ICS 35.020.
CCS L 70

DB23

黑　龙　江　省　地　方　标　准

DB23/T XXXX—XXXX

大数据平台数据接入规范

（征求意见稿）

联系人：李璐昆

联系电话：18945666832

联系邮箱：lilukun@cnhlj.cn

XXXX－XX－XX 发布　　XXXX－XX－XX 实施

黑龙江省市场监督管理局　发布

前　言

本文件依据GB/T 1.1-2020《标准化工作导则　第1部分：标准化文件的结构和起草规则》的规定起草。

请注意本文件的某些内容可能涉及专利，本文件的发布机构不承担识别专利的责任。

本文件由黑龙江省大数据产业协会提出。

本文件由黑龙江省工业和信息化厅归口。

本文件起草单位：哈尔滨财富通科技发展有限公司、黑龙江省大数据产业协会、黑龙江亿林网络股份有限公司、黑龙江省网络空间研究中心、黑龙江省标准化研究院、黑龙江大数据产业发展有限公司、黑龙江省信创科技有限公司、黑龙江交投信科科技有限责任公司、黑龙江交投千方科技有限公司、哈尔滨智路开发有限公司、黑龙江农投大数据公司、黑龙江省农投云产业有限公司。

本文件主要起草人：孙传友、杜飞、李璐昆、孙甲子、张驰、王磊、陈要武、杨大志、吕猛、唐丽、赵海洋、李冰冷、叶爽、王克云、李森、周全、何晨龙。

大数据平台数据接入规范

1 范围

本文件规定了大数据平台与各数据提供单位管理支撑系统进行数据接入的技术要求及数据采集接口、方式。

本文件适用于黑龙江省大数据平台进行数据采集功能研发、数据采集工具选型及其数据接入场景提供规范要求。

2 规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅该日期对应的版本适用于本文件；不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 29262 信息技术 面向服务的体系结构（SOA）术语

GB/T 35295 信息技术 大数据 术语

DB15/T 1872 大数据平台接入技术要求

3 术语和定义

GB/T 35295和GB/T 29262界定的以及下列术语和定义适用于本文件。

3.1

数据采集接入

将数据传输进入数据中心所采用的形式。

4 缩略语

下列缩略语适用于本文件。

FTP：标准的文件传输协议（File Transfer Protocol）

JDBC：java数据库连接（Java DataBase Connectivity）

HTTPS：超文本传输安全协议（Hyper Text Transfer Protocol over Secure Socket Layer或Hypertext Transfer Protocol Secure）

DTS:数据传输服务(Data Transfer Service)

ETL:将数据从来源端经过抽取、转换、加载至目的端的过程(Extract-Transform-Load)

NFS:是基于UDP/IP协议的应用，其实现主要是采用远程过程调用RPC机制，RPC提供了一组与机器、操作系统以及低层传送协议无关的存取远程文件的操作(Network File System)

5 概述

5.1 总体采集框架图

大数据平台支持从关系型数据库、文件、数据流等来源采集数据，实现各类离线数据及实时数据的采集与接入，包括设备采集数据、企业管理业务数据、外部数据等。其中离线数据主要分为关系型数据库所存储的结构化数据及文件系统所存储的非结构化文件数据，实时数据主要是设备采集监控及业务系统产生的实时流数据。其中关系数据库中的结构化数据可通过关系数据库抽取、实时数据库复制及自定义数据网管服务接口程序等方式实现数据接入；非结构化文件数据可通过源端FTP服务拉取及NFS服务拉取数据方式、目标端FTP服务推送及NFS服务推送方式、自定义数据网关接口程序、自定义webservice服务程序等方式实现数据接入；实时流数据主要通过往分布式消息队列推送数据的方式进行实时接入。总体采集框架图见图1。

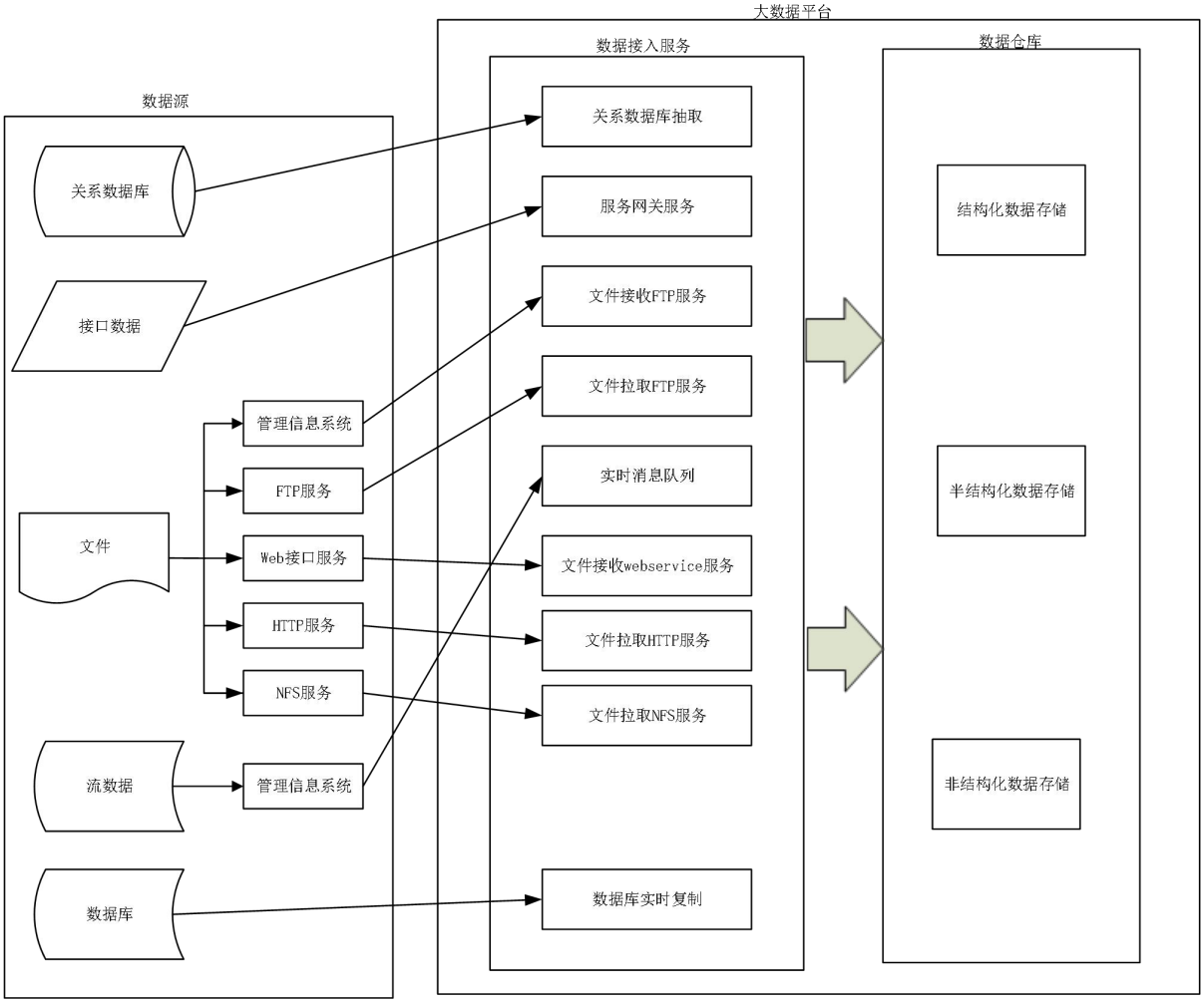

图 1　数据接入总体框架

5.2　数据源的接入方式

大数据平台作为多维数据的处理平台，支持各类数据源的采集与接入。以下是大数据平台九种常用数据源的接入方式：

a)　关系数据库抽取；
b)　服务/数据网关服务；
c)　实时消息队列；

d) 文件接收 FTP 服务；
e) 文件接收 webservice 服务；
f) 文件拉取 HTTP 服务；
g) 文件拉取 FTP 服务；
h) 文件拉取 NFS 服务；
i) 数据库实时复制。

6 接入要求

6.1 关系数据库抽取

6.1.1 功能要求

关系数据库抽取应提供管理信息系统关系数据库中的结构化数据到大数据平台数据存储的定时批量抽取功能。关系数据库数据抽取应具备以下主要功能：

a) 支持对主流的关系数据库进行数据抽取；支持对数据库中常用的数据类型进行数据抽取，至少包括数值型、字符型、日期/时间型等数据类型；
b) 支持“全量”和“增量”两种数据抽取模式；“全量”模式是指一次性将关系数据库中物理表的数据抽取到大数据平台。“增量”模式是指根据设置的抽取条件筛选符合条件的数据抽取到大数据平台；
c) 支持关系数据库中结构化数据抽取到大数据平台，包含结构化数据存储、半结构数据存储、非结构数据存储的数据仓库中；
d) 支持对关系数据库数据的采集内容和类型转换操作，至少包括选择具体的数据表、选择表中具体的字段、字段类型格式转换等操作；
e) 支持数据抽取操作的立即执行、定时调度运行。定时调度运行应提供多种调度策略，至少包括固定时间间隔运行、指定时间点运行、指定时间范围运行、一次或指定次数运行等策略；
f) 应提供图形化管理界面，应提供数据抽取模式设置、抽取源关系数据库配置、指定数据表配置、表字段选择配置、字段类型转换配置、大数据平台目标存储位置配置、运行策略配置等操作界面；
g) 应提供完善的日志和审计能力，可以记录数据抽取操作配置、运行时发生的各种事件；
h) 应提供完善的监控机制，运行过程中出现异常可快速的定位及解决。

