Metadata handling for Big Data projects

M Golosova¹, a, V Aulov¹ and A Kaida²
¹ National Research Center “Kurchatov Institute”, 1, pl. Kurchatova, Moscow, Russia
² National Research Tomsk Polytechnic University, 30, Lenina avenue, Tomsk, Russia

¹ golosova_mv@nrcki.ru

Abstract. Metadata is information about information. In business, industry or long living scientific experiments metadata grow and evolve with the project lifecycle. It leads to changes in the structure of the metadata, and with time it becomes complex, sophisticated and fluid, so that even simple lookup request appears to be complicated enough to require special tools. Another issue is that metadata can be produced and stored in different ways – paper or digital documents and tables, or databases, or something very specific – depending on the initial capabilities and requirements to its utilization. Due to this, to have a holistic view of the project one often has to perform so called multi source requests, aggregating information from a number of different sources. This kind of requests is not easy to implement, and can hardly be used for online services due to the significant execution time. This paper describes a possible solution by suggesting a method of metadata integration organization and providing an example of its application to information infrastructure of a HEP experiment.

1. Introduction

Metadata is information about information. In business, industry or long living scientific experiments metadata grow and evolve with the project lifecycle to fit project needs. To operate with the information and support its structure, project adopts an infrastructure (including databases, user interfaces, workflows, etc.). However the evolution of metadata inevitably leads to the necessity of changes in the information structure initially created for the project. And even when changes in the information structure are not so noticeable, with time projects still accumulate more and more metadata – historical and actual. Due to this, requirements to storage and access also change, which is most of the time an unpredictable process. While the volume scale can be estimated in advance more or less trustworthily, one can hardly imagine all the possible scenarios of information utilization: all forms of aggregations for regular reports or for online monitoring services, all lookup requests people may have, etc. Even simple lookup request can become complicated enough to require special tools beyond original infrastructure. In other words, the infrastructure becomes incapable of supporting the actual metadata handling.

Possible solution could be to reorganize the whole information infrastructure every 5–10 years, but it is a difficult (sometimes even impossible) task that requires a lot of time and manpower – and does not guarantee that new version will not become obsolete in a year or two.

This paper describes another approach: information from already existing sources is integrated to dedicated storage(s) “on the fly”. Dedicated storage is a storage having both data scheme and architecture optimized for a given type of requests. Their number, combination and data schemas
depend on current needs and can be easily adjusted according to new requirements. It allows to keep long-established workflow of users and systems and at the same time – provides flexibility to address newly appearing tasks, requiring access to the whole information field of a project, not a single information system.

A system described in this manuscript has been developed by a joint team of NR Tomsk Polytechnic University and NRC Kurchatov Institute under the R&D project of Data Knowledge Base for HENP experiments, initiated in 2016 [1]. This system provides flexible mechanism of metadata integration and, due to the complete isolation of project-related logic from integration process as a whole, can be used as a basis for knowledge base of any project.

2. Metadata integration system

2.1. Architecture

The metadata integration system is responsible for populating and updating of dedicated storages. It can be represented as a number of ETL (extract-transform-load) pipelines between original metadata sources and final storages. To construct and update user interfaces as quickly as possible (add information from previously unused source, add fields from a source already in use, change initial conditions for data extracting, etc.), this system has to meet following requirements:

- all-purposeness (be able to work with any set of original metadata sources and final storages);
- independency of ETL processes (allow to add or remove one process without affecting others);
- flexibility of ETL processes (allow to modify existing ETL process “on the fly”, to keep already integrated data and apply changes only to the future integration process).

Assuming that one can never predict all the possible types of data sources, final storages and transformations required, it was decided to use custom executable modules as basic construction elements for the ETL processes. These modules are responsible only for the selective logical operations on data units flowing through the ETL pipe: extract new and updated data from an original source; transform data unit(s) according to some algorithm – calculate derived values, generate surrogate keys, append unit with information from another original source, convert it to a specific format (required by dedicated storage), etc.; and load data unit(s) to the dedicated storage. There is a number of supervising and flow control operations, including but not limited to:

- processing parallelization and distribution;
- failure recovery;
- reprocessing of failed data units;
- forming a pipeline of executable modules;
- delivering data units between modules in the pipeline;
- triggering data taking from the initial sources by a schedule;
- storing offset of the last fully processed data unit.

