Investigating Computational Solutions for Metadata Processing in Software Engineering Experiments

Filipe A. Santana, André F. R. Cordeiro, Edson OliveiraJr

[1]Departamento de Informática – Universidade Estadual de Maringá (UEM)
Maringá – PR – Brazil

filipeamadeusantana@gmail.com,

cordeiroandrefelipe@gmail.com, edson@din.uem.br

Abstract. *Metadata consists of elements that describe specific data content. When structured according to standards, metadata improves data interoperability. In this context, it is understood that adopting metadata could contribute to describing experiments in Software Engineering. Furthermore, the use of tools can help in the management. Considering the context, this study includes a Systematic Mapping to identify solutions for the computational processing of metadata in SE experiments. Following the execution of the search strategy, a total of 31 studies were reviewed. The mapping did not achieve satisfactory results. Consequently, a Non-Systematic Review was conducted. In this case, a set of solutions were found.*

1. Introduction

Metadata describes resources to facilitate retrieval and interoperability [Cristina 2010]. When metadata are structured according to a specific standard, they are referred to as metadata standards [NISO 2017]. Experimental Software Engineering (ESE) employs empirical methods to assess Software Engineering (SE) practices and collect related evidence [Wohlin et al. 2012].

The experimental process encompasses scoping, planning, execution, analysis and interpretation, and presentation and package [Wohlin et al. 2012]. The management of artifacts generated from the execution of an experiment is not an easy task [Pfeiffer 2020]. A potential approach to collaborate with managing the generated artifacts could be using metadata [Rocha et al. 2017]. Given this context, this study aims to explore solutions for managing metadata from experiments in SE.

This study involved a Systematic Mapping (SM) and a Non-Systematic Review (NSR). For the SM study, one protocol was established and implemented for the SM, which aimed to explore the literature for solutions related to the computational processing of metadata. During the SM, over 30 papers were analyzed. As a final result, no one solution related was achieved. For the NSR, one kind of solution was considered. A manual search was performed to identify existing metadata editors for registering experimental metadata in SE. The Dublin Core (DC) metadata standard was selected to record experimental metadata in these editors. This standard was analyzed in previous studies [Santana et al. 2023b, Santana et al. 2023a, Santana et al. 2023c].

The subsequent sections provide a detailed account of the SM and NSR. Section **2** presents studies about metadata. Section **3** describes the objectives associated with this study. Section **4** details the methodology applied and the results obtained in the SM. Section **5** presents the methodology, results, and discussion obtained in the NSR. Section **6** describes the threats to the validity of these studies. Section **7** presents final remarks.

2. Research Studies about Metadata

Metadata provides information about data [Hong et al. 2010]. Metadata details the content, format, purpose, and structure of data [Al-Khalifa and Davis 2006, Colace et al. 2003]. Metadata is utilized in various areas and subfields of science [Formenton et al. 2018], and its use can be considered essential for the performance of a system [Cao et al. 2019].

In ESE, metadata represents a potential solution to help manage artifacts related to the experiments [Rocha et al. 2017]. When metadata is organized according to standards, it improves interoperability across diverse platforms and interdisciplinary [Pöttker et al. 2018, Hayslett 2023]. Moreover, metadata standards assist in the management of various types of resources, such as visual, auditory, textual, computational, and educational resources, by facilitating their search, storage, and reuse [Rehak and Mason 2003, Lorist and van der Meer 2001].

3. Objectives

Our research has general and specific objectives. The study aims to find a solution for the computational processing of metadata associated with experiments in SE. The specific objectives include:

- Investigating and evaluating existing solutions in the literature for the computational processing of metadata;
- Identifying and specifying key characteristics necessary for a solution useful to the needs of SE experiments.

4. Systematic Mapping Research Methodology

The methodology adopted for the SM consists of defining research questions, search strategies, research sources, search strings, selection criteria, and protocol evaluation.

4.1. Research Questions

After defining the study's objectives, we established the following research questions:

- **RQ1**: What solutions were found for the computational metadata processing?
- **RQ2**: What are the characteristics of these solutions?

4.2. Search Process

An automated search strategy was defined for the research. The following databases were utilized as sources for the literature review[1]: IEEE Xplore, ACM Digital Library, SpringerLink, Scopus, and ScienceDirect. These databases are globally recognized in Computer Science (CS) and SE.

