# Developing a Software Reference Architecture for Journalistic Knowledge Platforms

Marc Gallofré Ocaña, Andreas L. Opdahl

*University of Bergen, Bergen, Norway*

Abstract

For news organizations to survive and thrive in today's media landscape, they must utilize big data and artificial intelligence technologies effectively. News organizations that want to exploit techniques like machine learning and knowledge graphs for big data may manage to use them independently, but struggle to get everything to work together. A software reference architecture would help by providing a generic blueprint and capturing the tried-and-tested best practices for designing and implementing concrete solutions but, to the best of our knowledge, no suitable architecture has been proposed. This paper therefore outlines a software reference architecture for digitalization of newsrooms, along with a proof-of-concept of the architecture.

**Keywords**

Software Reference Architecture, Integrated Neural-Symbolic AI, Knowledge Graph, Newsroom

## 1. Introduction

News organizations must constantly adapt their business models to digital media innovations to improve information quality, competitiveness and growth. This changes how journalists and readers interact with news content and background information [1]. News agencies use big data and artificial intelligence (AI) for different journalistic purposes [2] that include: identifying and contextualizing newsworthy events for investigated journalism; facilitating data visualization in digital journalism; automating news writing in robot journalism (a.k.a. algorithmic journalism or automated journalism); and providing real-time fact-checking tools for political journalism. *Journalistic Knowledge Platforms* (JKPs) integrate these and related techniques into knowledge-centric big-data platforms. They are an emerging type of complex information system that integrates state-of-the-art Neural-Symbolic AI techniques [3] such as machine learning (ML), semantic knowledge representations and natural language processing (NLP) to support daily processes in newsroom workflows [4, 5]. In our research, we are focusing on JKPs that employ semantic knowledge graphs [6] for representing knowledge.

It is challenging for news organizations to evolve the many independent and task-specific systems they run today into cohesive and comprehensive JKPs [5]. On the management level, central challenges are that JKPs (a) are complex systems that must balance many concerns [7] and are thus challenging to adopt without architectural guidance; (b) must interoperate with a wide variety of in-house legacy systems and external services, including other JKPs [8]; and (c) are long-term investments that must be able to evolve to incorporate future best-of-breed components that replace or come in addition to existing ones [7]. On the technical level, JKPs also need to support (a) the ingestion of real-time news items from multiple sources of unstructured and semi-structure data which must be semantically annotated and represented (viz., lifted) and enriched in the knowledge base [9]; (b) the production of potentially newsworthy events which are continuously pushed to journalists [10]; (c) the different services for pulling information from the knowledge base [11]; and (d) mechanisms for continuously evolving and adapting machine learning models and ontologies for curating and enriching the knowledge base [12].

A software reference architecture (SRA) "is a generic architecture for a class of systems that is used as a foundation for the design of concrete architectures from this class" [13]. It defines the basic software elements and data flows that implement the functionalities of, and captures the best practices for designing and implementing complex information systems like JKPs. We can distinguish two types of SRA: practice-driven and research-driven [14]. Practice-driven SRAs are based on practical experience developing concrete architectures in a domain. They describe the "best-practices" and address legacy problems. Research-driven SRAs are designed for a class of systems where there are no experiences on developing them and are expected to become relevant in the future. They are based on the related research experiences. To a news organization, an SRA would bring a blueprint along with advice for how to evolve its current

*ECSA2021 Companion Volume, Robert Heinrich, Raffaela Mirandola and Danny Weyns, Växjö, Sweden, 13-17 September 2021*

marc.gallofre@uib.no (M. Gallofré Ocaña);
andreas.opdahl@uib.no (A. L. Opdahl)
0000-0001-7637-3303 (M. Gallofré Ocaña); 0000-0002-3141-1385 (A. L. Opdahl)

news production systems to become an integrated JKP.

Researchers have proposed several software architectures to deal with big data [15]. But most of them did not consider mechanisms for semantically representing and enriching heterogeneous data, continuously pushing live streams of data to the users, curating knowledge bases, and maintaining machine learning models. Hence, none of them are suitable for JKPs, or even for being adapted to the JKP domain. Therefore, in this paper, we address the question: "What would be a good software reference architecture for journalistic knowledge platforms?". To address this research question, we followed a design science approach [16] to develop a research-driven SRA for JKPs along with a proof-of-concept prototype of a JKP platform. To guide development we identified opportunities and challenges for JKPs from the literature [5], relied on our experience from developing previous JKP prototypes [17, 18], and collaborated with a software developer for the international newsroom market [17].

