VISIONARY AND THEORY PAPER

Continuous Metadata in Continuous Integration, Stream Processing and Enterprise DataOps

Mark Underwood[†]

DataKitchen Headquarters, One Broadway, 14th Floor Cambridge, MA 02142, USA

Keywords: Metadata; real time data; stream processing

Citation: Underwood, M.: Continuous metadata in continuous integration, stream processing and enterprise DataOps. Data Intelligence 5(1), 275-288 (2023). doi: doi.org/10.1162/dint_a_00193
Submitted: November 10, 2022; Revised: February 11, 2023; Accepted: February 13, 2023

ABSTRACT

Implementations of metadata tend to favor centralized, static metadata. This depiction is at variance with the past decade of focus on big data, cloud native architectures and streaming platforms. Big data velocity can demand a correspondingly dynamic view of metadata. These trends, which include DevOps, CI/CD, DataOps and data fabric, are surveyed. Several specific cloud native tools are reviewed and weaknesses in their current metadata use are identified. Implementations are suggested which better exploit capabilities of streaming platform paradigms, in which metadata is continuously collected in dynamic contexts. Future cloud native software features are identified which could enable streamed metadata to power real time data fusion or fine tune automated reasoning through real time ontology updates.

1. BACKGROUND AND OBJECTIVE

Longstanding formulations for metadata depict it as static "data about data." A dominant visualization presents metadata as an unmoving, timeless thermometer paired with "real data"—outdoor temperature. Similarly, experimenters may treat independent variables like birthdate and geolocation as static when compared to more fluid dependent measures such as blood pressure. But what software and knowledge management paradigms would be disrupted if the situation were reversed, with metadata showing the greater volume, velocity or variety? DevOps Continuous Integration / Continuous Deployment (CI/CD) has emerged [1] as a leading software engineering approach which integrates agile project management, incremental quality assessment and greater opportunities for automated test and deployment. In principle,

[†] Corresponding author: Mark Underwood (E-mail: dark@computer.org).

DevOps should improve collection, propagation and protection of metadata. Yet the current IEEE/ISO DevOps standard [2] is silent on the matter of metadata integration. The present analysis suggests how continuous metadata, analogous to CI, can be adapted to the CI/CD pipeline and support previously identified techniques for dynamic metadata flows. Such a capability could emerge from attempts to consolidate, better govern and map data flows ["fabric" or "mesh"] within enterprises. Other frameworks, which include constructs such as DataOps, are inspired by new capabilities in stream processing, the cloud native movement, and the need to improve enterprise data access and governance. Continuous metadata may reflect a radical reframing of metadata, but efforts to more fully enable ontology-based reasoning with that stream remain incremental.

2. THE CI/CD LANDSCAPE

Agile methodologies were developed in software engineering in part because it became clear that the goal of requirements-in-advance was unachievable. Agile methodologies such as Scaled Agile [3] emerged from quality processes like Plan-Do-Check-Act [4] and the Extreme Programming initiative. Iterating in smaller work elements with more frequent quality checks has been modestly successful at reframing how software projects are managed. There is considerable acceptance that software features—with associated metadata—are to be added at meticulously smaller stages, or sprints, as complex software is built. This procedure is also followed for update and maintenance activities—even when data and metadata has previously been specified, designed, collected and consumed in production.

Jenkins, an open-source project representative for how it enables CI/CD methods, automates aspects of routine developer tasks related to software build, test and deployment [5]. Jenkins and other CI/CD tools facilitate more frequent software releases—not yet real time, but at least conceptually continuous. CI/CD facilitates other important facets, such as keeping test harnesses, test data, and even metadata in synch with development. There are accompanying enhancements to Integrated Development Environments (e.g., Eclipse) through plugins, some of which are capable of injecting or inheriting metadata from domain-specific modeling tools such as the Business Process Modeling Language (BPMN) [6] or the System Modeling language (SysML) [7]. CI/CD tooling is widely adopted, especially for updating mobile phone apps. It is in the dual contexts of comparatively frequent code releases and large-scale data flows in which CI/CD-enabled continuous metadata arises.

While Scaled Agile attempts to map attributes and processes from enterprise portfolios down to individual sprints, Scaled Agile is light on metadata management and has yet to be studied as an ontology enabler. Nor does agile methodology overcome limitations that result from immature domain requirements representations. It is mostly silent on domain models. Despite these limitations, Scaled Agile does incorporate value stream mapping, from which some form of metadata analysis can proceed. Value streams, when made an integral part of agile processes [8], can flow through software build processes into production deployments, accompany data into data lakes, and to potentially flow continuously in stream processing engines like Kafka or Flink [9].