4.3. Selection Process

In this study, the search string used incorporated the following expressions or terms: "**computer science**" **AND** "**metadata processing**". These expressions were combined using the logical operator "**AND**". Table 1 provides details of the search string applied across the databases. Table 1 presents the search strings used in the ACM, Scopus, ScienceDirect, SpringerLink, and IEEE databases. Initially, the SM considered the search for solutions in CS, with the aim of analyzing which solutions could be applied or adapted in SE

Due to the large number of retrieved papers, it was necessary to apply a few filters across the databases as follows:

[1] Available databases: IEEE Xplore, ACM Digital Library, SpringerLink, Scopus, ScienceDirect.

Table 1. Search strings and results before and after applying filters in the bibliographic databases.

Database	Query	Retrieved Papers	Filtered Papers
ACM	“computer science” **AND** “metadata processing”	41	19
Scopus	“computer science” **AND** “metadata processing”	159	48
ScienceDirect	“computer science” **AND** “metadata processing”	57	38
SpringerLink	“computer science” **AND** “metadata processing”	277	49
IEEE	“computer science” **AND** “metadata processing”	16	13

- **SpringerLink:** discipline: computer science; content type: article;
- **ACM:** content type: proceedings; document type: research article;
- **Scopus:** subject area: computer science and engineering; document type: article language: English;
- **ScienceDirect:** article type: research articles; subject areas: computer science and engineering;

4.4. Extraction Process

During the processes of reading, analyzing, and selecting relevant studies, different criteria were applied to guide the inclusion and exclusion of the materials. The criteria used for these decisions are outlined below. For **inclusion**, the following criteria were considered: **study that describes metadata processing**.

For **exclusion**, the following criteria were considered: **study not related to the field of CS**; **study not related to the field of SE**; **study not related to metadata processing**; **study not written in English**; **studies not published in conferences or journals**; **duplicate studies**; **studies unavailable even after contacting the authors**; **secondary studies**.

4.5. Analysis Process

This subsection outlines the systematic approach used to evaluate the relevance of the studies included in the research. Based on this approach, the protocol employed to evaluate the relevance of the studies involves the following steps:

1. Review the study’s title
2. Read the study’s abstract
3. Examine the study’s keywords
4. Apply inclusion criteria. If any inclusion criterion is not met, the study should be discarded.

4.6. Results and Discussion

After applying the inclusion and exclusion criteria, 167 studies were initially identified. Subsequently, studies that contained the term ”metadata” in the title, keywords, or main text were selected, resulting in a refined set of 31 studies. A subsequent analysis was performed to identify and remove any duplicate articles. Following this, an in-depth examination of the remaining studies was conducted to identify solutions for the computational processing of metadata. Table 2 provides an overview of the selected studies and their proposed solutions for metadata management.

Table 2. Overview of Studies and Proposed Solutions for Metadata Management

Title	Proposed Solution
A GPU-Accelerated In-Memory Metadata Management Scheme for Large-Scale Parallel File Systems	Proposes a GPU-accelerated metadata management scheme for large-scale parallel file systems.