The rest of the paper is organized as follows: Section 2 summarises the research method; section 3 analyses the related literature, section 4 outlines the concerns for an SRA, section 5 introduces the SRA for JKPs, section 6 describes the implementation of the proposed SRA, and section 7 states our conclusions and further work.

## 2. Method

We have followed a design science approach [19, 20] to answer our research question. Design science "supports a pragmatic research paradigm that calls for the creation of innovative artifacts to solve real-world problems" [16]. In Information Systems, researchers typically conduct design science research following an iterative process that includes three different research cycles [21]. These cycles focus on understanding the application context or environment, studying and improving the theoretical framework, and evaluating the artifacts [20]. This approach allowed us to iteratively designing and validating the SRA for JKPs through the development and improvement of artifacts, based on the literature, and considering the contextual environment of developers and users of JKPs.

First, we identified the most relevant semantic knowledge graph-based JKP projects from our on-going systematic literature review on knowledge graphs for news work. From the identified projects, we derived the challenges and opportunities of JKPs [5] that together with our experiences with earlier JKP prototypes [17, 18], helped us to extract the concerns for the SRA. Then, we investigated an SRA for JKPs, designed a software architecture and prototyped a proof-of-concept a platform with an industrial collaborator for the international newsrooms market.

## 3. Related Work

### 3.1. Journalistic Knowledge Platforms

News agencies and researchers have developed several JKPs: The *Diari SEGRE* participated in the development of the *Neptuno* [22] which used semantic technologies for improving the creation, maintenance and exploration of news archives. The *AnnoTerra* project [23] proposed a prototype for integrating NASA earth science data sources to enhance news feeds from NASA's Earth Observatory using knowledge bases and semantic technologies. The BBC used knowledge graphs and linked open data to link information across news articles, and enrich their Content Management System (CMS) [24, 12]. The *Agencia EFE* in Spain and the *Agenzia ANSA* in Italy in collaboration with researchers developed the *NEWS* project [8] to automatize the metadata annotation of news and provide news intelligent information retrieval services using semantic technologies. Both the *Deutsche Welle* and BBC Monitoring collaborated with the *SUMMA* project [7] to develop a multilingual and multimedia platform employing NLP techniques for monitoring internal and external media production, and providing services for data journalists. The *Hermes* project [25] proposed a framework for searching and classifying news using semantic technologies to support decision makers. The *EventRegistry* project [9] developed a tool for collecting news articles, identifying and extracting information about the events, and summarizing and visualizing events. The *NewsReader* [10] proposed a platform to read multilingual streams of news and extract information about what, who, where and when for representing events in time using RDF and knowledge graphs, allowing users to find networks of actors and their implication over time. The *ASRAEL* [11] presented a system to aggregate news articles and leverage the Wikidata knowledge base for describing and clustering the events in news.

Typical problems faced in these projects are huge volumes of heterogeneous data, some of them arriving in real time [9, 10, 7]; complex processing pipelines that combine NLP, machine learning and knowledge representation [9, 10, 11]; and integration of legacy and external systems [12, 8] – problems that typically call for architectural guidance.

### 3.2. Software Reference Architectures

According to Nadal et al. [15], only two concrete software architectures [26, 27] and one SRA [15] for big data have considered the usage of semantic technologies. And none of them are suitable starting points for an SRA for JKPs. *LMS* [26] was designed with a clear focus on providing a *middleware* for sensor data and Internet of Things (IoT). *SOLID* [27] adapted the principles of the

*Lambda* processing architecture [28] to RDF[1] for gathering, storing and serving big data in real time. *Bolster* [15] extended the Lambda architecture by adding a new semantic layer to represent machine-readable metadata, contrary to JKPs that represent the data semantically. However, none of them cover mechanisms for enriching semantic data, continuously pushing live streams of data to users, and keeping machine learning models up-to-date. Pääkkönen and Pakkala [29] proposed an SRA for machine learning in edge computing environments that considered updating and maintaining of machine learning models, but did not consider semantic technologies.