Data warehouse processes such as dimension reduction can be employed to capture and process metadata streams, potentially in near-real time "over the wire." However, these were processes born of a desire to reduce, not add to big metadata. In some contexts, this can result in more manageable or less noisy signals. Such reductions, however, can require judgments from domain specialists. For instance, in order to retain sufficient attribute granularity to implement NIST Policy Decision Points [10] in real time, and to leverage domain-rich ontologies to inform those decisions [11], care must be taken if attributes are to be merged, discarded or translated. When data sources are fused, not uncommon in Complex Event Processing, Know Your Customer, and situation awareness applications, it can be even less clear which attributes to retain and which to process.

There are several motivations to catalog and govern enterprise data. Gartner advises that "data and analytics leaders who are engaged in the deployment of a data fabric to support composable data as the future of their data management solutions should enrich the data fabric by using existing administrative and management data from systems, platforms and data management as metadata" [12]. Governance motivates regulated enterprises, but observability and manageability are more common drivers. Continuous metadata processes could benefit enterprises in:

- Governance
- Security
- Data quality
- Compliance
- Human / IT resource optimization
- Transparency
- Decision Support
- AI, Machine Learning

Contexts exist in each of these domains where point in time metadata assessments are problematic. For instance, if privacy compliance laws change, additional metadata may need to be collected. New data partnerships bring new metadata, such as merchant product metadata used in connection with customer ecommerce events.

The need for such flexibility is built into some big data platforms. Apache Parquet allows for metadata to be embedded in its native formats [13]. Apache Arrow, designed for real time in-memory analytics, provides a custom metadata field at three levels to provide a mechanism for developers to pass application-specific metadata in Arrow protocol messages. This capability includes Field, Schema, and Message. VMware's Tanzu Application Service (TAS) allows metadata annotation for resources such as spaces and applications. Metadata can describe resource attributes in a TAS for VMs deployment, which can support operations, monitoring, and auditing.

> "For example, you can tag resources with metadata that describes the type of environment they belong to. You could also use metadata to describe app characteristics, such as front end or back end. Other examples include billing codes, points of contact, resource consumption, and information about security or risk" [14].

3. CLASSIFICATION PROPAGATION IN APACHE ATLAS

How does metadata propagate in a CI/CD pipeline? This process is illustrated by a series of Apache Atlas diagrams.[1] The diagrams shown below illustrate how Personally Identifiable Information [PII] is tagged as data moves through various CD processes in Apache Atlas, an open-source project. The Apache Atlas team describes the process as "update classification associated with an entity" [15].

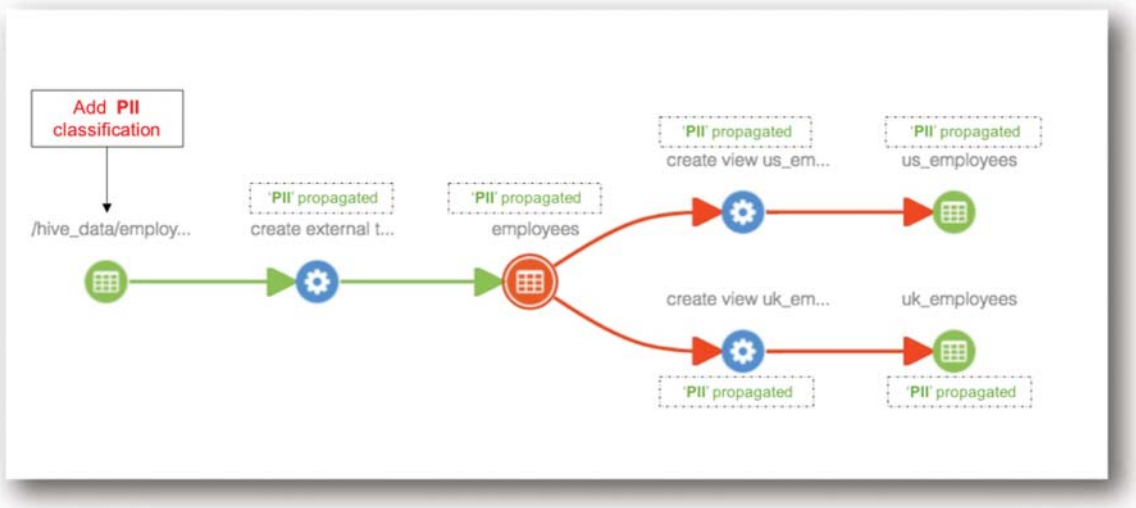

Figure 1. Injection of Personally Identifiable Info (PII) Metadata Propagation Using Apache Atlas.