6.1.2 应用场景

6.1.2.1 场景图

关系数据库抽取应用场景见图2。

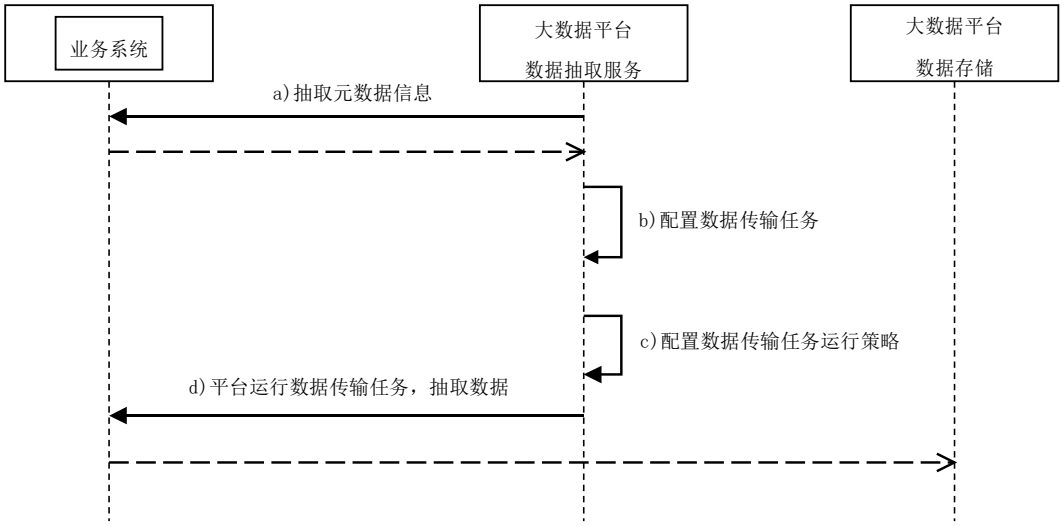

图 2　关系数据库抽取应用场景

6.1.2.2　应用场景描述

应用场景描述如下:

a)　关系数据库抽取服务，抽取数据源中数据库的元数据信息，包括数据库用户所属的表、字段信息；

b)　关系数据库抽取服务配置数据传输任务，包括抽取数据库的源表和大数据平台对应的存储目标表；

c)　关系数据库抽取服务配置数据传输任务运行策略，包括运行的开始时间、结束时间、运行频度；

d)　关系数据库抽取服务运行数据传输任务，从数据源的数据库抽取数据到平台数据存储中。

6.1.3　应用要求

应用要求如下：

a)　关系数据库抽取服务适用于关系数据库定时批量抽取场景，详细说明参见附录 A；

b)　数据源须提供关系数据库的访问链接，包括 IP、端口、数据库实例名、用户名、密码；

c)　数据源提供的数据库访问用户应具备数据库的元数据信息定义表的读取权限。

6.2　服务网关服务

6.2.1　功能要求

服务网关服务为数据源提供大数据平台中结构化数据或非结构化数据的接口数据接入。服务网关服务应具备以下主要功能：

a)　支持接入 Webservice、RESTful 方式的接口；

b)　支持包括结构化数据、非结构化数据的接口；

c)　支持接口编排，轻松实现多个接口的功能集成；

d)　提供图形化管理界面，用于接口数据存储位置、操作用户、目标存储位置的配置；

e)　应提供完善的日志和审计能力，可以记录接口数据配置及数据抽取操作配置、运行时发生的各种事件；

f)　应具备熔断管理机制，保证服务整体可用，是接口访问异常情况下的处理策略。

6.2.2 应用场景

6.2.2.1 场景图

服务网关服务应用场景见图3。

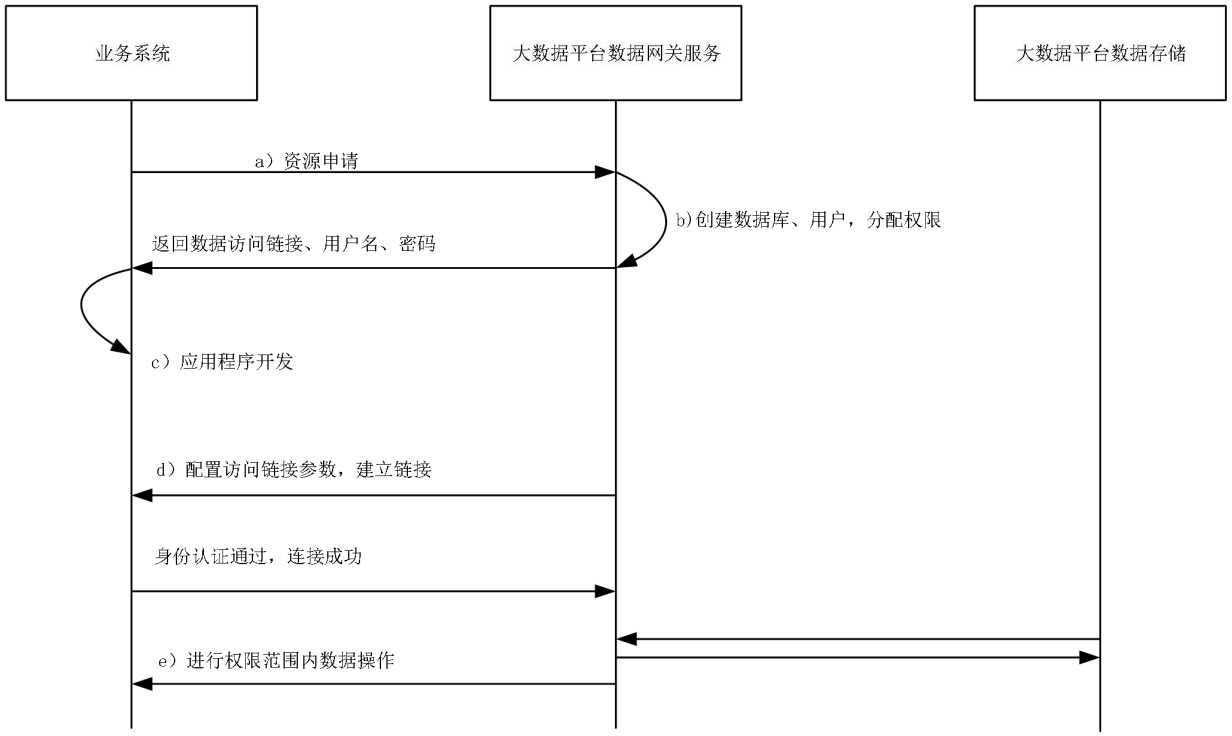

图 3 服务网关服务应用场景

6.2.2.2 应用场景描述

应用场景描述如下:

a) 数据源向大数据平台提供接口信息，包括：接口访问地址、输入参数、输出参数、验证方式等接口信息；
b) 大数据平台根据数据源提供的数据接口进行定义及编排；
c) 服务网关服务配置数据传输任务运行策略，包括运行的开始时间、结束时间、运行频度；
d) 服务网关服务运行数据传输任务，从数据源的数据接口中抽取数据到大数据平台数据仓库中。

6.2.3 应用要求

应用要求包括：

a) 服务网关服务适用于提供接口类数据的数据源，详细说明参见附录 B；
b) 提供数据接口的数据源需做好自身数据操作接口程序的开发。

6.3 实时消息队列

6.3.1 功能要求

实时消息队列采集为管理信息系统提供实时消息推入和缓存功能。实时消息队列应具备以下主要功能：

a) 应提供分布式消息队列的管理功能，支持消息主题的创建、删除、修改；
b) 应提供支持“点对点”和“发布-订阅”两个消息模式；
c) 支持消息的持久化存储操作并且支持持久化周期设置；
d) 应提供消息的发送和消费接口，包括链接建立、消息发送、消息消费、链接关闭；
e) 应提供分布式高可用的消息队列操作接口，支持消息的发送和消费；支持消息分区和备份操作；
f) 具有风格统一的图形化管理界面，支持消息队列主题的创建、删除、测试、授权访问的操作；
g) 具备完善的日志审计能力，可以记录消息发送和消费时发生的各种事件。

6.3.2 应用场景

6.3.2.1 场景图

实时消息队列应用场景见图4。

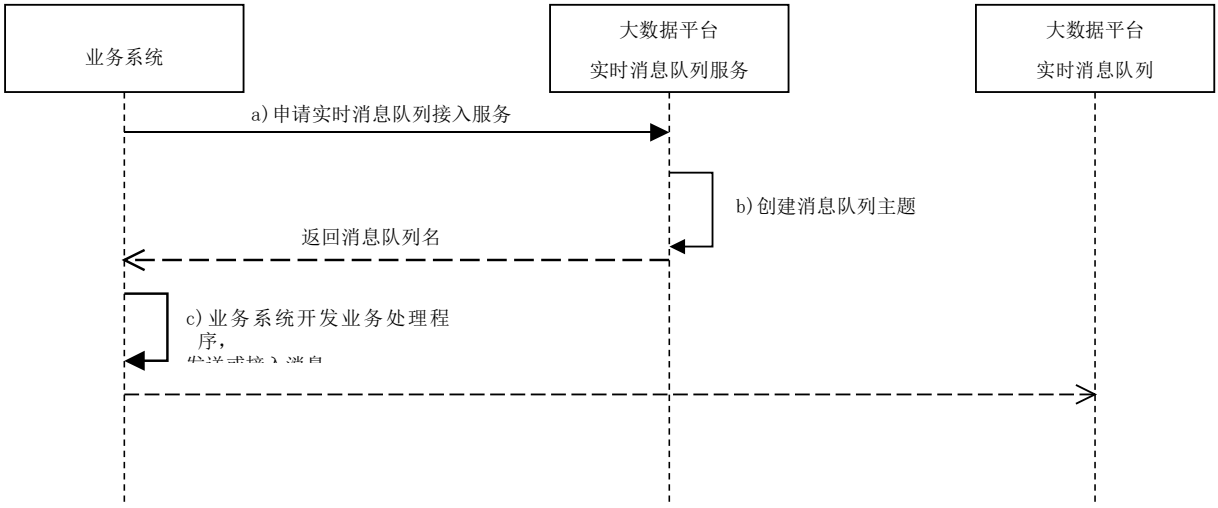

图 4 实时消息队列应用场景

6.3.2.2 应用场景描述

应用场景描述如下：

a) 管理信息系统须向大数据平台申请实时消息队列接入服务；
b) 大数据平台根据申请创建消息队列主题，返回消息队列名称；
c) 管理信息系统开发业务处理程序，调用平台实时消息队列接口，发送数据或接收数据。

6.3.3 应用要求

应用要求如下：

a) 实时消息队列采集适用于管理信息系统主动将数据封装为消息，发送到大数据平台的实时消息队列中。基于实时消息队列的消息缓存进行数据分析，如流计算实时处理等，详细说明参见附录 C；
b) 发送的消息内容格式支持字符串，发送的数据对象可以通过对象序列化机制转换为字符串格式的消息内容；

c) 管理信息系统应依照大数据平台提供的实时消息队列采集接口完成自身数据发送或接收接口的开发。

6.4 文件接收 FTP 服务

6.4.1 功能要求

文件接收FTP服务应提供外部系统文件数据的接收并存入到大数据平台数据仓库的功能。文件采集应具备以下主要功能：

a) 支持标准 FTP 协议接收数据；
b) 支持顺序型断点续传功能；
c) 支持接收的文件的重命名及指定存储目录；
d) 应支持对接收文件的完整性校验；
e) 应支持对客户端进行认证；
f) 支持图形管理功能，支持认证配置、文件目标位置配置、校验处理配置。