The Lambda architecture has been criticised for its design: data and code are duplicated in two layers, namely speed and batch layer, and data requests need to be coordinated between both. This increases development, implementation, maintenance efforts, and hardware demands [30, 31]. The Kappa processing architecture was therefore proposed [30]: it removes the batch layer, only deals with real-time computation, and provides a single data view which is only changed when the code is updated and the old view is recomputed. However, the batch jobs are not clearly defined and if needed, they have to reprocess the current data view [31]. To overcome this challenge, Cerezo et al. [31] proposed the Phi processing architecture, which is inspired by Lambda, but delays the data stream replication after the real-time computation is done and provides a single data view.

To the best of our knowledge, no suitable SRA for JKPs has been proposed. Our proposed SRA for JKPs is inspired by the Phi processing architecture to overcome the challenges of the previous big-data architectures. It is focused on representing and enriching data semantically and keeping up-to-date machine learning models, along with the identified domain specific concerns of JKPs.

## 4. Concerns for an SRA

We identified the following main concerns from previous studies [5, 32] and the analysis of similar systems [8, 7, 9, 12, 11, 10]:

**C1** JKPs must interoperate with heterogeneous in-house legacy systems, external services, and other JKPs [8];

**C2** must be able to incorporate future components that replace existing ones [7];

**C3** must ingest real-time news items from multiple heterogeneous sources that must be semantically lifted [9];

**C4** must produce potentially newsworthy events which are continuously pushed to journalists [10];

**C5** must provide different services for pulling information from the knowledge base [11];

**C6** must continuously evolve machine learning models for enriching the knowledge base;

**C7** are knowledge-centric systems that provide knowledge representations of news, events and background information from the real-world which is constantly changing [12];

**C8** contain different databases for specific purposes like multimedia files, historical archives and real-time news feeds [7];

**C9** must facilitate schema evolution [12];

**C10** need to consider privacy, provenance, terms-of-use and data quality [8];

**C11** support news production where time is a critical factor and delays can lower the value of information [8];

**C12** and must consider big data properties such as data heterogeneity, volume and velocity [10].

In addition, a good SRA must satisfy the usual requirements of being feasible, representative, essential, easy to grasp, long lasting, and technology independent [13, 33]. We return to discuss these concerns in Section 5.

## 5. Software reference architecture for JKPs

The proposed SRA for Journalistic Knowledge Platforms (Fig. 1) is organized as five core services: the *Ingestor*, *Knowledge Base*, *Curator*, *Feeder* and *Retriever*. Each service comprises several micro-services, each of which is designed as an independent component with a clear API facilitating its replacement and integration, making it easy to scale and distribute [34] (**C1**, **C2**). Solutions like *Docker*[2] can be used to improve the availability and replaceability of micro-services.

The SRA data flow and processing steps are inspired by the Phi architecture [31], which is designed for big data and delays the downstream processing as much as possible after the real-time processing (**C11**, **C12**). The real-time processing step applies the data transformations once and for all near the sources, providing and combining both knowledge representations and unstructured data. Hence, the SRA facilitates the integration of neuro-symbolic AI by provisioning both types of data from the beginning.

To efficiently process data streams, JKPs must exploit concurrent and parallel processing. For example, JKPs

[1] www.w3.org/TR/rdf-schema

[2] www.docker.com

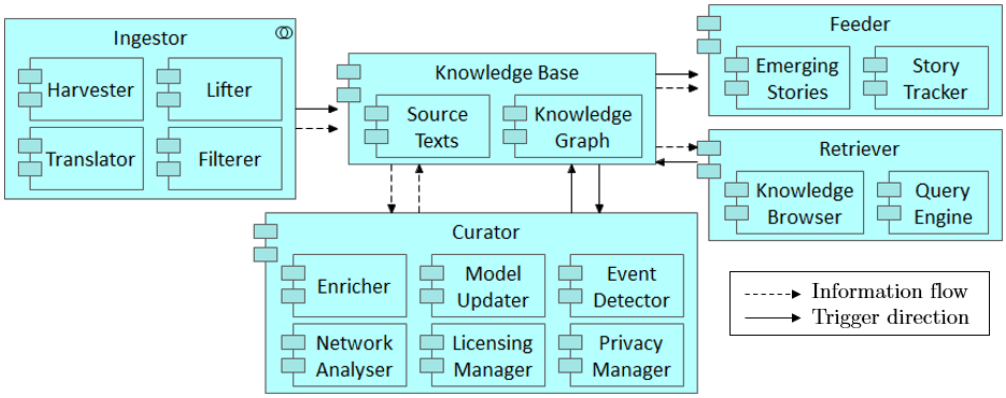


**Figure 1:** The SRA for JKPs (represented with ArchiMate 3.1 notation).

can integrate solutions like *Apache Spark*[3] for batch and *Apache Storm*[4] for real-time processing. To guarantee the message distribution along the different services, JKPs can employ message systems like *Apache Kafka*[5], and serialize the messages using *JSON-LD*[6], an extension of JSON for representing linked data.