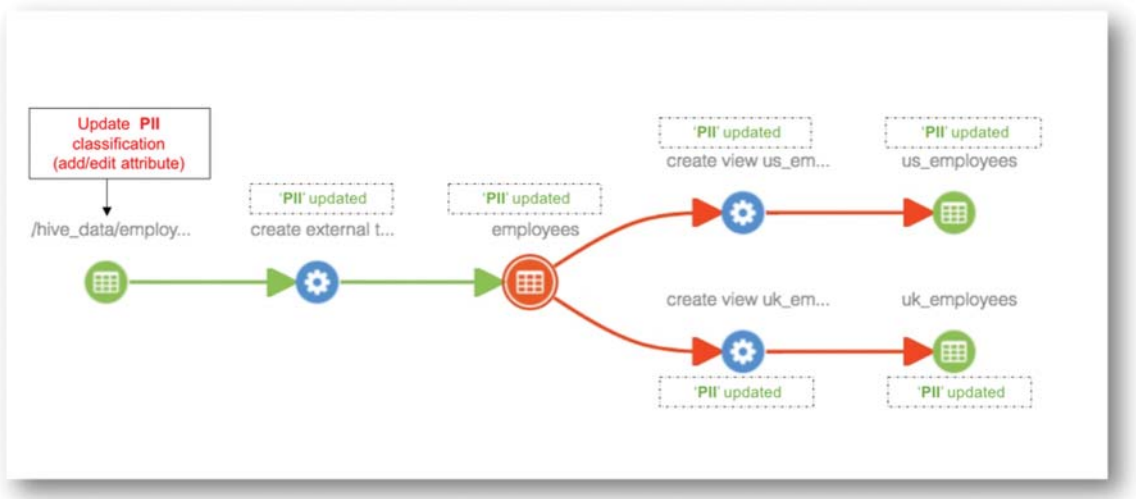

Figure 2. Update of PII Metadata Using Apache Atlas.

[1] Diagrams reproduced here were produced by Apache Atlas contributors and are licensed under Apache License 2.0.

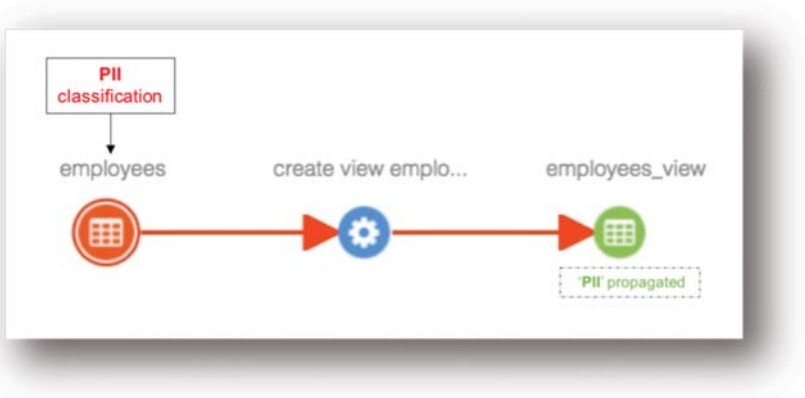

Figure 3. Metadata-enabled filtering of PII using Apache Atlas.

4. OCEAN SOUNDS OR HOW TO BOIL THE OCEAN

Domain specialists are more likely than developers to drive the push for continuous metadata. Consider a use case in oceanography.

In a proposed architecture for *in situ*, real-time ocean sound monitoring, Martinez et al. [16] contemplate a future capability in which "metadata could move beyond sensor deployment and focus on sensor calibration and operational history, providing traceability throughout the whole instrument's life-cycle." The envisioned capability, like many longitudinal investigations, makes assumptions about metadata needs based on current best practices, research hypotheses and the technical capabilities of current instrumentation. When instrumentation changes, metadata streams will need to reflect them. Moreover, as "instrumentation" moves, in many settings, toward machine-to-machine, software-enabled instrumentation, the need for continuous metadata may become clearer.