6.4.2 应用场景

6.4.2.1 场景图

文件接收FTP服务应用场景见图5。

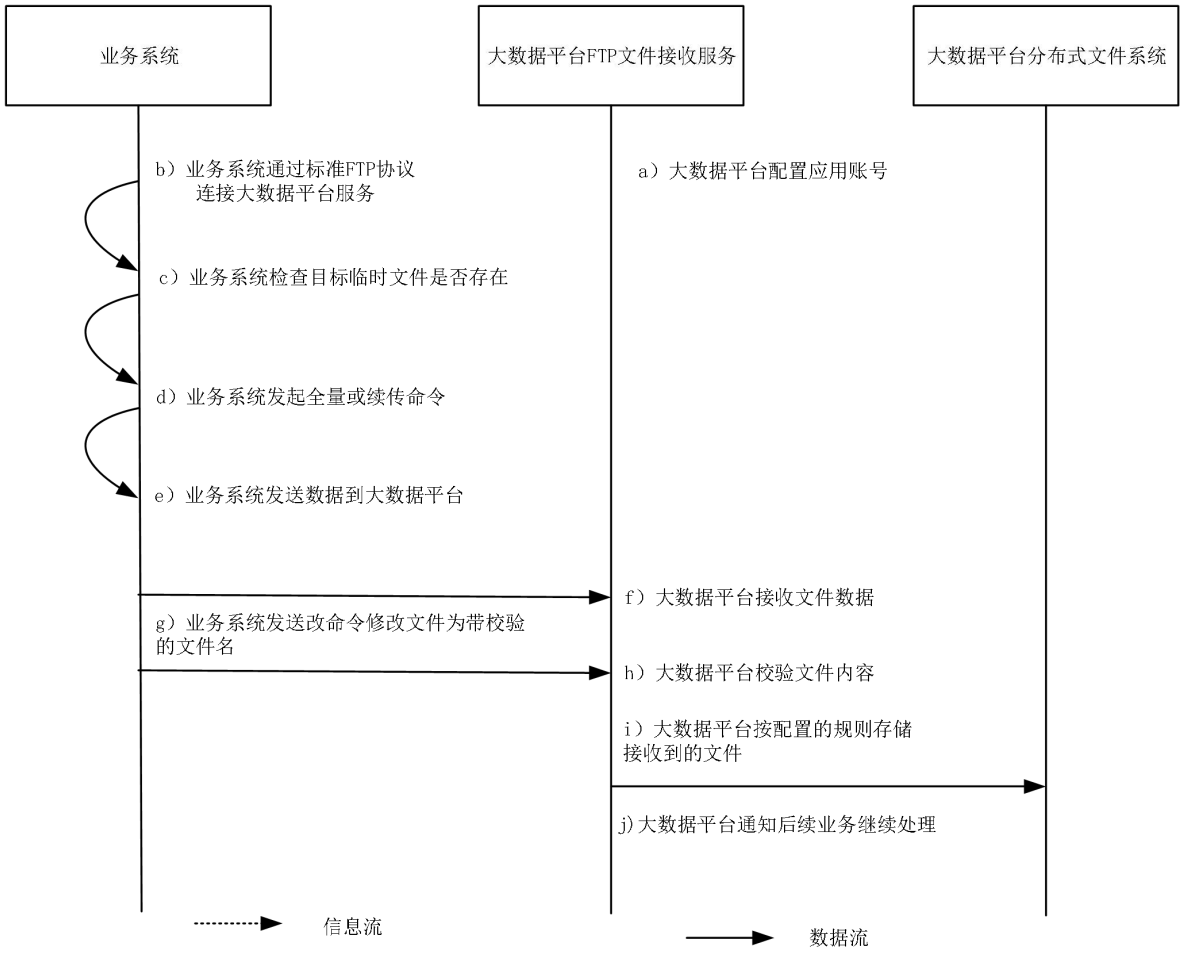

图 5　文件接收 FTP 服务应用场景

6.4.2.2　应用场景描述

应用场景描述如下：

a)　大数据平台配置应用账号、接收参数及存储位置；
b)　管理信息系统通过标准 FTP 协议连接大数据平台服务；
c)　管理信息系统检查目标临时文件是否存在；
d)　管理信息系统发起全量或续传指令；
e)　管理信息系统发送数据到大数据平台；
f)　大数据平台接收文件数据；
g)　管理信息系统发送数据校验文件；
h)　大数据平台根据校验文件校验数据文件内容；
i)　大数据平台按配置的规则存储接收到的数据文件；
j)　大数据平台回写数据存储状态；
k)　管理信息系统获取数据存储状态。

6.4.3　应用要求

应用要求如下：

a)　管理信息系统须在大数据平台注册并申请账号；
b)　管理信息系统须按平台协议规范开发上传功能；
c)　管理信息系统生成文件数据时须同时生成对应的完整性校验码；
d)　具体文件接收 FTP 服务 API 接口参见附录 D。

6.5　文件接收 web service 服务

6.5.1　功能要求

文件接收webservice服务应提供外部系统非结构化文件的接收并存入到大数据平台分布式文件系统功能。文件采集应具备以下主要功能：

a)　支持标准 webservice 协议接收数据；
b)　支持顺序型断点续传功能；
c)　支持接收的文件的重命名及制定存储目录；
d)　应支持对接收文件的完整性校验；
e)　应支持对客户端进行认证；
f)　支持图形管理功能，支持认证配置、文件目标位置配置、校验处理配置。

6.5.2　应用场景

6.5.2.1　场景图

文件接收webservice服务应用过程见图6。

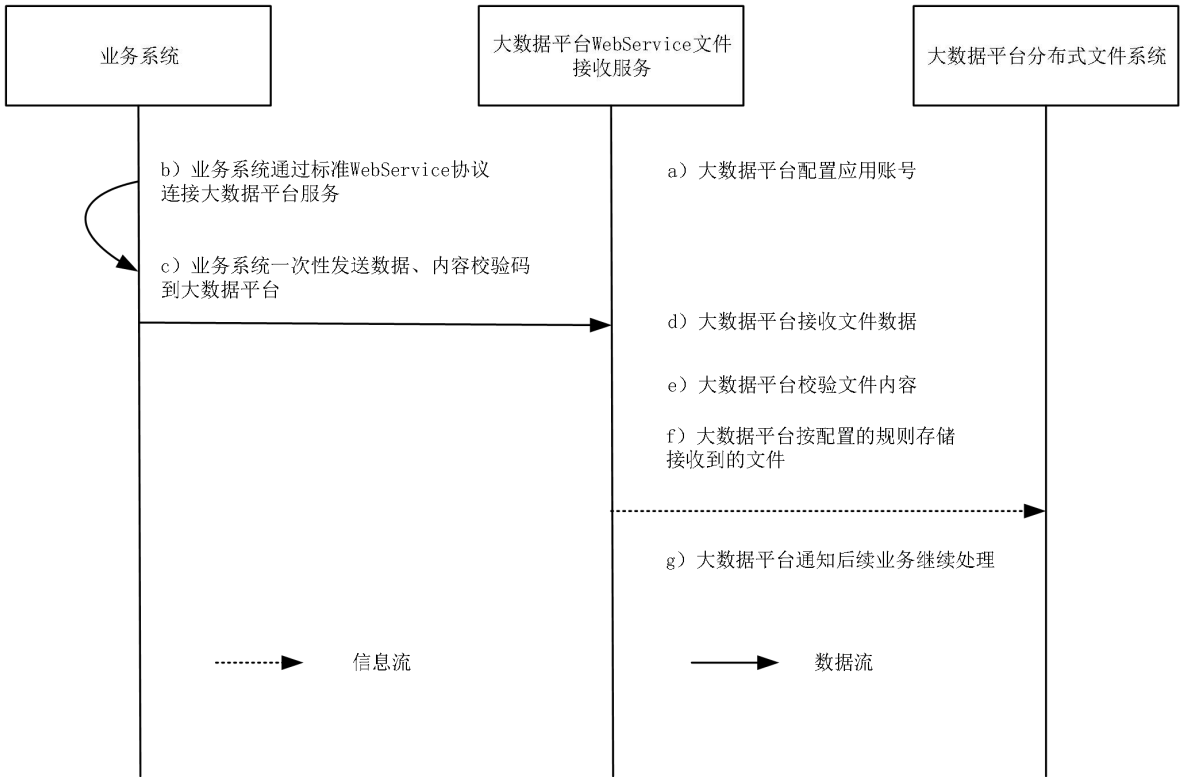

图 6　文件接收 WebService 服务应用场景

6.5.2.2　应用场景描述

应用场景描述如下：

a)　大数据平台配置应用账号、接收参数及存储位置；
b)　业务系统通过标准 webservice 协议连接大数据平台服务；
c)　业务系统一次性发送数据、内容校验码到大数据平台；
d)　大数据平台接收文件数据；
e)　大数据平台校验文件内容；
f)　大数据平台按配置的规则存储接收到的文件；
g)　大数据平台通知后续业务继续处理。

6.5.3　应用要求

应用要求包括：

a)　业务系统须在大数据平台注册并申请账号；
b)　业务系统必须按大数据平台加密方式进行登录加密；
c)　业务系统须按平台 webservice 规范调用服务；
d)　业务系统生成文件数据时须同时生成对应的完整性校验码；
e)　文件大小必须小于 2M；
f)　文件数据校验算法须支持标注的 SHA1 进行校验，结果转化为 16 进制 ASCII 字符表示；
g)　具体文件接收 webservice 服务 API 服务接口参见附录 E。