## 5.1. Ingestor

The Ingestor is an *Extraction-Transformation-Loading* (ETL) service where a *Harvester* continuously downloads and ingests scheduled and real-time news, social media messages and multimedia items from sources like RSS, APIs and web-sites (**C1**, **C2** **C3**). Additional services such as the *Translator* and *Filterer* pre-process and clean the downloaded items, for example, translating text into a canonical language, normalizing data types, standardizing formats, and filtering advertisements. *Lifters* then annotate and transform the news-related items into knowledge-graph representations in real time (**C7**) before they are uploaded to the Knowledge Base. These knowledge graphs are represented in RDF following predefined ontologies like the Event Description Ontology [35] (**C3**), made general to facilitate schema evolution (**C9**) and data exchange (**C1**).

The Lifter is composed of AI modules. which we designed to be replaced or extended to follow the state-of-the-art [36] (**C2**). To combine the results from the different AI modules, these can use NIF [37] or NAF [38] vocabularies to standardize their annotations. Every annotation must provide quality information (e.g., accuracy and support values) and provenance to trace back the annotations to the process that generated them (**C1**). As a result, the Ingestor processes the real-time transformations once and near the source before they are stored in the Knowledge Base, and further processed by the Feeder and Curator.

## 5.2. Knowledge Base

The Knowledge Base provides persistent storage for source items and knowledge representations. It is composed of multiple instance of dedicated databases for multimedia and text files such as the *Source Texts* service. It contains the *Knowledge Graph* for representing news-related knowledge (**C8**), which we designed to provide a single data view that constantly changes to capture real-world evolution and avoid unnecessary delays (**C9**, **C11**). The Knowledge Base also provides a middleware to interact with the different repositories and integrate legacy archives (**C1**).

These storage services must be prepared to handle large volumes of data and intensive write operations in real time (**C12**). For example, distributed databases like *Apache HBase*[7] and *Cassandra*[8] can store large data volumes. Although many of the open source graph databases and triple stores with support for RDF and SPARQL are not distributed, some of them can hold more than one billion ($10^9$) triples [39] (e.g., *Blazegraph*[9] and *Jena TDB*[10]). Strategies to manage graph database size include: (a) partitioning the graph database according to resource types/predicates, themes or geolocations, or a combination of these; and (b) reducing the data stored in the Knowledge Graph by only keeping the knowledge representations and storing the textual and multimedia information in dedicated databases.

[3] spark.apache.org
[4] storm.apache.org
[5] kafka.apache.org
[6] www.w3.org/TR/json-ld11
[7] hbase.apache.org
[8] cassandra.apache.org
[9] www.blazegraph.com
[10] jena.apache.org

### 5.3. Curator

The Curator provides a diverse set of functionalities, comprising services with different behaviors. It improves the data in the Knowledge Graph, produces new representations for different journalistic purposes (**C7**), and updates AI/ML models (**C6**). For example, the *Enricher* enhances the Knowledge Graph representations using external information from the LOD (e.g., Wikidata); the *Model Updater* keeps up-to-date and re-trains AI/ML models; the *Event Detector* and the *Network Analyzer* use information from the Knowledge Base to identify events and networks of actors; and the *Licensing* and *Privacy Managers* propagate prohibitions, permissions, and obligations to flag and rectify violations (**C10**).

Curator services can be analyzed from different perspectives: what triggers the service, what information is used, and the main information flow. Curator services such as the Enricher and the Event Detector continuously ingest lifted items from the Knowledge Base, and can use information from external and internal sources respectively to generate new knowledge. Other services such as the Licensing and Privacy Managers outputs alerts that need to be addressed by the user. Others such as the Model Updater and the Network Analyzer are triggered periodically. For example, the Model Updater can be automatically triggered hourly or even more frequently to adapt real-time models to the current events using the most recent data, and it can also be triggered daily or even weekly to re-train models to follow current developments. The Model Updater can access any data in the Knowledge Base, having access to both semantic representations and raw sources. Since every semantic representation must reference the source of its annotations, it facilitates the generation of training data.