Appealing as this use case may seem, a reality check is needed. While a CI/CD pipeline managed by Jenkins [or Ansible for Infrastructure as Code [17, 18]] and featuring a systematic, traceable, audit-friendly management of metadata is a compelling model, there are powerful anti-patterns. For instance, it may be difficult in large enterprises to implement across thousands of developers dispersed across international software supply chains—as is found with Boeing and Airbus projects. Conversely, in smaller enterprises, "coding" might only occur in limited settings, such as Excel spreadsheet macros or in-app scripts such as WordPress plugins. There are also "low-code" quasi-applications such as Bubble, Google's AppSheet, and Betty Blocks which can occupy some of the space formerly taken by full-fledged applications. Yet another application-like setting can be seen in autonomously managed analytics, described by some as a variant of Literate Programming [19], on display in the Jupyter [20] community, where computational notebooks can copy, transform, merge and train data [21]. Data can rapidly proliferate as analysts produce subsets, joins and supersets of original datasets. Continuous metadata could be implemented in any these settings, including compute islands once derided as "rogue," but which are emerging as specialized data consumers and producers—but one size may not fit all.

Metadata must be transported with these continuous merges, split and fuse operations for reasons of provenance, quality and manageability. Tagging of training sets, explained Jörg Schad, machine learning specialist at Arrango DB, is important for machine learning: "One needs additional information from the Machine Learning Pipeline besides the actual model: metadata capturing information about datasets, executions, features, models, and other artifacts from our machine learning pipeline" [22]. The ML pipeline is a data engineering variant of agile CI/CD.

Event correlation, data loss prevention and predictive maintenance are examples where advance knowledge of metadata—or metadata granularity—is partial at best. For many enterprise objectives, a 360° view of customers, voters, patients, vendors, and privacy regulations must be restructured as laws and science shift, sometimes in unpredictable ways. In multivariate causality scenarios such as drug trials, a medication could act as a suppressant at one dosage and an accelerant at a different dosage. Or vegetarian trial subjects might experience unanticipated side effects than omnivores. Recognizing such a pattern could be dependent on acquiring related metadata streams from sources not anticipated at the outset.

5. METADATA MADE DYNAMIC

The construct of changing metadata is not new. A 2011 study looked at Continuous Data Mining for XBRL [23]. In data warehouse frameworks, analysts considered "slowly changing dimensions" which can involve metadata [24]. Metadata in flux is a consideration for distributed file designs, where object metadata must be synchronized across networks [25]. There is recognition that metadata associated with science artifacts, such as software artifacts and instrumentation, must be reconciled [26].

Other approaches consider Complex Event Processing [27, 28] as an organizing principle in which event metadata plays a significant role. The more complex and the more time-dependent the metadata, the more likely that traditional, static metadata notions will prove insufficient. Also, existing work has shown how events can be part of ontologies [29], which contribute even more structured and numerous attribute metadata. Metadata is more often seen as a big data collateral artifact, such as embedded schemas in Parquet, rather than a major contributor to big data processing demands itself.

6. RECENT CONTINUOUS METADATA APPROACHES

Less well studied, by contrast, are processes exploited by recent commercial software offerings tackle the potential sprawl of data in large enterprises using a variety of tools [30]. Machine learning, and metadata tagging through CI/CD pipelines are processes within the software development life cycle which can be studied in GitHub repositories [31]. Though some have advocated for consistent use of metadata for software engineering [32], it has yet to be widely adopted. Often developers are unaware that "tags" are in fact metadata; this can result in a lack of metadata discipline, which limits its value for continuous metadata.

One explanation for the lack of systematic metadata adoption lies in the complex relationship between developers and domain experts. As noted by Edwards et al. [33] "… each domain has its own configuration

of classifications, instruments, dates and places." Software engineers are not knowledge engineers; they may be poorly equipped proxies for domain experts. In many current engineering paradigms, traceability from subject matter to code, if provided at all, is weakly instantiated and resists automation. In agile, as in other engineering paradigms, the gap between natural language, pseudo code and code is daunting. Even when developers are enlisted to provide declarative artifacts and metadata through integrated development environments or code generation, the resulting metadata streams may need further curation, if they can be used at all.

