Finding Your Way Through the Jungle of Big Data Architectures

Torsten Priebe, Sebastian Neumaier
Data Intelligence Research Group
St. Pölten University of Applied Sciences, Austria
{torsten.priebe, sebastian.neumaier}@fhstp.ac.at

Stefan Markus
Simplity AT GmbH
Vienna, Austria
stefan.markus@simplity.ai

Abstract—This paper presents a systematic review of common analytical data architectures based on DAMA-DMBOK and ArchiMate. The paper is work in progress and provides a first view on Gartner’s Logical Data Warehouse paradigm, Data Fabric and Dehghani’s Data Mesh proposal as well as their interdependencies. It furthermore sketches the way forward how this work can be extended by covering more architecture paradigms (incl. classic Data Warehouse, Data Vault, Data Lake, Lambda and Kappa architectures) and introducing a template with among others “context”, “problem” and “solution” descriptions, leading ultimately to a pattern system providing guidance for choosing the right architecture paradigm for the right situation.

Index Terms—Data Architecture, Data Lake, Logical Data Warehouse, Data Fabric, Data Mesh

I. INTRODUCTION

Data science and machine learning are currently on everyone’s lips. Since useful analyses are only possible on a solid data basis, architectures such as Data Fabric or Data Mesh are being proposed. But are those “modern” approaches really so different than “traditional” ones like Data Warehouse? The goal of this paper is a systematic review of the various (big) data architectures for analytics that have been proposed in the literature. In order to understand the similarities and differences, we propose a structure in ArchiMate notation [1] based on the key DAMA wheel elements [2].

Note that at this stage, the paper presents work in progress. We cover Gartner’s Logical Data Warehouse and Data Fabric concepts [3, 4] as well as Delghani’s Data Mesh proposal [5]. Moving forward, we want to cover further architecture paradigms including classic Data Warehouse (both Kimball and Inmon style) [6, 7], Lindstedt’s Data Vault [8]), Data Lake as well as Lambda and Kappa architectures [9, 10] and extend the model to clearly show the dependencies and shared elements as already indicated in figure 2 in the next section. This should ultimately lead to a pattern system similar to the GoF’s software design patterns [11]. This pattern system will then provide guidance for choosing the right architecture paradigm for the right situation.

II. FRAMEWORK

As mentioned above, we base our structural framework on the DAMA-DMBOK and ArchiMate. We represent the DAMA elements Data Integration & Interoperability, Data Storage & Operations, Data Quality, Data Security and Metadata as ArchiMate application functions of a Data Architecture, which in turn is represented as an application collaboration. We decided to split the DAMA element Data Warehousing & Business Intelligence as we consider a data warehouse as a concrete application component in an according architecture. Furthermore, We amend Business Intelligence with Data Science, which is devoted a chapter in the DAMA-DMBOK, but did not make it into the wheel [2]. Data Governance and Data Modeling & Design are considered ArchiMate capabilities as they are not functions of the architecture, but rather supporting capabilities of the organization. We add Data Sources (as a data object) and Data Consumers (as a business actor) to the model and indicate the data flow as shown in figure 1.

![Fig. 1. Structural framework based on DAMA-DMBOK and ArchiMate.](image)

Given the structure presented above, now consider the following data architecture paradigms: The (classic) Data Warehouse Architecture and the Data Lake Architecture both build the basis for Garnter’s Logical Data Warehouse Architecture. The Data Fabric Architecture and its recent variation Data Mesh Architecture encorporate ideas from the Lambda Architecture and Kappa Architecture paradigms, which however focus mainly on the Data Integration & Interoperability function. The interdependencies are shown in figure 2. The following sections will cover the Logical Data Warehouse, Data Fabric and Data Mesh architectures in more detail.
III. LOGICAL DATA WAREHOUSE

The Logical Data Warehouse concept was introduced by Gartner in 2012 [3] and provides recommendations how organizations can build a demand-driven, data management capability for analytical applications. According to the authors, architectural approaches such as Data Warehouse, Data Lake and Data Virtualization are not to be understood as concurrent solutions, but as complementary components of an overarching architecture. Gartner’s Logical Data Warehouse Solution Path [12] also explicitly mentions Sandboxes and Stream Processing as components of a Logical Data Warehouse Architecture. This is shown accordingly in figure 3.

IV. DATA FABRIC

The combination of different data storage and integration techniques, however, without restriction to concrete architecture archetypes such as data lake or warehouse, led to the term Data Fabric, which was originally coined in 2015 by George Kurian of NetApp and then adopted and advocated again by Gartner [4]. According to the authors, Data Fabric is a design concept for attaining reusable and augmented data integration services, data pipelines and semantics for flexible and integrated data delivery.

Data Fabric may be seen as a successor and generalization of the Logical Data Warehouse concept presented in the previous section. It builds upon the idea of “polyglot persistence” [13] resp. a Polyglot Data Store, which combines storage approaches like Relational Database, Graph Database and/or File/Blob Store (such as Hadoop)\(^1\). As mentioned before, it however does not promote concrete implementation paradigms like Data Warehouse or Data Lake as shown in figure 4.

V. DATA MESH

The main focus area of the Data Fabric concept is Metadata, which according to Gartner consists of an (augmented) Data Catalog and a Knowledge Graph containing semantically linked metadata. The use of artificial intelligence resp. machine learning to partially automate the metadata creation are emphasized by Gartner’s term “active metadata”. On the contrary, reporting and analysis tools are not considered core scope of the Data Fabric concept [4], but rather under the responsibility of the Data Consumers. Therefore the Data Fabric Architecture box in figure 4 does not span the Business Intelligence & Data Science function.

\(^1\)https://hadoop.apache.org
data products, a common platform is needed to allow homogeneous interactions with the data products. Figure 5 shows this paradigm by adding Business Domains (represented as ArchiMate business collaborations) and their domain Data Products with their interfaces to the picture. As you can see in the diagram, Data Pipelines are also owned by the Business Domains, i.e. each domain is responsible for its own data transformations. A domain can consume data products from another domain. Like for Data Fabric, there is a strong focus on Metadata with a Data Catalog providing a cross-domain inventory of available data products. Also like for Data Fabric, reporting and analysis tools are not within the core scope of (and therefore Business Intelligence & Data Science is outside the Data Mesh Architecture box). However, unlike the other data architecture paradigms presented, the Data Mesh concept does take the Data Sources into closer consideration. Operational data is served via Operational Data Products (resp. their interfaces) just like Analytical Data Products.

VI. CONCLUSION

In this paper we have provided a first systematic attempt of representing key (big) data architectures based on a common framework in a common semi-formal notation. We have used the DAMA-DMBOK and ArchiMate for this. In particular, this paper has covered Logical Data Warehouse, Data Fabric and Data Mesh architectures in more detail.

In future work, we want to cover further architecture paradigms such as Lambda and Kappa architectures, extend the model to clearly show the dependencies and shared elements – leading ultimately to a pattern system similar to the GoF’s software design patterns [11]. This will involve introducing also other common elements of pattern templates like “context”, “problem” and “solution”, which will then provide guidance for choosing the right architecture paradigms.

REFERENCES