和线性可扩展性。使得用户可以对 PB 级数据集进行交互查询分析。HAWQ 为用户提供了完整的，符合标准的 SQL 接口。更具体地说，HAWQ具有以下特征：

- 支持本地或云部署
- 健壮的 ANSI SQL 遵守 SQL-92, SQL-99, SQL-2003, OLAP 扩展
- 极高的性能 - 比其他Hadoop SQL引擎快很多倍
- 完整的事务处理能力和一致性保证:ACID
- 基于高速 UDP 的动态数据流引擎
- 基于按需虚拟段和数据位置的弹性执行引擎
- 支持多级分区和基于 List/Range 的分区表。
- 支持多种压缩方法: snappy, gzip
- 支持多语言: Python, Perl, Java, C/C++, R
- 通过 MADLib 实现先进的机器学习和数据挖掘功能
- 动态节点扩展:以秒为单位
- 最先进的三级资源管理:集成YARN分层资源队列。
- 轻松访问所有HDFS数据和外部系统数据(例如，HBase)
- 原生支持 Hadoop : 从 存储(HDFS), 资源管理(YARN) 到部署 (Ambari).
- 身份验证 & 细粒度授权：Kerberos，SSL和基于角色的访问
- HDFS和YARN的高级C/ c++访问库:libhdfs3和libYARN
- 支持绝大多数第三方工具: Tableau, SAS 等
- 标准连接: JDBC/ODBC

HAWQ 将复杂的查询分解为小任务，并将它们分发给 MPP 查询处理单元执行。

HAWQ 并行度的基本单位是段实例（segment instance）。一个并行查询处理系统由服务器上的多个工作段实例组成。查询提交给 HAWQ 后被优化，然后被分解为更小的组件，并被分派到一起工作的段实例，并生成单个结果集。所有关联操作(如表扫描、连接、聚合和排序)同时在各段并行执行。

因为 HAWQ 基于 Hadoop 分布式存储，所以不存在单点故障，支持全自动在线恢复。系统状态会持续受到监控，因此如果一个段失败，将自动从集群中删除。在此过程中，系统能继续为客户

提供查询服务，并且可以在必要时将这些段添加回系统。

Apache HAWQ 官方网址：http://hawq.apache.org/

Apache Pulsar：下一代分布式消息系统

2018年09月25日，Apache Pulsar 成功晋升成 TLP 的，参见这里

Apache Pulsar 是一个企业级的发布订阅（pub-sub）消息系统，最初由Yahoo开发，并于2016年底开源，现在是Apache软件基金会的一个孵化器项目。Pulsar在Yahoo的生产环境运行了三年多，助力Yahoo的主要应用，如Yahoo Mail、Yahoo Finance、Yahoo Sports、Flickr、Gemini广告平台和Yahoo分布式键值存储系统Sherpa。

向Pulsar发送数据的应用程序叫作生产者（producer），而从Pulsar读取数据的应用程序叫作消费者（consumer）。有时候消费者也被叫作订阅者。主题（topic）是Pulsar的核心资源，一个主题可以被看成是一个通道，消费者向这个通道发送数据，消费者从这个通道拉取数据。