### 5.4. Feeder

The Feeder continuously monitors the streams of linked data fragments coming from the Ingestor to push real-time news information to the user (**C4**, **C11**). For example, it provides news feeds to users, suggests emerging stories, identifies trends, and allow journalist to follow current stories or developments.

### 5.5. Retriever

The Retriever lets users pull information from the JKP on demand. It provides a front-end to access, visualize and explore the Knowledge Base and its services (**C5**). For example, the Retriever can provide an end-point with pre-packaged SPARQL queries for particular purposes, like finding news stories related to a particular person or retrieving relevant background information for a given news event.

## 6. Prototype

We are developing a prototype that instantiates the SRA for JKPs (Fig. 2). The current version implements central parts of the Ingestor (Harvester and Lifter), Knowledge Base (Source Text and Knowledge Graph), Curator (Enricher and Event Detector) and Retriever (Pull API) services. We used Docker Swarm for orchestrating and replicating the services. The platform runs on 17 cloud instances (with total of 38 vCPU, 152GB RAM and 20TB disk). The messages are serialized using JSON-LD and passed between services using Kafka. Previous prototypes running on a simpler infrastructure [17] have already implemented additional parts of the Feeder, Curator and Retriever, which we plan to adapt into the current prototype.

We designed the Ingestor with services for Harvesting and Lifting. Our Harvester crawls news-related websites and harvests RSS feeds, Twitter accounts, NewsAPI[11] and GDELT[12]. The Twitter API provides real-time tweets streams from specific accounts, geographical areas or topics. NewsAPI aggregates and provides streams of news articles from over 50000 news sources and blogs. GDELT provides semi-structured information about conflict events from news all around the world, that have been automatically translated to English from 65 different languages. Our Lifter [36] transforms the news and event streams in real time into semantic knowledge representations following the Event Description Ontology [35]. It employs out-of-the-box NLP systems such as DBpedia Spotlight[13] and SpaCy[14] for semantically annotating textual items and linking them to Wikidata and DBpedia. To integrate their outputs, we use NIF because it is the only vocabulary based on RDF and OWL for describing NLP annotations. The Knowledge Base includes a Source Texts service implemented with Apache Cassandra and a Knowledge Graph implemented with Blazegraph. Cassandra has been used to store the textual information together with the IRIs of the news items represented in the Knowledge Graph. This decision allowed us to reduce the data stored in the Knowledge Graph; provide provenance by tracking news representations back to its source; and facilitate new training material for ML models based on the current state of our system. The Curator implements an Enricher and Event Detector. Our Enricher extends the lifted items with location-related background information extracted from DBpedia. Our Event Detector provides journalists with real-time events detected from GDELT streams. The Retriever exposes an API for pulling lifted news items from the Knowledge

[11] newsapi.org
[12] www.gdeltproject.org
[13] www.dbpedia-spotlight.org
[14] spacy.io

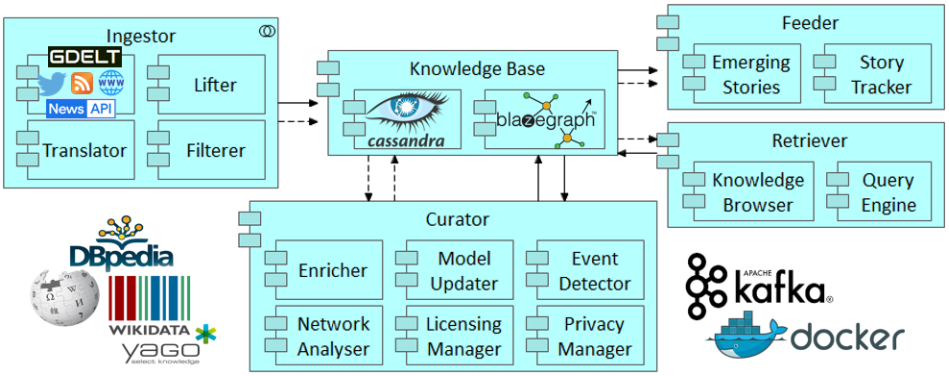


**Figure 2:** Visual representation of the instantiated architecture for the JKP prototype.

Base for a research challenge task[15], allowing external users to interact with our system.