Most CI/CD-ready tools are weakly provisioned for metadata. For instance, in the Cloud Native Community, The Update Framework [TUF] is a tool designed to protect the software supply chain when new updates are released [34]. But TUF supports only a limited set of metadata types, which was clearly not designed to support a rich complement of metadata through a build pipeline.

Major cloud platform providers offer scalable, adaptable resources increasingly well-suited to big data analytics. Amazon Web Services (AWS) cites metrics and logging capabilities that leverage its CloudWatch and Embedded Metric Format. These permit AWS users to "record an event with whatever metrics, dimensions and metadata" desired [35]. Despite such extensions, to date rich, standardized domain ontologies are only hinted at [36]. For continuous metadata, as well as other facets of engineering that benefit from automated reasoning from metadata, the gap between metadata theory and practice in major cloud platforms remains wide.

Kubernetes is a dominant cloud orchestration platform developed to support large scale containerization. Containerization allows for Kubernetes workloads to be injected with metadata, which can enable analysts to correlate applications with specific containers, pods, or hosts. Kubernetes accommodates CI/CD practices, though Kubernetes use of metadata is largely ad hoc.

Scalable cloud platforms are friendly to creation of knowledge graphs. Graphs can be manually created, automatically generated or developed through hybrid methods. Such graphs can support metadata analysis, such as in KGBase or OKG-Soft annotations [37], which structures software metadata within an open knowledge graph. No published examples of CI/CD processes directly leveraging knowledge graphs were discovered.

Models may be used to help identify how metadata can be used but can be tagged themselves in order to facilitate discovery, mapping, and inference. These can contribute generally to model-based systems engineering [MBSE], such as envisioned by Object Management Group Object Definition Metamodel™ [38], or by domain-specific models [39]. Scaled Agile explicitly embraces MBSE [40], though its exact CI/CD implementation is more supportive of checklists than code. Regardless of current standards, sense-making for continuous metadata can be facilitated by models, whatever their provenance.

Despite inconsistent progress in CI/CD processes outside of traditional developer communities, there are a few examples outside the discipline. Metadata management figures prominently in one building industry standard [41] and [42]. A resource conforming to the Haystack ontology standard offers "… Haystack

compliant control system vendors to test their control modules and algorithms without the need for a physical building or modifying their hardware and software, enabling faster and cheaper development cycles."

Solutions are nontrivial. Caution is abundant in advice offered by Grafana Labs for its Loki cloud logging stack:

> "Use dynamic labels sparingly. Too many label value combinations leads to too many streams. The penalties for that in Loki are a large index and small chunks in the store, which in turn can actually reduce performance. To avoid those issues, don't add a label for something until you know you need it!" [43].

7. BIG AND CONTINUOUS: RADICAL REFRAMING OF METADATA

This author credits an Amazon Web Services representative with a key insight in the early days of commercial big data. "Guess what's probably our largest database?" he asked. Social Security? No, we were told. It was the footprint of AWS performance, tuning, billing and infrastructure management system. In other words, AWS found itself investing heavily in metrics to characterize what it sells. Because what it sells varies over time, and because those services are often enabled for real time consumption (e.g., AWS Lake Formation [44]), point in time metadata solutions would prove inadequate.

Continuous metadata can improve cybersecurity. Greater integration of metadata with the CI/CD cycle championed in DevOps can support Attribute Based Access Controls (ABAC), safety and privacy through the use of embedded ontologies [45]. Updates to metadata streams as releases produce new software features improve available attribute granularity, which in turn makes policy decision points more effective for enforcing data protection. Yet automated reasoning with continuous metadata is rare, partly because platforms rarely engineer systematic connections between software, data streams and semantics—either through ontologies or through linked data.

There are a few published examples of ontological reasoning using cloud native services. One example is the Data Use Ontology project [46], which leverages, among other resources, the cloud-friendly ELK Reasoner [47]. Still, connective tissue between surging open-source big data / analytics streaming platforms and ontologies remains thin. Continuous metadata cloud support for ontologies in current practice appears to be limited to prototypes [48].

Scattered progress on continuous metadata is in evidence, though not typically identified as such. Future work can be expected to further enable more standardized uses of metadata in CI/CD, computational workbooks and infrastructure management. Optimists might anticipate metadata- and tag-enabled ontologies to play a larger role in what some call Composable Data Services [49]. Whatever the long-term impact of machine learning, it teaches that training sets and learning sources can be varied and voluminous. Gartner analysts Meyer and Zaidi write that semantics enrichment is essential to allow for "ML algorithms to recognize otherwise disparate data as commonly defined and having potential for integration" [12].