6.6　文件拉取 HTTP 服务

6.6.1 功能要求

文件拉取HTTP服务，应提供通过访问HTTP协议实现将文件信息抽取到大数据平台存储的功能。文件拉取HTTP服务应具备以下主要功能：

a) 支持顺序型断点续传功能，支持外部文件存储断点续传能力的自动识别及模式匹配；
b) 支持文件压缩传输，提供文件压缩传输规则设置；
c) 支持文件加密传输，提供文件加密传输规则设置；
d) 支持设置文件同步、异步拉取，支持设置拉取并行度；
e) 支持制定目标文件存储位置、文件名，提供文件类型转换规则，支持常见类型转换；
f) 支持全量文件采集，支持外部数据一次性初始化导入；
g) 支持定时轮询文件采集，采集新增的文件，支持文件列表规则过滤；
h) 支持图形管理功能，支持文件源配置、文件目标存储配置、文件压缩和加密传输规则配置、文件同步/异步传输规则配置、文件传输并行度配置、文件定时及实时策略配置、文件采集过滤配置。

6.6.2 应用场景

6.6.2.1 基于 HTTP 协议的全量采集应用场景图

基于HTTP协议的全量采集应用场景见图7。

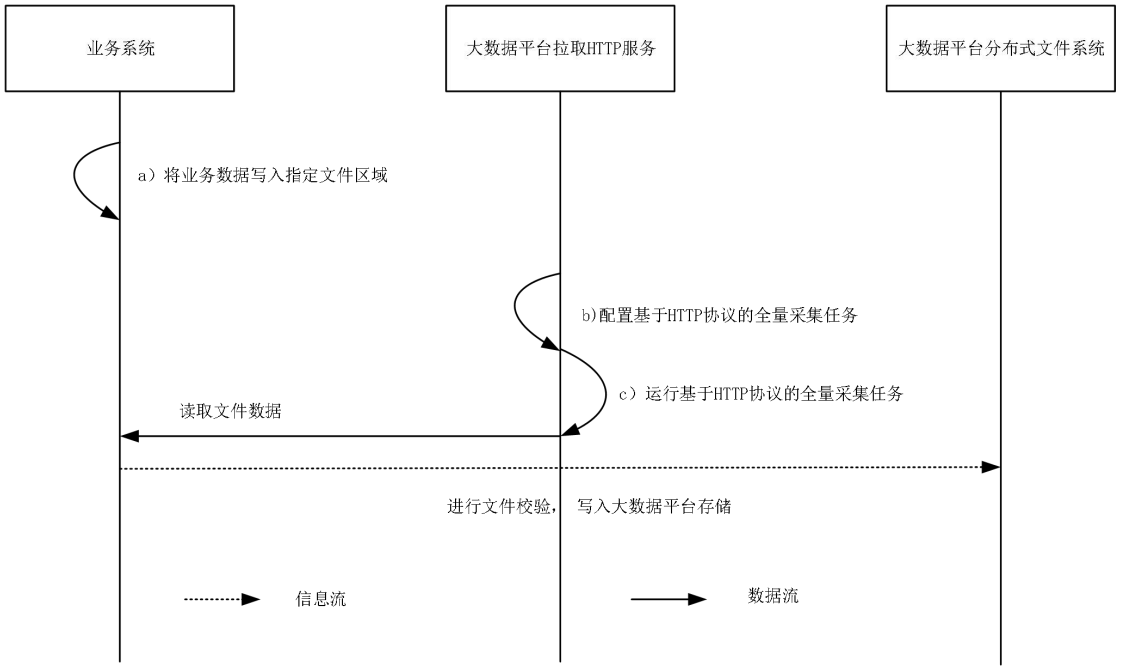

图 7 基于 HTTP 协议全量采集应用场景

6.6.2.2 基于 HTTP 协议的全量采集应用场景描述

应用场景描述如下：

a) 业务系统应提供初始文件存储位置；
b) 大数据平台全量文件采集提供基于 HTTP 协议的采集任务配置，包括：存储文件位置、采集文件列表方式、是否文件校验、传输是否压缩、加密、同步/异步、并行度、存储目标位置及文件存储类型转换和命名规则；

c） 大数据平台运行文件采集任务，读取文件存入大数据平台数据存储中。

6.6.2.3 基于 HTTP 协议的定时轮询采集应用场景图

基于HTTP协议的定时轮询采集应用场景见图8。

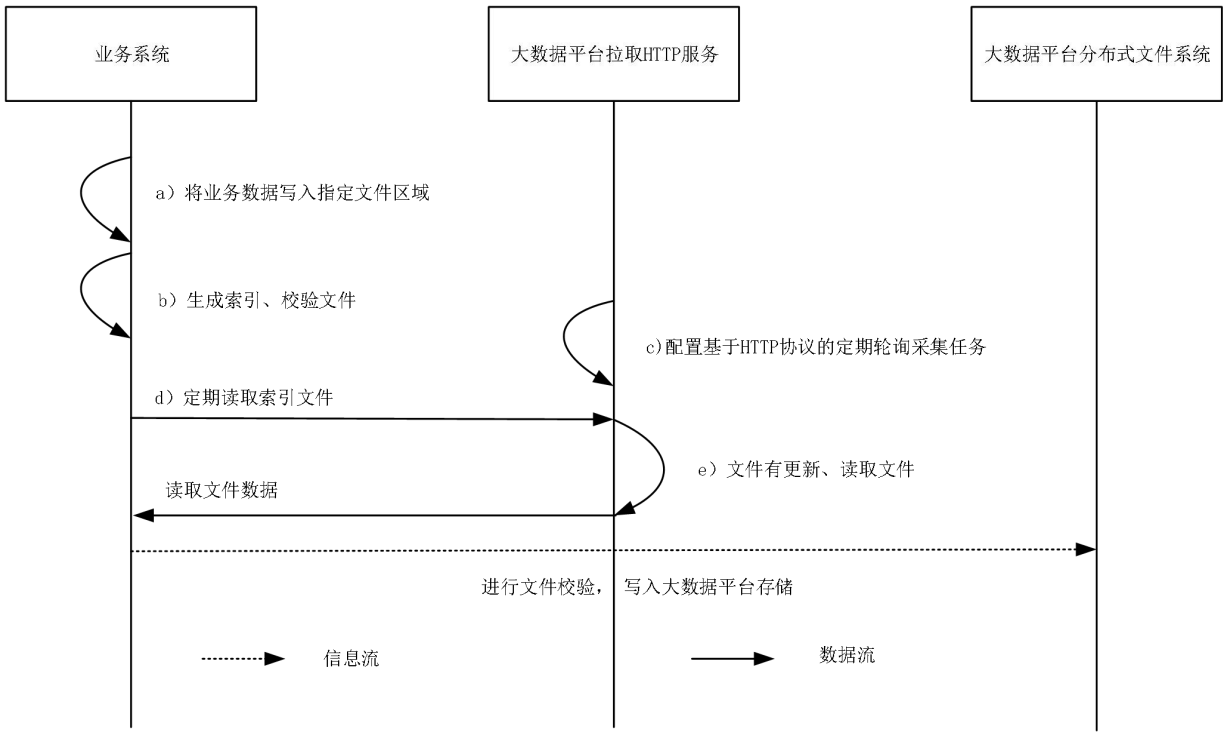

图 8 基于 HTTP 协议轮询采集应用场景

6.6.2.4 基于 HTTP 协议的定时轮询采集应用场景描述

应用场景描述如下：

a） 业务系统应按照业务需求生成数据文件，存储在指定文件区域；

b） 业务系统应根据校验算法 SHA1 在相同目录下生成校验文件，校验文件名称与数据文件相同，文件扩展名为.SHA1；

c） 大数据平台轮询文件采集服务配置基于 HTTP 协议的定期轮询采集任务，定期轮询采集文件；

d） 大数据平台定期读取解析索引及校验文件内容；如果业务系统数据文件出现新增时，大数据平台读取数据文件内容，并且进行文件校验，如果文件完整，则将数据文件写入大数据平台中。

6.6.3 应用要求

应用要求如下：

a） 业务系统须现将业务数据保存为文件；

b） 文件数据校验算法须支持标准的 SHA1 进行校验，结果转化为 16 进制 ASCII 字符表示；

c） 数据文件可通过 HTTP 协议访问；

d） 业务系统生成文件数据时须同时生成对应的完整性校验码；

e） 具体文件拉取 HTTP 服务 API 接口参见附录 F。

6.7 文件拉取 FTP 服务

6.7.1 功能要求

文件拉取FTP服务，应提供通过访问FTP协议实现将文件数据抽取到大数据平台数据仓库的功能。文件拉取FTP服务应具备以下主要功能：

a) 支持顺序型断点续传功能，支持外部文件存储断点续传能力的自动识别及模式匹配；
b) 支持 FTP 服务登录用户名和密码设置；
c) 支持文件压缩传输，提供文件压缩规则设置；
d) 支持文件加密传输，提供文件加密传输规则设置；
e) 支持设置文件同步、异步拉取，支持设置拉取并行度；
f) 支持指定目标文件存储位置、文件名，提供文件类型转换规则，支持常见类型转换；
g) 支持全量文件采集，支持外部数据一次性初始化导入；
h) 支持定时轮询文件采集，采集新增的文件，支持文件列表规则过滤；
i) 支持图形管理功能，支持 FTP 连接配置、文件源配置、文件目标存储配置、文件压缩和加密传输规则配置、文件同步/异步传输规则配置、文件传输并行度配置、文件定时及实时策略配置、文件采集过滤配置。