## 7. Conclusion

We have proposed an SRA for JKPs to support the adoption of JKPs in newsrooms. Our architectural decisions were based on reported experiences with existing platforms, supported by our own experience developing a JKP in collaboration with industry. The SRA supports implementing, maintaining, and evolving JKPs. It represents those components and functionalities that are essential for JKPs and provides a vocabulary to compare and understand different JKP realisations. We designed the SRA to be technology independent, open-ended and long-lasting, with components and services that can evolve or be replaced and integrated with legacy systems. The SRA considers the integration of AI components, the evolution of ML models, and provides both knowledge representations and unstructured data together to facilitate the integration of neuro-symbolic AI. We have also presented a proof-of-concept implementation of a JKP that instiantiates the proposed SRA.

Because JKPs handle big data, they need a processing architecture that minimizes delays and maximizes information flow. Thus, the SRA is inspired by the Phi architecture and provides a single data view through the Knowledge Base. This reduces unnecessary data duplication, does not require coordination between different data views, and makes information available as soon as it arrives in the Knowledge Base. To reduce delays further, the SRA also includes a component for pushing streams of newsworthy events to journalists.

In Section 5 we have discussed how the proposed SRA implements the identified main concerns. Tables 1 and 2 summarize the mapping between the discussed concerns and the architecture patterns and core services.

**Table 1**
Mapping between architecture patterns and concerns

| Architecture pattern | Concern |
|---|---|
| Micro-services | C1,C2 |
| Phi architecture | C11,C12 |

**Table 2**
Mapping between core services and concerns

| Service | Concern |
|---|---|
| Ingestor | C1,C2,C3,C7,C9 |
| Knowledge Base | C1,C8,C9,C11,C12 |
| Curator | C6,C7,C10 |
| Feeder | C4,C11 |
| Retriever | C5 |

## 8. Further work

We plan to evaluate the correctness and the utility of the proposed SRA, being this part one of the main challenges of designing an SRA [40, 41]. Hence, we plan to map existing JKPs from the literature into our proposed SRA to verify that it covers the essential components of JKPs and evaluate its correctness. We also want to conduct a qualitative evaluation of the SRA with component developers to validate its feasibility, understandability and utility. This qualitative evaluation would be composed by questionnaires inspired by the ISO/IEC 25000 standard and a series of interviews. Besides, we plan to extend our prototype to facilitate empirical evaluation of the SRA instantiation. Possible empirical paths include:

[15] multimediaeval.github.io/editions/2021/tasks/emergingnews

measuring its ability to detect events and newsworthy information, experiments allowing outsiders to interface with and extend the prototype, and testing interoperability and scalability. From experiments such as these, we can evaluate how well the SRA facilitates the interaction with and expansion of JKP components, and the performance of the resulting solutions.

We also consider designing a software architecture maturity model to provide an efficient path for the evolution of current news organizations systems into JKPs. This maturity model would bring news organizations with a tool to evaluate how far are their systems from the proposed SRA and its implementation as JKP. Additionally, we plan to extend the current SRA with process views for selected components, aiming to improve understanding and ease the adoption path.

The experiences learned from designing an SRA for JKP could be extended to other domains with similar needs. This process would allow us to evolve and abstract the proposed SRA towards a more generic SRA for integrating neural-symbolic AI systems in similar contexts.

## Acknowledgments

This research was supported by the Norwegian Research Council IKTPLUSS project 275872.

We would like to thank past and present research fellows, colleagues and assistants, who together are the true formulas for any of the scientific manuscripts that we have ever written.

## References

[1] J. Vázquez Herrero, S. Direito-Rebollal, A. S. Rodríguez, X. García, Journalistic Metamorphosis: Media Transformation in the Digital Age, Springer International publishing, 2020. doi:10.1007/978-3-030-36315-4.

[2] S. C. Lewis, O. Westlund, Big data and journalism, Digital Journalism 3 (2015). doi:10.1080/21670811.2014.976418.

[3] P. Hitzler, F. Bianchi, M. Ebrahimi, M. K. Sarker, Neural-symbolic integration and the Semantic Web, Semantic Web 11 (2019) 1–9. doi:10.3233/SW-190368.

[4] P. Costa, G. Marques, A. Serra, D. Moraes, A. Busson, A. Guedes, G. Lima, S. Colcher, Towards neural-symbolic AI for media understanding, 2020. doi:10.5753/webmedia_estendido.2020.13083.

[5] M. Gallofré Ocaña, A. L. Opdahl, Challenges and opportunities for Journalistic Knowledge Platforms, in: Proceedings of the CIKM 2020 Workshops, 2020. URL: http://ceur-ws.org/Vol-2699/paper43.pdf.