These operations are performed by special supervising processes. And as these operations do not depend on the content of the data units, running the pipe nor the data schemas in the original and dedicated data storages, it can be said that same implementation of the supervisors can be used to construct ETL processes not only within knowledge base for HENP experiments, but for any other project requiring information integration.

2.2. Node topology

Every ETL process consists of three types of nodes: source (responsible for producing of data units running through the pipeline), processor (data transformation) and sink (loading data to the final storage). Simplest workflow can be represented as a linear pipeline: single source, single sink and a number of processors between them, applying one after another. But in more complicated situations there can be a need to split the pipe, or join information from different sources, or loop processing. To achieve this, the metadata integration system must support arbitrary topologies, specifying links between nodes, rules for split and join operations, conditions for loop processing, etc.
2.3. Implementation

The metadata integration system, developed for HENP Data Knowledge Base, is based on Apache Kafka [2]. Apache Kafka is an open-source stream-processing software platform developed by the Apache Software Foundation, written in Scala and Java. It provides:

- possibility to run parallel processing, distributing it over a number of computing nodes;
- mechanism to organize data flow through nodes according to a configured topology;
- temporary storages for data units query between topology nodes;
- possibility to restart processing from the same data unit it was stopped on (storing current offset).

Supervisor processes are derived from base Java classes provided by Kafka and Kafka Streams libraries. Their main duty is to run an external executable module with configured command line as a subprocess and transfer data to it from Kafka internal storages (topics), and vice versa.

Topology of processor nodes should be described in a configuration file. This file is used by topology constructor program, derived from topology builder class of Kafka Streams library.

External executable modules can be written in any programming language as long as they can be executed on cluster nodes. The only requirement for these modules is to support internal communication protocol, described in the next subsection, and use it to get input and write output data.

2.4. Internal communication protocol

The internal communication protocol was initially created to be as simple as possible for implementation, so that worker program could be written from scratch on any program language without usage of any specific library implementing this protocol. It allows a developer to choose the most appropriate language to implement given worker’s logic – without reference to other workers’ implementation. And, speaking in wider terms, it also allows developers and experts with significantly different background to work on the same project using habitual means to perform tasks in the most efficient way.

Communication between supervisor and worker (executable module) is carried on through standard input and output streams. All information and error messages from the worker are caught by supervisor on the worker’s standard error stream. Data units are passed as text (as metadata mostly do consist of text data), each unit terminated with special End-Of-Message symbol (EOM). To avoid data accumulation in the stream between supervisor and worker, special signal symbol (End-Of-Process, EOP) is passed from worker to supervisor when the worker is ready to operate with new input data.

For sink modules, and sometimes for processors too, it can be useful to perform batch processing. To combine messages with data units into a batch, another special symbol – End-Of-Batch (EOB) – is used. Supervisor passes one by one a number of messages (limited to a configured maximum batch size), and after the last EOM adds EOB. Worker, on its side, reads input messages till it meets EOB, and only then starts operating (e.g., passes set of data to a final storage through its bulk load interface).
Initially there was a pre-defined set of special (control) symbols, defined independently for every type of nodes due to the implementation specifics. However, to avoid possible difficulties, it was decided to specify a common default set of these symbols valid for every type of nodes – and, moreover, make them configurable, so that supervisor could pass protocol configuration to the worker on the startup and make sure that both participants are sharing the same set of symbols. In case of worker not accepting suggested configuration, both must use the default set.

| Control symbol | Hex | ASCII name         |
|----------------|-----|--------------------|
| EOM            | 0A  | LF (line feed)     |
| EOP            | 00  | NUL (null)         |
| EOB            | 17  | ETB (end of transmission block) |

3. Application

Described metadata integration system is successfully applied to two use cases of ATLAS [3] experiment metadata integration: scientific papers metadata and production system online monitoring.