Collecting metadata for future analysis may become a default practice as it becomes clear that one-off, point-in-time metadata collection may prove inadequate for many settings.

In earlier work, Obrst, Whitaker & Meng [50] explored "dynamic context" and "knowledge-enhanced objects interpreted on the fly."

> "We envision the future in terms of objects traveling across dynamically determined domain boundaries and application *contexts* arising in the course of execution. Conceptually, a context embodies the mechanism necessary for a system to receive an object at runtime and to examine, interrogate, interpret and determine if it can be dynamically assimilated."

Scalable, parallel cloud services can process continuous metadata, connect and embellish contexts, even refine existing ontologies. As a currently preferred destination for Continuous Integration, labeled Kubernetes namespaces offer one design pattern, but others will emerge. Domain experts will call upon metadata streams to extend domain boundaries and recalibrate the landscape, possibly with each sprint. The allure of autonomous reasoning is too powerful for it to be otherwise.

REFERENCES

[1] Zampetti, F., et al.: CI/CD Pipelines Evolution and Restructuring: A Qualitative and Quantitative Study. In: 2021 IEEE International Conference on Software Maintenance and Evolution (ICSME), pp. 471–82 (2021)
[2] IEEE. IEEE / ISO 2675-2021 Standard for DevOps:Building Reliable and Secure Systems Including Application Build, Package, and Deployment. IEEE Stand (2021). Available at: https://standards.ieee.org/ieee/2675/6830/. Accessed 2 December 2022
[3] Hasselbring, W., Steinacker, G.: Microservice Architectures for Scalability, Agility and Reliability in E-Commerce. In: 2017 IEEE International Conference on Software Architecture Workshops (ICSAW), pp. 243–246 (2017)
[4] What is the Plan-Do-Check-Act Cycle?. ASQ.com. (2020). Available at: https://asq.org/quality-resources/pdca-cycle. Accessed 2 December 2022
[5] LinuxFoundation. Jenkins: The leading open source automation server. Cloud Native Computing Foundation (CNCF). Available at: https://jenkins.io (2023). Accessed 2 January 2023
[6] OMG. Business Process and Model Notation (BPMN). OMB Standards and ISO/IEC 19510. Available at: https://www.omg.org/bpmn/ (2013). Accessed 2 December 2022
[7] OMG. SysML | ISO/IEC 19514:2017. OMG Standards. Available at: https://www.omgsysml.org/ (2017). Accessed 2 December 2022
[8] Putta, A., Paasivaara, M., Lassenius, C.: Adopting Scaled Agile Framework (SAFe): A Multivocal Literature Review. In: Proceedings of the 19th International Conference on Agile Software Development: Companion. New York, NY, USA: Association for Computing Machinery; (XP '18). Available at: https://doi.org/10.1145/3234152.3234164 (2018). Accessed 2 December 2022
[9] Langhi, S., Tommasini, R., Valle, E.D.: Extending Kafka Streams for Complex Event Recognition. In: 2020 IEEE International Conference on Big Data (Big Data), pp. 2190–2197 (2020)
[10] Hu, V.C., et al.: Guide to attribute based access control (abac) definition and considerations. NIST Spec Publ, pp. 162–800 (2014)