[6] Aidan Hogan, et al., Knowledge graphs, ACM Comput. Surv. 54 (2021). doi:10.1145/3447772.

[7] U. Germann, R. Liepins, G. Barzdins, D. Gosko, S. Miranda, D. Nogueira, The SUMMA platform: A scalable infrastructure for multi-lingual multimedia monitoring, in: Proceedings of ACL 2018, System Demonstrations, 2018. doi:10.18653/v1/P18-4017.

[8] N. Fernández, D. Fuentes, L. Sánchez, J. A. Fisteus, The NEWS ontology: Design and applications, Expert Systems with Applications 37 (2010). doi:10.1016/j.eswa.2010.06.055.

[9] G. Leban, B. Fortuna, J. Brank, M. Grobelnik, Event registry: Learning about world events from news, in: Proceedings of the 23rd International Conference on World Wide Web, 2014. doi:10.1145/2567948.2577024.

[10] P. Vossen, R. Agerri, I. Aldabe, A. Cybulska, M. van Erp, A. Fokkens, E. Laparra, A.-L. Minard, A. P. Aprosio, G. Rigau, M. Rospocher, R. Segers, Newsreader: Using knowledge resources in a cross-lingual reading machine to generate more knowledge from massive streams of news, Special Issue Knowledge-Based Systems, Elsevier 110 (2016). doi:10.1016/j.knosys.2016.07.013.

[11] C. Rudnik, T. Ehrhart, O. Ferret, D. Teyssou, R. Troncy, X. Tannier, Searching news articles using an event knowledge graph leveraged by wikidata, in: Companion Proceedings of The 2019 World Wide Web Conference, 2019. doi:10.1145/3308560.3316761.

[12] Y. Raimond, T. Scott, S. Oliver, P. Sinclair, M. Smethurst, Use of semantic web technologies on the BBC web sites, in: Linking Enterprise Data, 2010. doi:10.1007/978-1-4419-7665-9_13.

[13] S. Angelov, P. Grefen, D. Greefhorst, A framework for analysis and design of software reference architectures, Information and Software Technology 54 (2012). doi:10.1016/j.infsof.2011.11.009.

[14] S. Angelov, J. J. M. Trienekens, P. Grefen, Towards a method for the evaluation of reference architectures: Experiences from a case, in: Software Architecture. ECSA 2008., 2008. doi:10.1007/978-3-540-88030-1_17.

[15] S. Nadal, V. Herrero, O. Romero, A. Abelló, X. Franch, S. Vansummeren, D. Valerio, A software reference architecture for semantic-aware big data systems, Information and Software Technology 90 (2017). doi:10.1016/j.infsof.2017.06.001.

[16] A. Hevner, S. Chatterjee, Design Science Research in Information Systems, 2010. doi:10.1007/978-1-4419-5653-8_2.

[17] A. Berven, O. Christensen, S. Moldeklev, A. Opdahl, K. Villanger, A knowledge graph platform for newsrooms, Computers in Industry (2020).

doi:10.1016/j.compind.2020.103321.

[18] M. Gallofré Ocaña, L. Nyre, A. L. Opdahl, B. Tessem, C. Trattner, C. Veres, Towards a big data platform for news angles, in: 4th Norwegian Big Data Symposium (NOBIDS) 2018, 2018. URL: http://ceur-ws.org/Vol-2316/paper1.pdf.

[19] H. A. Simon, The sciences of the artificial, MIT press, 1996.

[20] A. R. Hevner, S. T. March, J. Park, S. Ram, Design science in information systems research, MIS Quarterly 28 (2004). URL: http://www.jstor.org/stable/25148625.

[21] A. R. Hevner, A three cycle view of design science research, Scandinavian journal of information systems 19 (2007). URL: https://aisel.aisnet.org/sjis/vol19/iss2/4.

[22] P. Castells, F. Perdrix, E. Pulido, M. Rico, R. Benjamins, J. Contreras, J. Lorés, Neptuno: Semantic web technologies for a digital newspaper archive, in: The Semantic Web: Research and Applications. ESWS 2004., 2004. doi:10.1007/978-3-540-25956-5_31.

[23] D. B. Ramagem, B. Margerin, J. Kendall, Annoterra: building an integrated earth science resource using semantic web technologies, IEEE Intelligent Systems 19 (2004). doi:10.1109/MIS.2004.3.