6.7.2 应用场景

6.7.2.1 基于 FTP 协议的全量文件采集应用场景图

基于FTP协议的全量文件采集应用场景见图9。

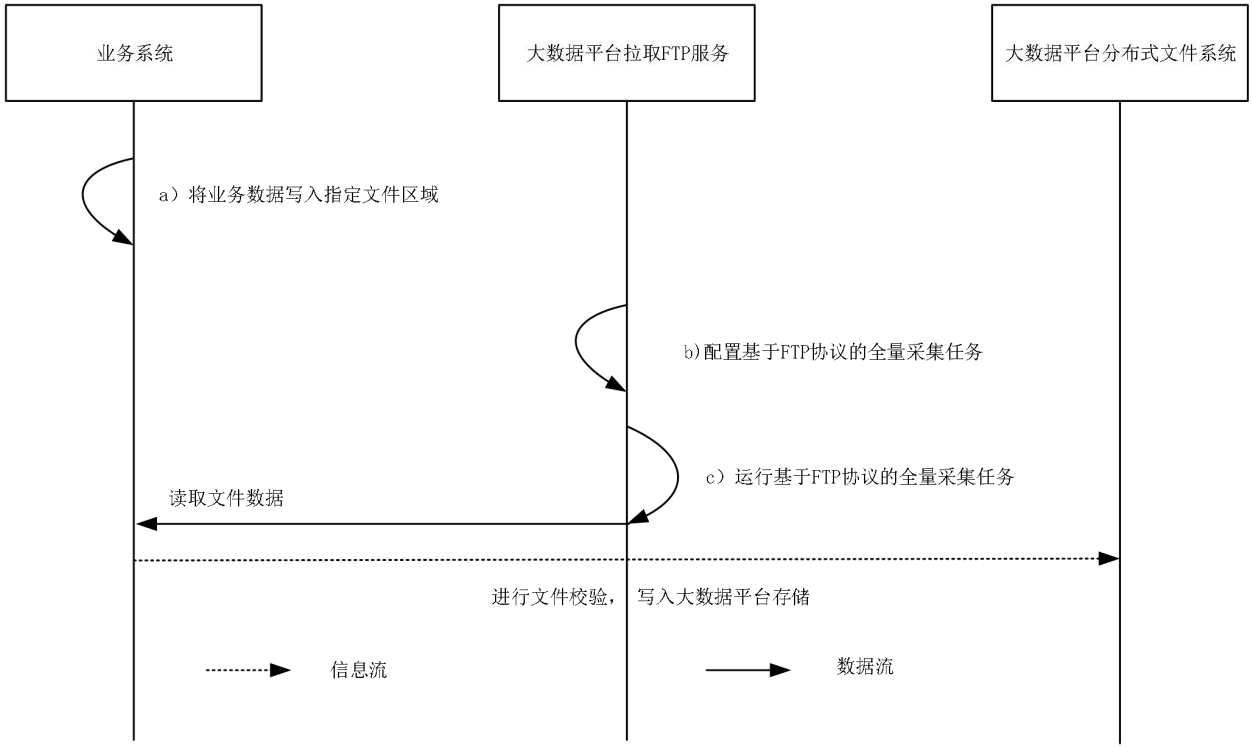

图 9 基于 FTP 协议全量采集应用场景

6.7.2.2 基于 FTP 协议的全量文件采集应用场景描述

应用场景描述如下：

a) 业务系统应提供初始文件存储位置；
b) 大数据平台全量文件采集提供基于 FTP 协议的采集任务配置，包括：存储文件位置、采集文件列表方式、是否文件校验、存储目标位置及文件存储命名规则；
c) 大数据平台运行文件采集任务，读取文件存入大数据平台数据存储中。

6.7.2.3 基于 FTP 协议的定时轮询采集应用场景图

基于FTP协议的定时轮询采集应用过程见图10。

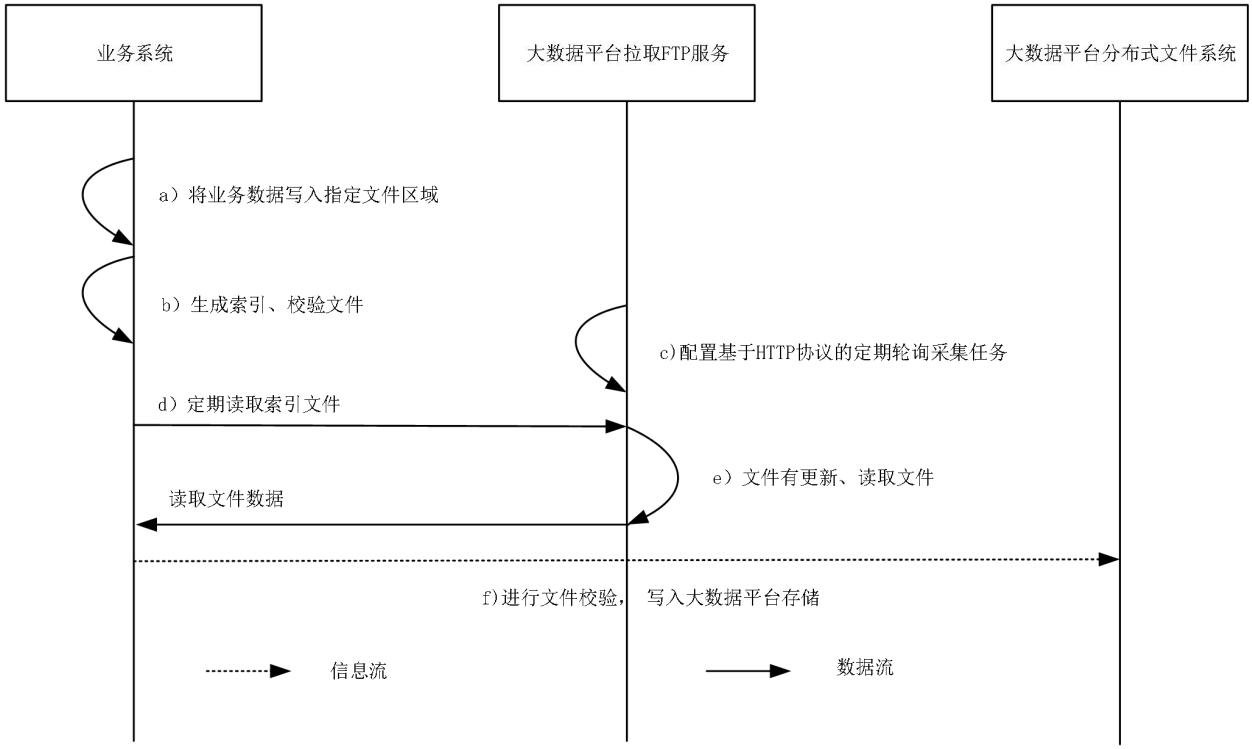

图 10 基于 FTP 协议轮询采集应用场景

6.7.2.4 基于 FTP 协议的定时轮询采集应用场景描述

应用场景描述如下：

a) 业务系统应按照业务需求生成数据文件，存储在指定文件区域；
b) 业务系统应根据校验算法 SHA1 在相同目录下生成校验文件，校验文件名称与数据文件相同，文件扩展名为.SHA1；
c) 大数据平台轮询文件采集服务配置基于 FTP 协议的定期轮询采集任务，定期轮询采集文件；
d) 大数据平台定期读取解析索引及校验文件内容；如果业务系统数据文件出现新增时，大数据平台读取数据文件内容，并且进行文件校验，如果文件完整，则将数据文件写入大数据平台中。

6.7.3 应用要求

应用要求如下：

a) 业务系统须先将业务数据保存为文件，并设置访问权限；
b) 文件数据校验算法须支持标准的 SHA1 进行校验，结果转化为 16 进制 ASCII 字符表示；

c) 数据文件可通过 FTP 协议访问；
d) 业务系统生成文件数据时须同时生成对应的完整性校验码；
e) 具体文件拉取 HTTP 服务 API 接口参见附录 G。

6.8 文件拉取 NFS 服务

6.8.1 功能要求

文件拉取NFS服务，应提供通过访问NFS文件系统，实现将文件信息抽取到大数据平台数据存储的功能。文件拉取NFS服务应具备以下主要功能：

a) 支持顺序型断点续传功能，支持外部文件存储断点续传能力的自动识别及模式匹配；
b) 支持指定目标文件存储位置、文件名，提供文件类型转换规则，支持常见类型转换；
c) 支持全量文件采集，支持外部数据一次性初始化导入；
d) 支持定时轮询文件采集， 采集新增的文件， 支持文件列表规则过滤；
e) 支持图形管理功能，支持文件源配置、文件目标存储配置、文件压缩和加密传输规则配置、文件定时及实时策略配置、文件采集过滤配置。

6.8.2 应用场景

6.8.2.1 NFS 全量文件采集应用场景图

NFS全量文件采集应用场景见图11。

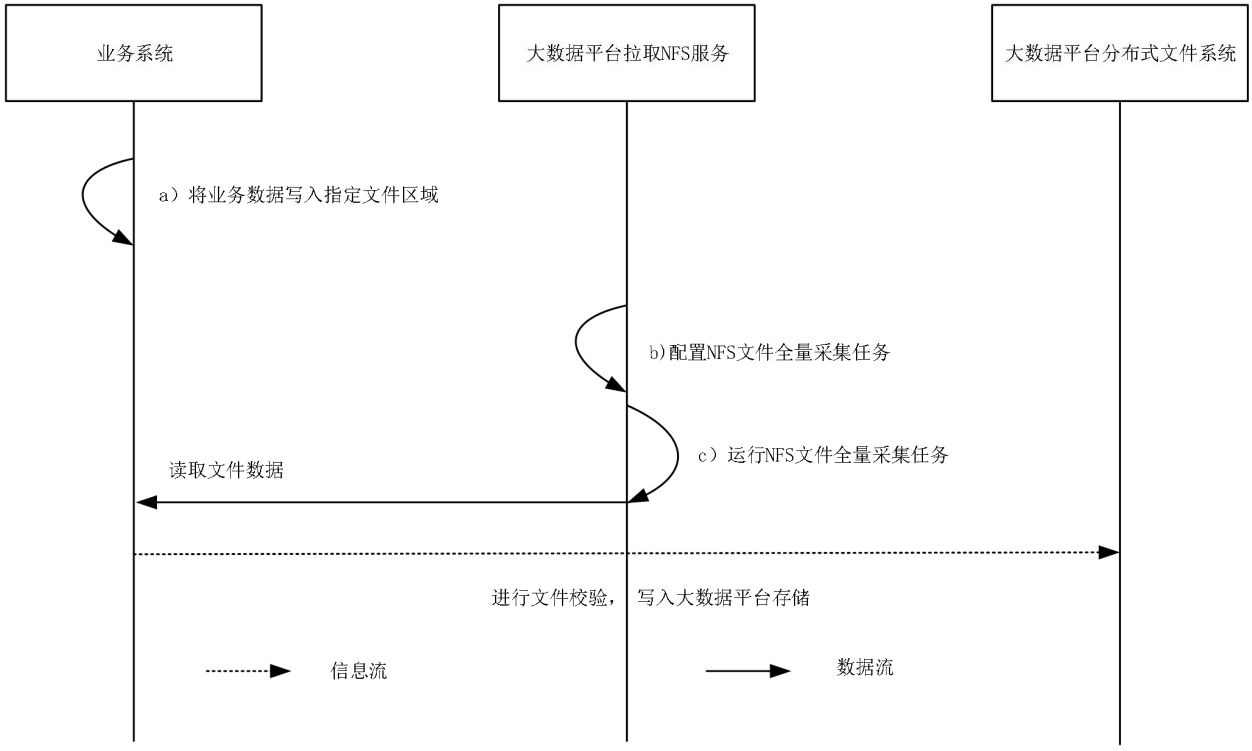

图 11 NFS 文件全量采集应用场景

6.8.2.2 NFS 全量文件采集应用场景描述

应用场景描述如下：

a) 业务系统应提供初始文件存储位置；
b) 大数据平台全量文件采集提供采集 NFS 文件任务配置，包括：存储文件位置、采集文件列表方式、是否文件校验、存储目标位置及文件存储命名规则；
c) 大数据平台运行文件采集任务，读取文件存入大数据平台数据存储中。