ATLAS experiment at the LHC at CERN is one of the most data intensive experiments in High Energy Physics. Its history dates back more than a quarter of a century, and currently the managed data volume of the experiment is more than 380 petabytes. To stay in touch with all the inner processes, such as data modelling or processing, the community uses different monitoring interfaces. However, even though the amount of metadata describing the data processing (one of the most intensive metadata-producing part of the project) since 2006 lies in the range of terabytes, these interfaces commonly suffer from the same issue: if they make an aggregation and do it upon request (to provide the most actual information), users have to wait a few minutes to see the result. 5 minutes can be fast enough for offline data processing or regular report generation – but it is not good enough for interactive user interface [4]. User needs – information lookup by key, aggregation, full-text search or search by links between objects – are way too different to fulfil them all by a single universal storage. Even the most universal type of the storages, relational database (technically providing the possibility to execute query of any type), becomes non-relational\(^1\) one when every day there appears up to 2 million of new records – and the records are not only written to the database, they also can be updated every time when the status of processing task changes.

A common practice to address this issue is to create pre-processed summaries (views) for the most common time-consuming requests – but it narrows down the potentialities to create flexible interfaces allowing users to parameterize their request and get the exact information they need. Also, for very “heavy” requests (say, with execution time above an hour) it is not possible to actualize corresponding view too often and provide more actual information than the one dated with the moment when the request execution was started. Due to these limitations, this approach also can’t be used as a universal solution.

The Data Knowledge Base provides another solution by means of metadata integration system, described in previous section. Two main use cases addressed by Data Knowledge Base are discussed in the following to illustrate its usage on example.

The first use case is the reconstruction of connectivity between data samples and published scientific results based on them. Information about this connectivity for archive data is available only in human-oriented form: PDF documents containing lists of data samples or tables with data parameters (data taking year, center-of-mass energy, etc.). To extract and link together all the available

---

\(^1\) Non-relational database, NoSQL database – storage for Big Data with specific type of queries optimized, while others are strictly forbidden (as their execution would require scanning through all the records stored, which for Big Data may take hours or even days).
information, information about scientific papers and internal supporting notes for these papers was integrated into OpenLink Virtuoso [5] – ontology storage with reasoning mechanism (a mechanism to derive new knowledge from existing concepts and roles that are not expressed in the initial ontology).

Figure 2 represents the ETL process created for this task. It consists of single Source module, processing pipeline (divided in two independent pipelines after stage 1.5 and then joined again) and single Sink module. The processing pipeline includes one processor of a special type: standard processor, which can be reused for any ETL process, independently of the inner content of the passed data units.

![Figure 2. ETL process for ATLAS scientific paper metadata integration.](image)

**Stage 1.0** (Source): extract base metadata for all papers from ATLAS GLANCE system [6].

**Key Filter** (Standard processor): filter already known papers, leaving only new, added after the last run (data unit: single paper metadata).

**Stage 1.5** (Processor): get additional metadata for a single paper from CERN Document Server, including lists of supporting notes [7].

**Stage 5.5** (P): convert paper metadata into TTL and SPARQL statements, acceptable for Virtuoso.

**Stage 1.8** (P): download internal notes PDF documents to HDFS storage for further processing.

**Stage 3.0** (P): parse and analyse PDF documents to extract additional metadata (e.g. lists of datasets, in which the paper is based).

**Stage 5.4** (P): convert internal notes metadata into TTL and SPARQL statements.

**Stage 6.0** (Sink): upload information to Virtuoso RDF storage.

Another use case is the creation of arbitrary summary pages for data samples lookup and monitoring usage. For the lookup service the most user-friendly variant is the full-text search by all text information available for data sample: its name, project and campaign where it was created, description, tags, etc. Elasticsearch full-text search engine provides this functionality along with aggregation capabilities, so it was chosen as the dedicated storage for both tasks [8].
Information about data samples and sample processing tasks is integrated from Production System Oracle database, Rucio Distributed Data Manager, ATLAS Metadata Interface and ATLAS Analytics CloudLab Elasticsearch cluster and is accessible for researchers via web-interface [9–11]. This interface is used for information lookup in knowledge base as in the integrated information field of the experiment. It also provides users with pre-designed summary tables with actual data, aggregated by different parameters from number of information systems previously considered only as isolated sources to be used independently from each other. Reaction time of the interface pages amounts to a few seconds. This result, in comparison with minutes required to obtain same aggregated view from original metadata sources, indicates a significant progress toward user-friendly multi-functional interactive navigation system for the integrated information field of ATLAS experiment. Chosen approach allows bringing together wider scope of information systems and addressing almost every possible user need with the most appropriate technologies, meaning that all requests will be executed fast enough to make interactive navigation seamless for users.