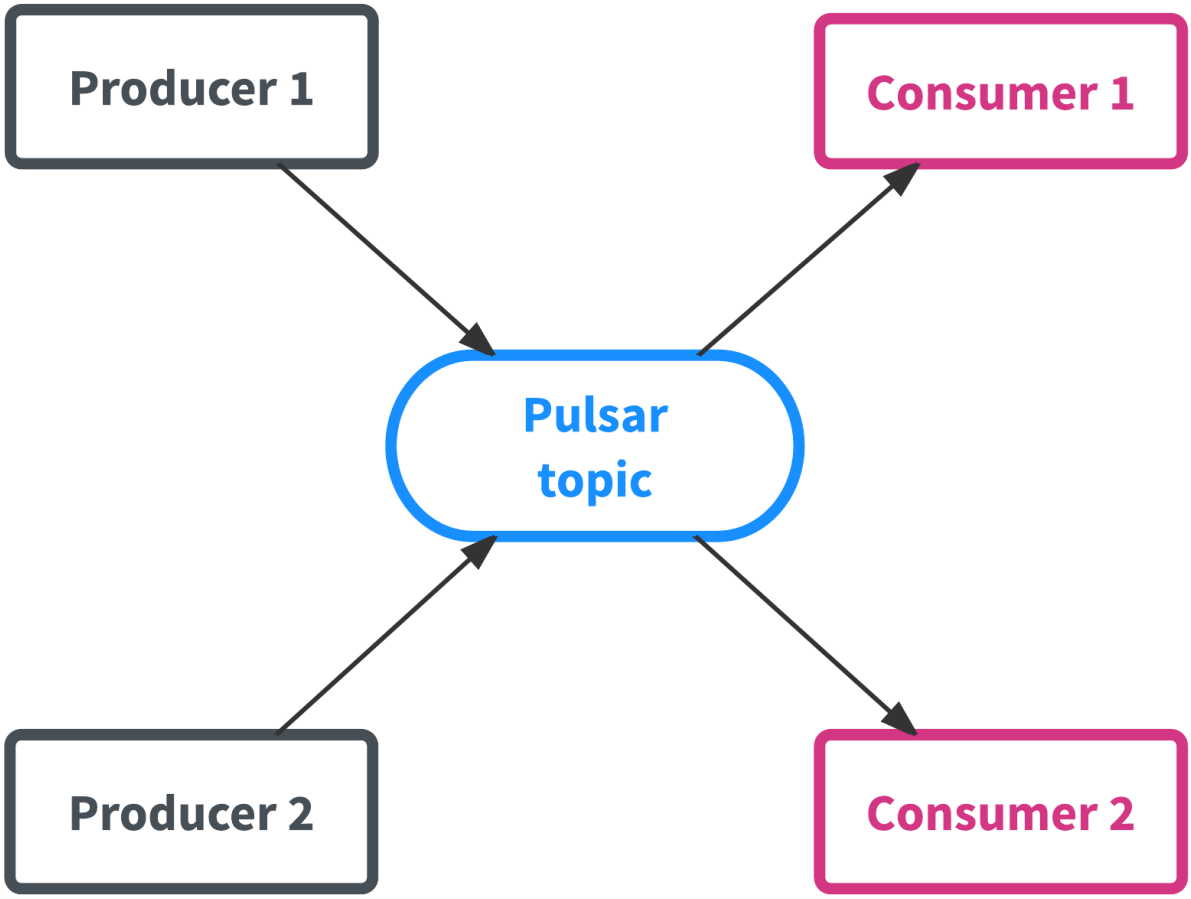


图1：生产者、消费者和主题

构建Pulsar的目的是为了支持多租户（multi-tenant）应用场景。Pulsar的多租户机制包含了两种资源：资产（property）和命名空间（namespace）。资产代表系统里的租户。假设有一个Pulsar集群用于支持多个应用程序（就像Yahoo那样），集群里的每个资产可以代表一个组织的团队、一个核心的功能或一个产品线。一个资产可以包含多个命名空间，一个命名空间可以包含任意个主题。