6.8.2.3 NFS 文件定时轮询采集应用场景图

NFS文件定时轮询采集应用场景见图12。

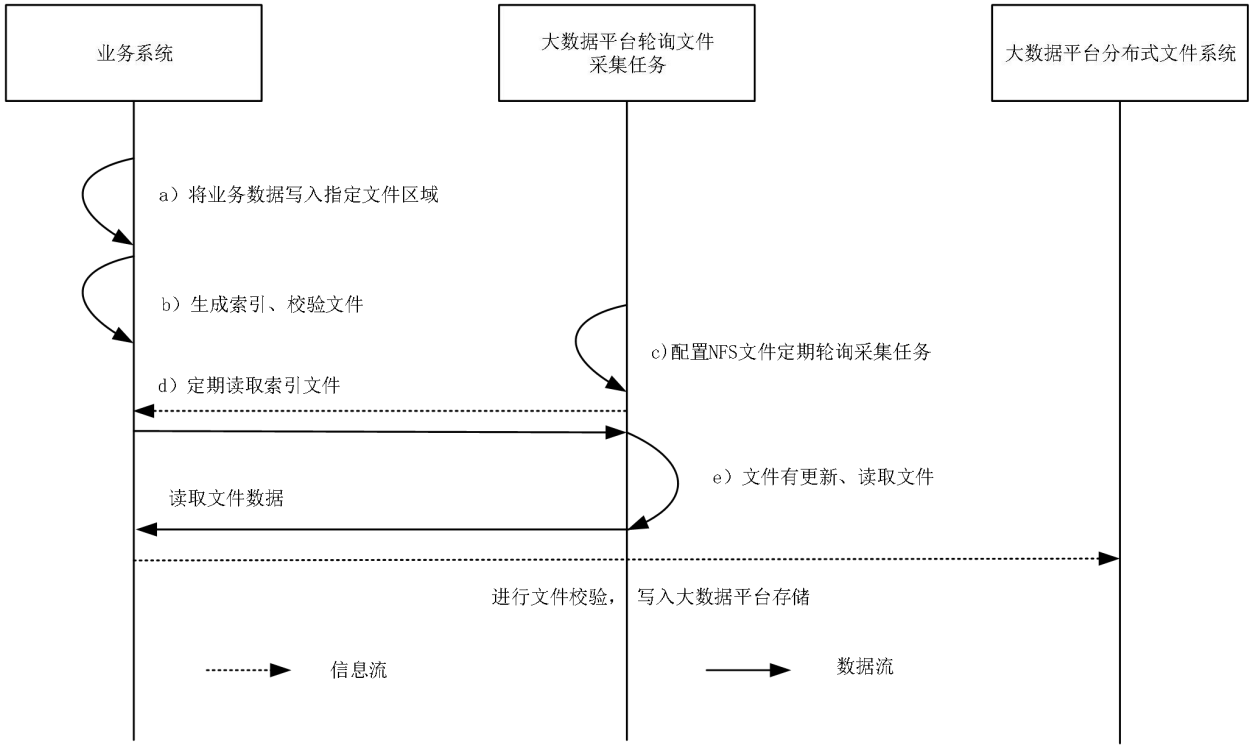

图 12 NFS 文件定时轮询采集应用场景

6.8.2.4 NFS 文件定时轮询采集应用场景描述

应用场景描述如下：

a) 业务系统应按照业务需求生成数据文件，存储在指定文件区域；
b) 业务系统应根据校验算法 SHA1 在相同目录下生成校验文件，校验文件名称与数据文件相同，文件扩展名为.SHA1；
c) 大数据平台轮询文件采集服务配置 NFS 文件定期轮询采集任务，定期轮询采集文件；
d) 大数据平台定期读取解析索引及校验文件内容；如果业务系统数据文件出现新增时，大数据平台读取数据文件内容，并且进行文件校验，如果文件完整，则将数据文件写入大数据平台中。

6.8.3 应用要求

应用要求如下：

a) 业务系统须先将业务数据保存为文件，并设置访问权限；
b) 文件数据校验算法须支持标准的 SHA1 进行校验，结果转化为 16 进制 ASCII 字符表示；
c) 数据文件可通过 NFS 访问；

d) 业务系统生成文件数据时须同时生成对应的完整性校验码；
e) 具体文件拉取 NFS 服务 API 接口参见附录 H。

6.9 数据库实时复制

6.9.1 功能要求

数据实时复制应提供业务系统关系型数据库中的结构化数据到大数据平台数据存储的增量低时延复制功能，数据实时复制应具备以下主要功能：

a) 支持对主流的关系型数据库进行低时延增量复制，至少包括 oracle、MySQL、PostgreSQL 等关系型数据库；支持对数据库中常用的数据类型进行数据抽取，至少包括数值型、字符型、日期/时间型等数据类型；
b) 支持秒级时延的关系型数据库增量复制能力；同时支持在全量复制的基础上，无缝自动切换到增量复制模式；
c) 支持关系型数据库中结构化数据抽取到大数据平台关系型数据存储、非关系型数据存储、分布式文件存储、实时数据存储及消息队列；
d) 支持对关系型数据库数据的内容和类型转换操作，至少包括不同数据库差异转换、字段类型格式转换、时区转换等操作；
e) 支持对复制对象的过滤，至少包括模式过滤、表过滤、字段过滤、数据行过滤；
f) 支持数据复制操作的手动触发、定时调度及外部触发运行。定时调度运行应提供多种调度策略，至少包括固定时间间隔运行、指定时间点运行、指定时间范围运行、一次或指定次数运行等策略；外部触发支持标准 webservice 接口；
g) 应提供中心图形管理界面，应提供源端元数据查看、数据复制场景管理、数据表配置、表字段；
h) 选择配置、字段类型转换配置、触发机制配置、目标端输出配置、运行策略配置、运行监控等操作界面。

6.9.2 应用场景

6.9.2.1 场景图

数据库实时复制应用场景见图13。

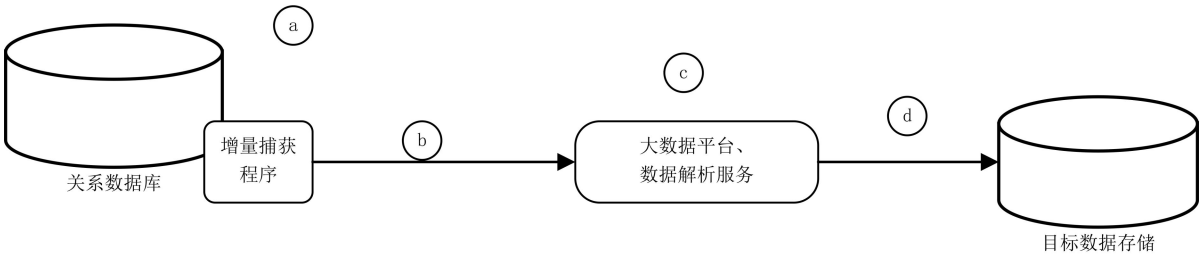

图 13 数据库实时复制应用场景

6.9.2.2 应用场景描述

应用过程如下：

a) 业务系统关系型数据库服务器上部署增量捕获程序；
b) 增量捕获程序捕获到增量数据通过 TCP 或消息队列发送到大数据平台增量接收服务；

c) 大数据平台解析增量数据，并将增量数据存储到大数据平台中；
d) 大数据平台分发增量数据到目标数据存储。

6.9.3 应用要求

应用要求包括：
a) 源数据库必须是 oracle、MySQL、PostgreSQL 等关系型数据库及 TiDB、OceanBase、达梦等国产数据库；
b) 源数据库表须包含主键或唯一索引；
c) 源端数据库须开启附加日志和强制归档模式；
d) 源端数据库的在线日志文件、归档日志文件须存放在文件系统上。

7 数据采集接入要求

7.1 综述

数据采集接入的方式应包括通信网关数据接入、分布式日志聚合数据接入、传统关系数据库数据并行接入、ETL数据接入等。

7.2 通信网关数据接入要求

7.2.1 一般要求

通信网关应实现各类现场设备的连接，在各种设备协议、网络协议做报文转换，以实现不同物理连接方式、通信协议的统一和标准化。

7.2.2 功能要求

通信网关应具备的基本功能如下:
a) 支持多种通讯链路，如:RS-232/422/485、CAN、以太网、WIFI、ZIGBEE、GPRS/WCDMA/CDMA2000等；
b) 支持采集现场的多种设备协议，并以多种设备协议向其他系统或设备提供数据分发服务。如:OPC、MODBUS、IEC61850、IEC60870、DNP3、BACNET、PLC 等；
c) 提供众多高级的功能特性，如:脚本系统，数据存储，设备报警，触发器，远程同步更新，网管系统等；
d) 基于互联网的应用开发、交流互动、在线调试、技术支持等；
e) 图形化的操作配置，带给用户良好的用户体验。

7.3 分布式日志聚合数据接入要求

7.3.1 一般要求

分布式日志聚合数据接入，简称 DTS-Flume，应包含如下组件:
a) Event:事件。一个数据单元，带有一个可选的消息头。如果是文本文件，通常是一行记录，这也是事务的基本单位；
b) Flow:流。Event 从源点到达目的点的迁移的抽象；
c) Client:客户端。操作位于源点处的 Event，将其发送到 DTS-Flume Agent；
d) Agent:代理。一个独立的 DTS-Flume 进程，包含组件 Source、Channel、Sink；

e) Source:源点。用来消费传递到该组件的 Event；
f) Channel:通道。中转 Event 的一个临时存储，保存有 Source 组件传递过来的 Event；
g) Sink:目的点。从 Channel 中读取并移除 Event，将 Event 传递到 Flow Pipeline 中的下一个 Agent；
h) Flow Pipeline:流管道。代理相互连接组成的一个事件传输途径。