[24] G. Kobilarov, T. Scott, Y. Raimond, S. Oliver, C. Sizemore, M. Smethurst, C. Bizer, R. Lee, Media meets semantic web – how the BBC uses DBpedia and linked data to make connections, in: The Semantic Web: Research and Applications, volume 5554, 2009. doi:10.1007/978-3-642-02121-3_53.

[25] J. Borsje, L. Levering, F. Frasincar, Hermes: A semantic web-based news decision support system, in: Proceedings of the 2008 ACM Symposium on Applied Computing, SAC '08, 2008. doi:10.1145/1363686.1364258.

[26] D. Le-Phuoc, H. Q. Nguyen-Mau, J. X. Parreira, M. Hauswirth, A middleware framework for scalable management of linked streams, Journal of Web Semantics 16 (2012). doi:10.1016/j.websem.2012.06.003, the Semantic Web Challenge 2011.

[27] M. A. Martínez-Prieto, C. E. Cuesta, M. Arias, J. D. Fernández, The Solid architecture for real-time management of big semantic data, Future Generation Computer Systems 47 (2015). doi:10.1016/j.future.2014.10.016, special Section: Advanced Architectures for the Future Generation of Software-Intensive Systems.

[28] N. Marz, How to beat the cap theorem, 2011. URL: http://nathanmarz.com/blog/how-to-beat-the-cap-theorem.html.

[29] P. Pääkkönen, D. Pakkala, Reference architecture and classification of technologies, products and services for big data systems, Big Data Research 2 (2015). doi:10.1016/j.bdr.2015.01.001.

[30] J. Kreps, Questioning the lambda architecture, 2014. URL: https://www.oreilly.com/radar/questioning-the-lambda-architecture.

[31] F. Cerezo, C. E. Cuesta, J. C. Moreno-Herranz, B. Vela, Deconstructing the Lambda architecture: An experience report, in: 2019 IEEE International Conference on Software Architecture Companion (ICSA-C), 2019. doi:10.1109/ICSA-C.2019.00042.

[32] M. Gallofré Ocaña, T. Al-Moslmi, A. L. Opdahl, Data privacy in Journalistic Knowledge Platforms, in: Proceedings of the CIKM 2020 Workshops, 2020. URL: http://ceur-ws.org/Vol-2699/paper44.pdf.

[33] S. Martínez-Fernández, C. P. Ayala, X. Franch, H. M. Marques, Benefits and drawbacks of software reference architectures: A case study, Information and Software Technology 88 (2017). doi:10.1016/j.infsof.2017.03.011.

[34] N. Dragoni, S. Giallorenzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, L. Safina, Microservices: Yesterday, Today, and Tomorrow, Springer International publishing, 2017. doi:10.1007/978-3-319-67425-4_12.

[35] A. L. Opdahl, B. Tessem, Ontologies for finding journalistic angles, Software and Systems Modeling (2020). doi:10.1007/s10270-020-00801-w.

[36] T. Al-Moslmi, M. Gallofré Ocaña, Lifting news into a Journalistic Knowledge Platform, in: Proceedings of the CIKM 2020 Workshops, 2020. URL: http://ceur-ws.org/Vol-2699/paper42.pdf.

[37] S. Hellmann, J. Lehmann, S. Auer, M. Brümmer, Integrating NLP Using Linked Data, in: The Semantic Web – ISWC 2013, 2013. doi:10.1007/978-3-642-41338-4_7.

[38] A. Fokkens, A. Soroa, Z. Beloki, N. Ockeloen, G. Rigau, W. R. Van Hage, P. Vossen, NAF and GAF: Linking linguistic annotations, in: Proceedings 10th Joint ISO-ACL SIGSEM Workshop on Interoperable Semantic Annotation, 2014.

[39] W3C, Largetriplestores, 2020. URL: https://www.w3.org/wiki/LargeTripleStores.

[40] M. Galster, P. Avgeriou, Empirically-grounded reference architectures: A proposal, in: Proceedings of the Joint ACM SIGSOFT Conference – QoSA and ACM SIGSOFT Symposium – ISARCS on Quality of Software Architectures – QoSA and Architecting Critical Systems – ISARCS, Association for Computing Machinery, 2011. doi:10.1145/2000259.2000285.

[41] P. Ataei, A. Litchfield, Big data reference architectures, a systematic literature review, in: Australasian Conference on Information Systems, 2021. URL: https://aisel.aisnet.org/acis2020/30.