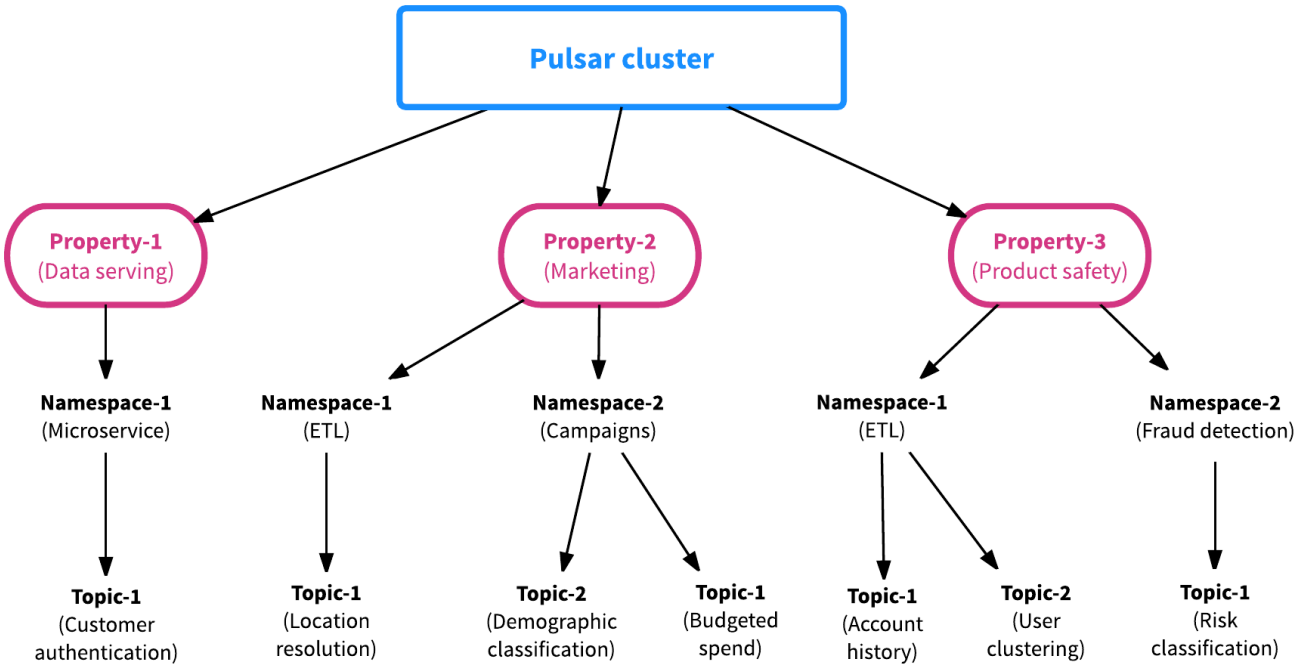


图2：Pulsar各个组件间的关系

命名空间是Pulsar最基本的管理单元。在命名空间层面，我们可以设置权限、调整复制选项、管理跨集群的数据复制、控制消息的过期时间或执行其他关键任务。命名空间里的主题会继承命名空间的配置，所以我们可以一次性对同一个命名空间内的所有主题进行配置。命名空间可以分为两种：

- 本地（local）——本地命名空间只在集群内可见。
- 全局（global）——命名空间对多个集群可见，可以是同一个数据中心内的集群，也可以是跨地域数据中心的集群。该功能取决于是否启用了集群复制功能。

虽然本地命名空间和全局命名空间的作用域不同，但它们都可以在不同的团队或不同的组织内共享。如果应用程序获得了命名空间的写入权限，就可以往该命名空间内的所有主题写入数据。如果写入的主题不存在，就会创建该主题。

每个命名空间可以包含一到多个主题，每个主题可以有多个订阅者，每个订阅者可以接收所有发布到该主题的消息。为了给应用程序提供更大的灵活性，Pulsar提供了三种订阅类型，它们可以共存在同一个主题上：

- 独享（exclusive）订阅——同时只能有一个消费者。
- 共享（shared）订阅——可以由多个消费者订阅，每个消费者接收其中的一部分消息。
- 失效备援（failover）订阅
  ——允许多个消费者连接到同一个主题上，但只有一个消费者能够接收消息。只有在当前消费者发生失效时，其他消费者才开始接收消息。