7.3.2 功能要求

7.3.2.1 DTS-Flume 应能从数据源收集数据，并传送到目的地。为了保证可靠性，在送到目的地之前，应先缓存数据，待数据真正到达目的地后，再删除缓存的数据。
7.3.2.2 Flume 传输的数据的基本单位是 Event，当 Event 是文本文件时，应是一行记录，这是事务的基本单位。Event 从 Source，流向 Channel，再到 Sink，本身为一个 byte 数组，可包含 headers 信息。Event 表示一个数据流的最小完整单元，其来自外部数据源，完成处理后应输出到外部目的地。
7.3.2.3 DTS-Flume 运行的核心是 Agent。它是一个完整的数据收集工具，应含有三个核心组件，分别是 source、channel、sink。通过这些组件，Event 可以从一个地方传输到另一个地方，如图 14 所示。

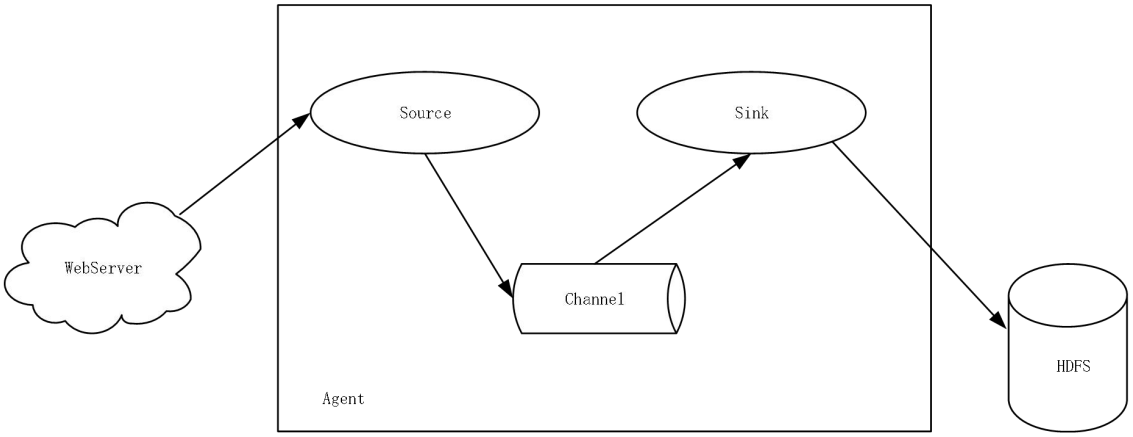

图 14 Agent 数据收集工具

7.3.2.4 source 可接收外部源发送过来的数据。不同的 source 可接受不同的数据格式。其中的目录池(spooling directory)数据源，可监控指定文件夹中的新文件变化，如文件夹中有文件产生，应立刻读取其内容。
7.3.2.5 channel 是一个存储区域，接收 source 的输出，直到有 sink 消费掉 channel 中的数据。channel 中的数据直到进入到下一个 channel 中或者进入终端才会被删除。当 sink 写入失败后，可自动重启，不会造成数据丢失，具有高可靠性。
7.3.2.6 sink 应对 channel 中的数据进行处理，然后送给外部源或者其他 source。如数据可以写入到 HDFS 或者 HBase。

7.4 传统关系数据库数据并行接入要求

7.4.1 一般要求

传统关系数据库数据并行接入简称DTS-Sqoop，是一个用来将OTS和关系型数据库中的数据相互转移的工具，应能够将一个关系型数据库(例如:MySQL ，Oracle，Postgres等等)中的数据导进到OTS中，也能够将OTS数据导进到关系型数据库中。

7.4.2 功能要求

DTS-Sqoop工作流程如下:

a) 读取要导入数据的表结构，生成运行类，默认是 Query Result，并生成 jar 包，然后提交给 Hadoop；
b) 设置 job 任务的各个参数；
c) 由 Hadoop Map/Reduce 来执行 Import 命令，包括如下步骤:
   1) 对要导入的数据进行切分；
   2) 完成切分后，记录各个读取数据范围，以便读取；
   3) 读取以上 2)记录的所有数据范围；
   4) 然后创建 Record Reader 从数据库中读取数据；
   5) 创建 Map；
   6) Record Reader 从关系型数据库中一行一行读取数据，并设置好 Map 的 Key 和 Value，交给 Map；
   7) 运行 Map。

7.5 ETL 数据接入要求

7.5.1 一般要求

ETL数据接入简称DTS-Kettle，是个ETL工具集，应能够完成数据的抽取、转换处理和加载等功能，应给用户提供图形化的用户环境来定义数据处理任务。

7.5.2 功能要求

7.5.2.1 工作流

DTS-Kettle包含transformation和job等两种脚本文件。transformation完成针对数据的基础转换，job则完成整个工作流的控制。可视化操作界面中列出的是一个transformation中基本的属性，可以通过各个节点来查看，包括:

a) DB 连接:显示当前 transformation 中的数据库连接，每一个 transformation 的数据库连接都需要单独配置；
b) Steps:一个 transformation 中应用到的环节列表；
c) Hops:一个 transformation 中应用到的节点连接列表；
d) Core Objects 中列出的是 transformation 中可以调用的环节列表，可以通过鼠标拖动的方式对环节进行添加。包括:
   1) Input:输入环节；
   2) Output:输出环节；
   3) Lookup:查询环节；
   4) Transform:转化环节；
   5) Joins:连接环节；
   6) Scripting:脚本环节。

7.5.2.2 转换常用环节的功能说明表

转换常用环节的功能说明见表1。

表 1　DTS-Kettle 转换环节

类别	环节名称	功能说明
Input	文本文件输入	从本地文本文件输入数据
	表输入	从数据库表中输入数据
	获取系统信息	读取系统信息输入数据
Output	文本文件输出	将处理结果输出到文本文件
	表输出	将处理结果输出到数据库表
	插入/更新	根据处理结果对数据库表进行插入更新。根据查询条件中的字段进行判断，如果数据库中不存在相关记录则插入，否则为更新
	更新	根据处理结果对数据库进行更新，若需要更新的数据在数据库表中无记录，则会报错停止
	删除	根据处理结果对数据库记录进行删除，若需要删除的数据在数据库表中无记录，则会报错停止
Lookup	数据库查询	根据设定的查询条件，对目标表进行查询，返回需要的结果字段
	流查询	将目标表读取到内存，通过查询条件对内存中数据集进行查询
	调用 DB 存储过程	调用数据库存储过程
Transform	字段选择	选择需要的字段，过滤掉不要的字段，也可做数据库字段对应
	过滤记录	根据条件对记录进行分类
	排序记录	将数据根据指定条件进行排序
	空操作	无操作
	增加常量	增加需要的常量字段
Scripting	Modified Java Script Value	扩展功能，编写 JavaScript 脚本，对数据进行相应处理
Mapping	映射（子转换）	数据映射
Job	Sat Variables	设置环境变量
	Get Variables	获取环境变量

7.5.2.3　Job 中基本的属性

可视化操作界面上列出的是一个Job中基本的属性，可以通过各个节点来查看，包括:

a)　DB 连接:显示当前 Job 中的数据库连接，每一个 Job 的数据库连接都需要单独配置;

b)　Job entries:一个 Job 中引用的环节列表见表 2。

表 2　Job 常用环节的功能说明

类别	环节名称	功能说明
Job entries	START	开始

	DUMMY	结束
	Transformation	引用 Transformation 流程
	Job	引用 Job 流程
	Shell	调用 Shell 脚本
	SQL	执行 SQL 语句
	FTP	通过 FTP 下载
	Table exists	检查日标表是否存在，返回布尔值
	File exists	检查文件是否存在，返回布尔值
	JavaScript	执行 JavaScript 脚本
	Create file	创建文件
	Delete file	删除文件
	Wait for file	等待文件，文件出现后则继续下一个环节
	File Compare	文件比较，返回布尔值
	Wail for	等待时间，设定一段时间，kettle 流程处于等待状态
	Zip file	压缩文件为 ZIP 包

7.6 消息集群数据接入要求

7.6.1 一般要求

消息集群数据接入简称KQS，是一种分布式的、基于发布/订阅的消息系统。KQS应完成如下目标:

a) 以时间复杂度为 O(1)的方式提供消息持久化能力，即使对 TB 级以上的数据也能保证常数时间复杂度的访问性能；
b) 高吞吐率。即使在配置一般的商用机器上也能做到单机支持每秒 100K 条以上消息的传输；
c) 支持集群服务器之间的消息分区及分布式消费，同时保证每个分区内的消息顺序传输；
d) 同时支持离线数据处理和实时数据处理；
e) 支持在线水平扩展。

7.6.2 功能要求

KQS应由下列组件组成:

a) Broker。KQS 集群包含一个或多个服务器，这种服务器被称为 broker；
b) Topic。每条发布到 KQS 集群的消息都有一个类别，这个类别被称为 Topic。不同 Topic 的消息应分开存储。一个 Topic 的消息虽然保存于一个或多个 broker 上但用户只需指定消息的 Topic 即可生产或消费数据而不必关心该数据存于何处；
c) Partition。Partition 是物理分区的概念，每个 Topic 包含一个或多个 Partition；
d) Producer。负责发布消息到 KQSBroker；
e) Consumer。消息消费者，向 KQSBroker 读取消息的客户端；
f) Consumer Group。每个 Consumer 属于一个特定的 Consumer Group，系统管理员可为每 Consumer 指定所属的组名，若不指定则属于默认的组。

8 数据传输

8.1 一般要求

大数据服务提供者应：

a) 区分安全域内、安全域间等不同的大数据服务相关的数据传输场景，建立相应的数据传输安全策略和规程；
b) 采用满足数据传输安全策略相应的安全控制措施，如安全通道、可信通道、数据加密等；
c) 建立数据传输接口安全管理工作规范，包括安全域内、安全域间等数据传输接口规范；
d) 具备在构建传输通道前对两端主体身份进行鉴别和认证的能力；
e) 具备对传输数据的完整性进行检测的能力以及相应的恢复控制措施；
f) 建立机制对数据传输安全策略的变更进行审核和监控，包括对通道安全配置、密码算法配置、密钥管理等保护措施的审核及监控。