[11] Obrst, L., McCandless, D., Ferrell, D.: Fast Semantic Attribute-Role-Based Access Control (ARBAC) in a Collaborative Environment, pp. 703–710 (2012). Available at: citeulike-article-id:14606778. Accessed 2 December 2022
[12] Beyer, M., Zaidi, E.: How to Activate Metadata to Enable a Composable Data Fabric. Gart Anal Rep (2020)
[13] Kim, S.: Document Your Dataset Using Apache Parquet Open Data Science Platform—Part 2.3: Document Your Dataset Using Apache Parquet. Medium. Available at: https://medium.com/geekculture/add-metadata-to-your-dataset-using-apache-parquet-75360d2073bd (2021). Accessed 2 December 2022
[14] VMware. Tanzu Application Service for VMs. Tanzu Docs. Available at: https://docs.pivotal.io/application-service/2-12/adminguide/metadata.html (2021). Accessed 2 December 2022
[15] Apache Atlas. Classification Propagation. Apache Atlas Documentation. Available at: https://atlas.apache.org/2.0.0/ClassificationPropagation.html (2019). Accessed 2 December 2022
[16] Martínez, E., et al.: Metadata-Driven Universal Real-Time Ocean Sound Measurement Architecture. IEEE Access 9, 28282–28301 (2021)
[17] Red_Hat. Ansible: Automation for Everyone. Ansible. Available at: https://ansible.com (2023). Accessed 2 January 2023
[18] Deslauriers, J., Kovacs, J., Kiss, T.: Abstractions of Abstractions: Metadata to Infrastructure-as-Code. 2022 IEEE 19th Int Conf Softw Archit Companion, ICSA-C 2022, pp. 230–232 (2022)
[19] Knuth, D.E.: Literate Programming. Comput J 27(2), 97–111 (1984). Available at: http://dblp.uni-trier.de/db/journals/cj/cj27.html#Knuth84. Accessed 2 December 2022
[20] Project Jupyter. Jupyter: Free software, open standards, and web services for interactive computing across all programming languages. Jupyter.org. Available at: https://jupyter.org (2023) Accessed 2 January 2023
[21] Quaranta, L., Calefato, F., Lanubile, F.: KGTorrent: A Dataset of Python Jupyter Notebooks from Kaggle. In: 2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR), pp. 550–554 (2021)
[22] Schad, J.: From Data to Metadata for Machine Learning Platforms. Inside Big Data. Available at: https://insidebigdata.com/2020/05/15/from-data-to-metadata-for-machine-learning-platforms/ (2020). Accessed 2 January 2023
[23] Pan, D., Pan, Y.: XBRL Metadata Repository and Continuous Data Mining. In: 2011 International Conference on Network Computing and Information Security, pp. 162–165 (2011)
[24] Kimball, R., Ross, M.: The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling. 3rd ed. Indianapolis, IN: Wiley (2013). Available at: https://www.safaribooksonline.com/library/view/the-data-warehouse/9781118530801/
[25] Zhou, J., et al.: A Highly Reliable Metadata Service for Large-Scale Distributed File Systems. IEEE Trans Parallel Distrib Syst 31(2), 374–392 (2020)
[26] Li, K., Lin, X., Greenberg, J.: Software citation, reuse and metadata considerations: An exploratory study examining LAMMPS. Proc Assoc Inf Sci Technol 53(1), 1–10 (2016). Available at: https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/pra2.2016.14505301072 Accessed 2 January 2023
[27] Hammad, R., Wu, C.S.: Provenance as a Service: A Data-centric Approach for Real-Time Monitoring. In: 2014 IEEE International Congress on Big Data, pp. 258–265 (2014)
[28] Kotevska, O., Lbath, A., Gelernter, J.: Event model to facilitate data sharing among services: Closing the gap between research and implementation. In: 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT). pp. 577–584 (2016)
[29] Brown, S., et al.: The Rich Event Ontology. In: Proceedings of the Events and Stories in the News Workshop (Internet). Association for Computational Linguistics, pp. 87–97 (2017). Available at: https://aclanthology.org/W17-2712. Accessed 2 January 2023