4. Future plans
The internal communication protocol, described in section 2.4, was designed to be as simple as possible in implementation – but, as a consequence, it is not very extendable, which leads to significant limitations on interaction between parts of the system and the processing topology flexibility as a whole. Further steps forward of the project include the development of a new, more
elaborate protocol to be maintained together with the original one. New protocol is meant to be flexible enough to provide support to a variety of additional features being considered (or not even considered yet) as possible extensions of the whole system. It will allow to address wider range of metadata integration use cases with less data transfer and temporary storage overhead.

To improve usability, user interfaces for common tasks of ETL process creation, modification, overview, administration and monitoring will be created.

Experience gained in DKB project can be also used to create knowledge base for large computing facility in Kurchatov Institute which provides storages, computation power and infrastructure for number of scientific experiments, such as those in CERN (ATLAS, ALICE, LHCb), next generation genome sequencing and cryogenic electron microscopy.

5. Conclusion
The Data Knowledge Base for HENP experiments, being developed since 2016, has proved itself in case of ATLAS experiment as a flexible instrument for organization of integrated information field. Instance of DKB, installed in CERN, is being integrated in Production System as one of instruments for data processing tasks monitoring and analysis and is actively evolving in this direction.

The approach and technologies used in this work have no inner limitations on subject area and can be used as a foundation for knowledge base of any other scientific, business-related or industrial project.

Acknowledgements
The work was supported by the Russian Ministry of Science and Education under contract No. 14.Z50.31.0024 and by the Russian Science Foundation under contract No. 16-11-10280.

Creation of computing and storage infrastructure for Knowledge Base was supported by the NRC "Kurchatov Institute" (order No. 1608).

References
[1] Grigorieva M, Aulov V, Klimentov A and Gubin M 2016 Knowledge base of scientific experiment Open Systems. DBMS 4 42
[2] Apache Kafka [Online] https://kafka.apache.org/ [accessed on: 20.08.2018]
[3] ATLAS collaboration 1992 ATLAS: Letter of Intent for a General-Purpose pp Experiment at the Large Hadron Collider at CERN (Letter of Intent CERN-LHCC-92-04, CERN-LHCC-I-2) (CERN)
[4] Nielsen J 1993 Usability Heuristics in Usability Engineering (San Francisco: Morgan Kaufmann Publishers Inc) 134–8
[5] Erling O and Mikhailov I 2007 RDF Support in the Virtuoso DBMS CEUR Proceedings vol. 301. Proc. of the 1st Conference on Social Semantic Web, Leipzig, Germany, Sep 26-28
[6] ATLAS GLANCE [Online] https://atglance.web.cern.ch/atglance/ [accessed on: 20.08.2018]
[7] CERN Document Server [Online] https://cds.cern.ch [accessed on: 20.08.2018]
[8] Elasticsearch [Online] https://www.elastic.co/products/elasticsearch [accessed on: 20.08.2018]
[9] Barreiro F H, Borodin M, De K, Golubkov D, Klimentov A, Maeno T, Mashinistov R, Padolski S, Wenaus T on behalf of the ATLAS Collaboration 2017 The ATLAS Production System Evolution: New Data Processing and Analysis Paradigm for the LHC Run2 and High-Luminosity Journal of Physics: Conference Series 898 052016
[10] Rucio Distributed Data Management system [Online] https://rucio.cern.ch/ [accessed on: 20.08.2018]
[11] ATLAS Metadata Interface [Online] http://ami.in2p3.fr/index.php/en/ [accessed on: 20.08.2018]