8.2 功能要求

大数据服务提供者应：

a) 确保存储架构具备数据存储跨地域的容灾能力；
b) 建立满足应用层、数据平台层、操作系统层、数据存储层等不同层次的数据存储加密需求的数据存储加密架构。

附 录 A
（资料性）
关系数据库抽取接入说明

关系数据库抽取通过大数据平台提供的Web界面进行操作，完成数据接入操作。参考：DB15/T 1872-2020，大数据平台接入技术要求附录A。

附 录 B
（资料性）
服务网关服务接入说明

服务网关服务应用流程，通过大数据平台的服务网关服务提供的Web界面进行操作，参考：DB15/T 1872-2020，大数据平台接入技术要求附录B。

附 录 C
（资料性）
实时消息队列接入说明

实时消息队列应用流程及实时消息队列API接口参考：DB15/T 1872-2020，大数据平台接入技术要求附录C。

附 录 D
（资料性）
文件接收 FTP 服务接入说明

文件接收FTP服务接口及文件接收FTP服务示例参考：DB15/T 1872-2020，大数据平台接入技术要求附录D。

附 录 E
（资料性）
文件接收 webservice 服务接入说明

E.1 文件接收webservice服务API接口

文件接收webservice服务API接口见表E.1。

表 E.1 文件接收 webservice 服务 API 接口清单

序号	接口方法	接口说明
1	Login（String appId）	WebService 服务登陆。参数说明如下： appId：应用标识
2	Upload （ String fileName,File file,String chkSum,boolean overwrite,String authToken）	上传文件。参数说明如下： fileName：上传文件名 file：文件内容 chkSum：文件内容校验码 overwrite：是否覆盖原有文件内容 authToken：应用认证令牌信息
3	getFileSize（String fileName）	获取文件大小。参数说明如下： fileName：文件名

E.2 文件接收webservice服务

业务系统在生成好文件后，通过调用大数据平台webservice文件接收服务。调用服务示例如下:

```
public class SendFile2Ws {
  private static final String appId = ""; //request from bigdata team
  private static final String target = "网址链接";
  public static void main(String args[]) {
    String fileName = args[()], filePath = args[1], chkSum = args[2];
    boolean overwrite = false;
    try{
      Client client = new Client(new URL(target));
      Object[] results = client.invoke( "login", new Object[] { appId });
      String authToken=results[0].toString0;
      File file = new File(filePath);
      client.invoke("upload", new Object[] { fileName, file, chkSum, overwrite, authToken });
    }catch(Exception e){
      e.printStackTrace0;
     }
   }
 }
```

附 录 F
（资料性）
文件拉取 HTTP 服务接入说明

F.1 文件拉取HTTP服务API接口

文件拉取HTTP服务API接口清单见表F.1。

表 F.1 文件拉取 HTTP 服务 API 接口清单

序号	接口方法	接口说明
1	uploadFile（String actionUrl，String[] uploadFilePaths）	上传文件。参数说明如下： actionUrl：请求地址 inStream：文件路径列表
2	getFileSize（String fileName）	获取文件大小。参数说明如下： fileName：文件名

F.2 文件拉取 HTTP 服务示例

业务系统在生成好文件后，通过调用大数据平台HTTP文件拉取服务。调用服务示例如下:

```
public class HttpTransFile{
    @Suppess Warnings("finally")
    public static String uploadFile(String actionUrl, String[] uploadFilePaths) {
        String end = "\r\n";
        String twoHyphens="- -";
        String boundary="******";
        DataOutputStream ds = null;
        InputStream inputStream = null;
        InputStreamReader inputStreamReader = null;
        BufferedReader reader = null;
        StringBuffer resultBuffer = new StringBuffer();
        String tempLine = null;
        try {
            URL url = new URL(actionUrl);
            URLConnection urlConnection = url.openConnection();
            HttpURLConection httpURLConnection = (HttpURLConnection) urlConnection;
            try {
                URL url = new URL(actionUrl);
                URLConnection urlConnection = url.openConnection();
                HttpURLConection httpURLConnection = (HttpURLConnection) urlConnection;
                httpURLConnection.setDolnput(true);
                httpURLConnection.setDoOutput(true);
                httpURLConection.setUseCaches(false);
                httpURLConection.setRequestMethod("POST");
                httpURLConnection.setRequestProperty("Connection", "Keep-Alive");
                httpURLConnection.setRequestProperty("Charset", "UTF-8");
            httpURLConnection.setRequestProperty"Content-Type","multipart/form-data;boundary="
+ boundary);
                ds = new DataOutputStream(httpURLConnection.getOutputStream());
                for (int i= 0; i < uploadFilePaths.length; i++){
                    String uploadFile = uploadFilePaths[i];
                    String filename = uploaFile.substring(uplodFile.lastTndexOF("//")+ 1);
                    ds.writeBytes(twoHyphens + boundary + end);
                    ds.writeBytes("Content-Disposition: form-data; " + "name =\"file" + i
+"\",filename =\"" + filename+ "\"" + end);
                    ds.writeBytes(end);
                    FileInputStream fStream = new FileInputStream(uploadFile);
                    int bufferSize = 1024;
                    byte[] buffer = new byte[bufferSize];
                    int length=-1;
                    while ((length = fStream.read(buffer))! =-1){
```

```
                  ds.writer(buffer,0,length)
}
                    ds.writeBytes(end);
                    /* close streams */
                    fStream.close();
                }
                ds.writeBytes(twoHyphens+boundary+twoHyphens+end);
                /* close streams */
                ds.flush();
                if(httpURLConnection.getResponseCode()>=300){
                    throw new Exception("HTTP Request is not success, Response code is "+
httpURLConnection.getResponseCode());
                }
                if(httpURLConnection.getResponseCode()==HttpURLConnection.HTTP_OK){
                    inputStream=HttpURLConnection.getInputStream();
                    inputStreamReader=new InputStreamReader(inputStream);
                    reader=new BufferedReader(inputStreamReader);
                    tempLine=null;
                    resultBuffer-new StringBuffer();
                    while((tempLine=reader.readLine())!=null){
                       resultBuffer.append(tempLine);
                       resultBuffer.append("\n");
                        }
                }
            }catch(Exception e){
                e.prinstStackTrace();
            }finaly{
                if(ds!=null){
                    try{
                        ds.close();
                    }catch(IOException e) {
                        e.printStackTrace();
                    }
                }
                if(reader!=null){
                    try{
                        reader.close();
                    }catch(IOException e) {
                        e.printStackTrace();
                    }
                }
```

```
            if(inputStreamReader!=null){
                try{
                    inputStreamReader.close();
                    }catch(IOException e) {
                    e.printStackTrace();
                }
            }
            if(inputStream!=null){
                try{
                    inputStream.close();
                }catch(IOException e) {
                    e.printStackTrace();
                }
            }
            return resuItBuffer.toString();
        }
    }
}
public static void main(String[]args){
            String str = uploadFile("地址链接".new String[]{"需上传文件所在地址 1","需上传文件所在地址 2"});
    }
}
```

附 录 G
（资料性）
文件拉取 FTP 服务接入说明

文件拉取FTP服务API接口及文件拉取FTP服务示例参考：DB15/T 1872-2020,大数据平台接入技术要求附录E。

附 录 H
（资料性）
文件拉取 NFS 服务接入说明

H.1 文件拉取 NFS 服务 API 接口

文件拉取NFS服务API接口清单见表H.1。

表 H.1 文件拉取 NFS 服务 API 接口清单

序号	接口方法	接口说明
1	downloadViaNFS (final String ip,final String user,final String password,final String dir)	NFS文件下载。参数说明如下: ip:NFS服务器地址 user:NFS服务用户名 password: NFS 服务密码 dir:文件所在路径
2	getFileSize(String fileName)	获取文件大小。参数说明如下: fileName:文件名

H.2 文件拉取 NFS 服务

业务系统在生成好文件后，通过调用大数据平台NFS文件拉取服务。调用服务示例如下：

```
public void downloadViaNFS(final String ip,final String user,final String password,final String dir){
    logger.debug("NFS download begin!");
    try {
        String url = "nfs://"+ip+"/"+dir;
        XFile xf = new XFile(url);
        if (xf.exists()){
            logger.debug("URL is OK!");
        }else{
            logger.debug("URL is bad!");
            return;
        }
        XFileExtensionAccessor nfsx = (XFilcExtensionAccessor)xf.getExtensionAccessor();
        if(!nfsx.loginPCNFSD(ip. user, password){
            logger.debug("login failed"!);
            return;
        }
        private static FTPClient fpClient = new FTPClient();
        private static String encoding = System.getProperty(" file.encoding");
public static boolean downFileViaFtp(String ip, int port, String username,String password, String
remotePath, String fileName,String localPath) {
            boolean result = false;
            try {
                int reply;
                ftpClient.setC ontrolEncoding(encoding);
                ftpClient.connect(ip, port);
                ftpClient,login(username, password);
                ftpClient.setFileType(FTPClient.BINARY_ FILE_TYPE);
                reply = fpClient.getReplyCode();
                If (!FTPReply.isPositiveCompletion(reply)) {
                    fipClient.disconnect();
                    System.err. printn("FTP server refused conection.");
                }
                String[]fileList = xf.list();
                XFile temp = null;
                long startTime = System.currentTimeMillis();
                int filesz= 0;
                for(String file:fileList){
                temp = new XFile(url+ "/"+file);
                XFileInputStream in = new XFileInputStream(temp) ;
                XFileoutputStream out = new XFileoutputStream(tempDir +File separator+file);
```

```
                    int c;
                    byte[] buf = new byte[8196];
                     while ((c = in.read(buf)>0) {
                        filesz += c;
                        out.write(buf, 0, c);
                        if (temp.can Write()){
                            temp.delete();
                            logger.debug(file + "is deleted!");
                        }else{
                            logger.debug(file + " can not be delted!");
                        }
                    }
                    long endTime = System.currentTimeMillis();
                    long timeDiff = endTime - starTime;
                    int rate = (int) ((filesz /1000) / (timeDiff / 1000.0));
                    logger.debug(filesz + " bytes copied @ " + rate + "Kb/sec");
                }catch (lOException e) {
logger.debug(e);
                }


            }


        }
    }
}
public static void main(String[] args) {
    downloadViaNFS(ip, username, password,dir );
    ...
    IHdfsService      hdfsService      =      (IHdfsService)
    UtilFactory.getHadoopUtil(UtilType.HDFSService, "hdfs");
    ...
    hdfsService.upload(hdfspath, localpath, overwrite);
}
```