[30] Nexla. Build Your Data Mesh: Decentralized, governed, data as a product (2021). Available at: https://www.nexla.com/data-mesh/. Accessed 1 December 2021
[31] Vasilescu, B., et al.: Quality and Productivity Outcomes Relating to Continuous Integration in GitHub. In: Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering. New York, NY, USA: Association for Computing Machinery, pp. 805–816 (2015). (ESEC/FSE 2015). Available at: https://doi.org/10.1145/2786805.2786850. Accessed 2 January 2023
[32] Roantree, M., Kennedy, J.B., Barclay, P.J.: Using a Metadata Software Layer in Information Systems Integration. In: Dittrich KR, Geppert A, Norrie MC, editors. Advanced Information Systems Engineering. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 299–314 (2001)
[33] Edwards, P.N., et al.: Science friction: Data, metadata, and collaboration. Soc Stud Sci 41(5), 667–690 (2011). Available at: https://doi.org/10.1177/0306312711413314. Accessed 2 January 2023
[34] TUF. The Update Framework: Roles and Metadata. Cloud Native Computing Foundation (2021). Available at: https://theupdateframework.io/metadata/. Accessed 2 January 2023
[35] Fischer, M.: Enhancing workload observability using Amazon CloudWatch Embedded Metric Format. Amazon Web Services Blog (2020)
[36] Shankar, N., Emani, P., Srivastava, S.: Easily manage your data lake at scale using AWS Lake Formation Tag-based access control. AWS Blogs (2021). Available at: https://aws.amazon.com/blogs/big-data/easily-manage-your-data-lake-at-scale-using-tag-based-access-control-in-aws-lake-formation/. Accessed 1 Dec 2021
[37] Garijo, D., et al.: OKG-Soft: An Open Knowledge Graph with Machine Readable Scientific Software Metadata. In: 2019 15th International Conference on eScience (eScience), pp. 349–358 (2019)
[38] Object Management Group. Ontology Definition Metamodel (OMG) Version 1.1 (2014). Available at: https://www.omg.org/spec/ODM/. Accessed 2 January 2023
[39] Gil, Y., et al.: Mint: Model integration through knowledge-powered data and process composition (2018)
[40] ScaledAgileInc. Model-Based Systems Engineering in SAFe (2021). Available at: https://www.scaledagileframework.com/model-based-systems-engineering/. Accessed 5 Dec 2021
[41] Fierro, G., et al.: Shepherding Metadata Through the Building Lifecycle. In: Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation. New York, NY, USA: Association for Computing Machinery; pp. 70–79 (2020) (BuildSys '20). Available at: https://doi.org/10.1145/3408308.3427627. Accessed 5 Dec 2021
[42] Fierro, G., et al.: Formalizing Tag-Based Metadata With the Brick Ontology. Front Built Environ 6, 152 (2020). Available at: https://www.frontiersin.org/article/10.3389/fbuil.2020.558034. Accessed 5 Dec 2021
[43] Grafana_Labs. Grafana Loki label best practices. Grafana Labs Documentation (2021). Available at: https://grafana.com/docs/loki/latest/best-practices/. Accessed 13 Dec 2021
[44] Shankar, N., Emani, P., Srivastava, S.: Easily manage your data lake at scale using AWS Lake Formation Tag-based access control. AWS Blogs (2021)
[45] Underwood, M.A.: IDE-Centered ABAC with Models and Ontologies for Safer DevOps. DevSecOps Days (2019). Available at: https://devsecopsdays2019.sched.com/speaker/mark_underwood.202p0aqj. Accessed 5 Dec 2021
[46] Lawson, J., et al.: The Data Use Ontology to streamline responsible access to human biomedical datasets. Cell Genomics 1(2), 100028 (2021). Available at: https://www.sciencedirect.com/science/article/pii/S2666979X21000355. Accessed 2 January 2023
[47] Hagerty, P.: How to Install the ELK Stack on Google Cloud Platform. Logz Blogs (2017). Available at: https://logz.io/blog/elk-stack-google-cloud/. Accessed 12 Dec 2021

[48] Parmelee, M., Obrst, L.: Technologies for Metadata and Ontology Storage. In: Handbook of Metadata, Semantics and Ontologies (Internet). World Scientific Publishing Group, pp. 523–570 (2014). Available at: https://www.worldscientific.com/doi/abs/10.1142/9789812836304_0020. Accessed 2 January 2023

[49] Cunningham, J.: Netflix Data Mesh: Composable Data Processing. Flink Forward Conference (2020). Available at: https://www.youtube.com/watch?v=TO_IiN06jJ4. Accessed 2 January 2023

[50] Obrst, L., Whittaker, G., Meng, A.: Semantic Context for Object Exchange. In: Brezillon, P., Turner, R., Pomerol, J.-C., Turner, E., editors. Workshop on Reasoning in Context for AI Applications AAAI-1999. Orlando, FL: AAAI Press, pp. 80–85 (1999)

AUTHOR BIOGRAPHY

An occasional poet and electric violinist, Mark Underwood is an Information Security Strategic Initiatives Advisor at a US Fortune 500 financial services enterprise. Research interests include model-based systems engineering, intelligent agents and ontology support for domain-specific cybersecurity, privacy and safety. Current Chair of the IEEE 2957 Big Data Governance and Metadata Management Working Group, he also contributed to IEEE/ISO 2675 (DevOps), IEEE 7001 (Standard for Transparency of Autonomous Systems), IEEE 7010 (Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being) and IEEE 7000 Model Process for Addressing Ethical Concerns during System Design.