图3展示了这三种类型的订阅。Pulsar的订阅机制解耦了消息的生产者和消费者，在不增加复杂性和开发工作量的情况下为应用程序提供了更大的弹性。

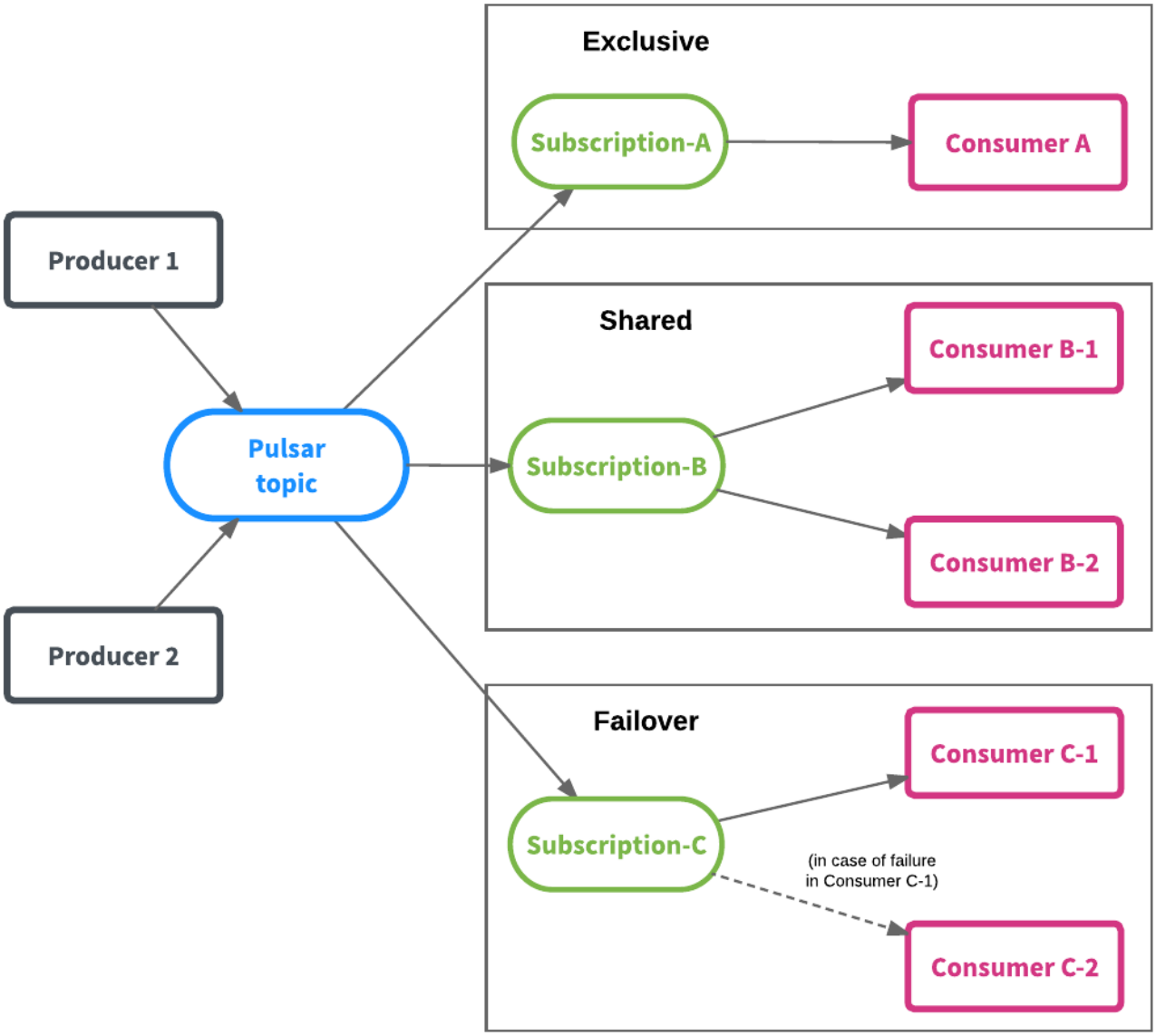


图3：不同类型的Pulsar订阅

Apache Pulsar官方网址：http://pulsar.apache.org/

Apache Griffin ： 分布式系统的数据质量解决方案

2018年12月12日，Apache Griffin 成功晋升成 TLP 的，参见这里

Apache Griffin 是开源的大数据数据质量解决方案，支持批处理和流模式，其是基于 Apache Hadoop 和 Apache Spark 构建，由 eBay 开发，并于 2016年12月07日进入 Apache 孵化。Griffin 提供了一个可以处理不同的任务，如定义数据质量模型，执行数据质量测量，自动化数据分析和验证，以及跨多个数据系统的统一数据质量可视化的全面的框架，旨在解决大数据应用中数据质量领域的挑战。

在eBay，当人们在处理大数据(Hadoop或者其它streaming系统)的时候，数据质量的检测是一个

挑战。不同的团队开发了他们自己的工具在其专业领域检测和分析数据质量问题。于是我们希望能建立一个普遍适用的平台，提供共享基础设施和通用的特性来解决常见的数据质量问题，以此得到可信度高的数据。

目前来说，当数据量达到一定程度并且有跨越多个平台时（streaming数据和batch数据），数据数量验证将是十分费时费力的。拿eBay的实时个性化平台举个例子，每天我们都要处理大约600 M的数据，而在如此复杂的环境和庞大的规模中，数据质量问题则成了一个很大的挑战。

在eBay的数据处理中，发现存在着如下问题：

- 当数据从不同的数据源流向不同的应用系统的时候，缺少端到端的统一视图来追踪数据沿袭(Data Lineage)。这也就导致了在识别和解决数据质量问题上要花费许多不必要的时间。
- 缺少一个实时的数据质量检测系统。我们需要这样一个系统：数据资产(Data Asset)注册，数据质量模型定义，数据质量结果可视化、可监控，当检测到问题时，可以及时发出警报。
- 缺乏一个共享平台和API服务，让每个项目组无需维护自己的软硬件环境就能解决常见的数据质量问题。

为了解决以上种种问题，Griffin 平台从而诞生了。其主要包含以下特性：

- 精确度检测：验证结果集数据是否与源数据是一致的
- 数据剖析：利用数据集的一致性、独特性和逻辑性，来进行统计分析和数值评估。
- 异常监测：利用预先设定的算法，检测出不符合预期的数据
- 可视化监测：利用控制面板来展现数据质量的状态。
- 实时性：可以实时进行数据质量检测，能够及时发现问题。
- 可扩展性：可以用于多个数据系统。
- 可伸缩性：工作在大数据量的环境中，目前运行的数据量约1.2PB(eBay环境)。
- 自助服务：Griffin提供了一个简洁易用的用户界面，可以管理数据资产和数据质量规则；同时用户可以通过控制面板查看数据质量结果和自定义显示内容。

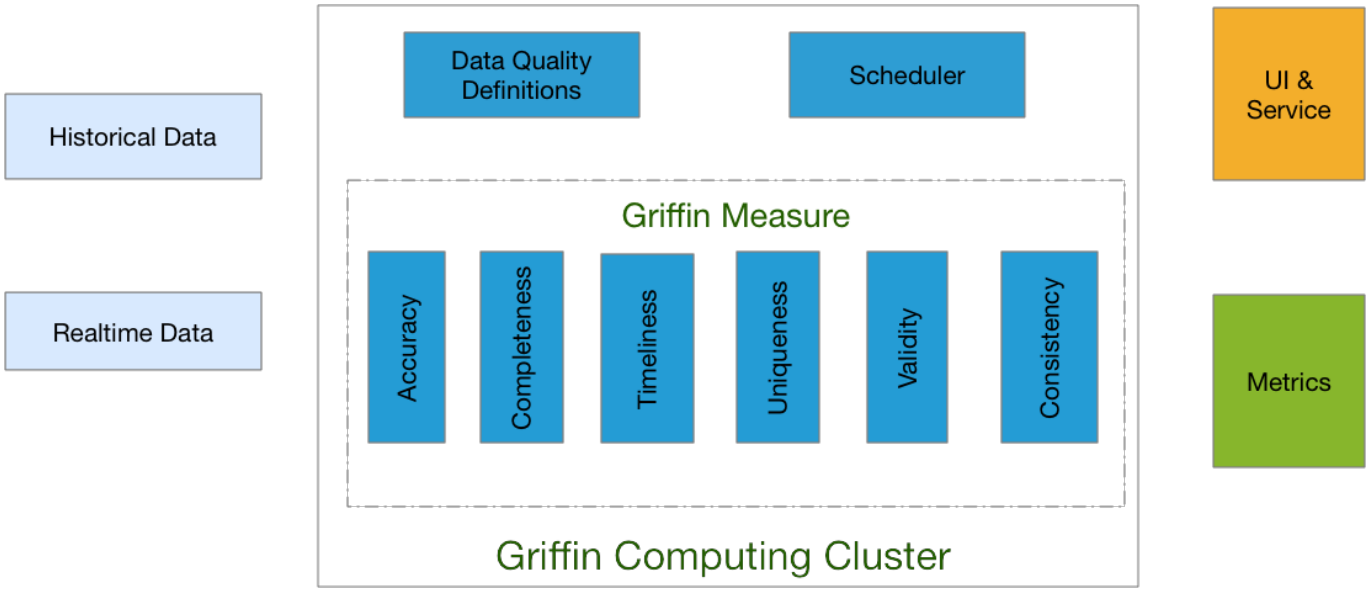
Apache Griffin 主要包含三层：数据收集处理层（Data Collection&Processing Layer）、后端服务层（Backend Service Layer）和用户界面（User Interface）。
数据收集处理层
在这一层，最关键的是模型引擎(Model Engine)，Griffin是模型驱动的解决方案。基于目标数据集（targetdata-set）或者源数据集（作为高真的基准数据源 –“golden reference data”），用户可以选择不同的数据质量维度来执行目标数据质量验证。我们有内置的程序库来支持以下检测方式：我们支持两种类型的数据源，batch数据和streaming数据。对于batch数据，我们可以通过数据连接器从Hadoop平台收集数据。对于streaming数据，我们可以连接到诸如Kafka之类的消息系统来做近似实时数据分析。在拿到数据之后，模型引擎将在我们的spark集群中计算数据质量。

后端服务层
在这一层上，有三个关键组件：

- 核心服务：用来管理元数据，例如模型定义、订阅管理和用户定制等等。

- 作业调度：根据模型的定义创建并调度作业，然后触发模型引擎的运行并取得度量值结果，然后存储度量值，并在检测到数据质量问题时发送电子邮件通知。
- REST服务：我们提供了内置的REST服务来实现Griffin的各项功能，例如注册数据资产，创建数据质量模型，度量发布，度量检索，添加订阅等等。因此，开发人员可以基于这些web服务开发自己的用户界面。

用户界面
Griffin有一个内置的可视化工具，它是基于AngularJS和eCharts开发的web前端应用，可以很好地展现数据质量结果。
架构如下：



如果想及时了
解Spark、Hadoop或者Hbase相关的文章，欢迎关注微信公共帐号：iteblog_hadoop

本博客文章除特别声明，全部都是原创！
原创文章版权归过往记忆大数据（过往记忆）所有，未经许可不得转载。
本文链接: 【】（）