Title	Proposed Solution
A Reference Architecture for Organizing the Internal Structure of Metadata-Based Frameworks	Introduces a pattern language and a reference architecture to enhance the internal structure of metadata-based frameworks.
AdaM: An Adaptive Fine-Grained Scheme for Distributed Metadata Management	Presents AdaM, an adaptive fine-grained metadata management scheme utilizing deep reinforcement learning to address the trade-offs associated with time-varying access patterns.
Adaptive Metadata Rebalance in Exascale File System	Proposes a metadata rebalance model aimed at minimizing failures during the metadata rebalance period, validated through cost analysis.
An Asynchronous Traversal Engine for Graph-Based Rich Metadata Management	Suggests optimizations for an asynchronous traversal engine, including traversal-affiliate caching and execution merging, along with a general traversal language for describing patterns in property graph-based metadata management.
An Effective Grouping Method for Unstructured Data Based on Swift	Utilizes Swift with a grouping-based machine learning approach and prefetching cache strategy to enhance access performance for unstructured data.
An Adaptive Metadata Management Scheme Based on Deep Reinforcement Learning for Large-Scale Distributed File Systems	Proposes a self-adaptive metadata cache policy that dynamically integrates server-side and client-side cache management strategies, alongside a distributed metadata processing 2PC protocol to ensure data consistency.
An Efficient Ring-Based Metadata Management Policy for Large-Scale Distributed File Systems	Introduces the AngleCut hashing scheme to partition the metadata namespace tree, catering to large-scale distributed storage systems.
Analytical Metadata Modeling for Next Generation BI Systems	Proposes SM4AM, an RDF-based semantic metamodel for metadata management, advocating ontological metamodeling as the appropriate solution for the heterogeneity of data models in BI 2.0.
ARCE: Towards Code Pointer Integrity on Embedded Processors Using Architecture-Assisted Run-Time Metadata Management	Describes ARCE, a solution based on a shallow 3-stage pipeline processor, demonstrating its effectiveness against code pointer attack vectors.
Effective Metadata Management in Exascale File System	Proposes a high-performance metadata management model and system designed to overcome existing limitations in exascale file systems.
Efficient Processing of Secured XML Metadata	Develops an approach to enable efficient searching of encrypted XML metadata.
Efficient TVA Metadata Encoding for Mobile and Ubiquitous Content Services	Proposes a new TVA metadata encoding scheme, optimized for mobile and ubiquitous devices, based on Efficient XML Interchange (EXI).
GraphTrek: Asynchronous Graph Traversal for Property Graph-Based Metadata Management	Introduces GraphTrek, a general asynchronous graph traversal engine designed for processing rich metadata management in native graph databases.
Metadata Distribution and Consistency Techniques for Large-Scale Cluster File Systems	Presents the Dynamic Dir-Grain (DDG) metadata distribution policy, balancing namespace locality and load distribution through dynamic partitioning.
Mlock: Building Delegable Metadata Service for the Parallel File Systems	Proposes a delegable metadata service (DMS) to reduce latency in metadata accesses and optimize performance for small files.
Optimal Metadata Replications and Request Balancing Strategy on Cloud Data Centers	Proposes strategies for metadata replication and load balancing to minimize mean response time in cloud data centers.
Prefetching-Based Metadata Management in Advanced Multitenant Hadoop	Proposes a prefetching-based approach to improve metadata management performance in a multitenant Hadoop environment.
Taking Advantage of Metadata Semantics: The Case of Learning-Object-Based Lesson Graphs	Suggests leveraging lesson graph semantics through a context diffusion approach to propagate metadata-based processes along the graph edges.
Text Mining Using Metadata for Generation of Side Information	Proposes a clustering approach for text data with side information, utilizing pattern discovery techniques and both unsupervised and supervised learning to enhance clustering quality based on text metadata and side information.

Table 2 presents the selected studies regarding their titles and the proposed solution for metadata processing. Nine studies did not present any solutions for metadata processing and were therefore discarded from the analysis.

Given that the solutions in the reviewed studies do not directly the objectives of this study, we conducted a new, non-systematic review of existing metadata editors to effectively address the gaps identified in the literature.

5. Non-Systematic Review Research Methodology

Due to time constraints, a full systematic review was not feasible, so we conducted an NSR to explore metadata editors (one kind of solution) for SE experiments using the DC standard. Unlike the SM, the NSR search lacked a defined protocol and was limited to the first four pages of Google, using the terms "metadata editor" and "Dublin Core editor." We categorized the identified tools into two groups: those supporting DC and document editors.

5.1. Results

Following the results from the search for metadata editors, a more detailed evaluation was undertaken. This assessment focused on the features offered by each editor, with a ranking based on the number of supported features. Editors that did not support DC were excluded from this evaluation. The features of DC editors are detailed in Table 3.

Table 3. Comparison of metadata editors and their features

Feature	Dublin Core Generator	DC-dot	GeoNetwork	DC-Template	Omeka
Supports DC	Yes	Yes	Yes	Yes	Yes
Simple Generator	Yes	Yes	No	Yes	Yes
Advanced Generator	Yes	No	Yes	No	Yes
XML Output	Yes	Yes	Yes	Yes	Yes
XHTML Output	Yes	Yes	No	No	No
HTML Output	Yes	Yes	No	Yes	No
XMP Output	No	No	No	No	No
ZIP Output	No	No	Yes	No	No
PDF Output	No	No	Yes	No	Yes
CSV Output	No	No	Yes	No	Yes
XLSX Output	No	No	No	No	Yes
RDF Output	No	Yes	No	Yes	Yes
Insert File	No	No	Yes	No	Yes
Metadata File Output	No	No	Yes	No	Yes
Automatic Metadata Reading	No	Yes	Yes	No	Yes
API	No	No	Yes	No	Yes
Web Editor	Yes	Yes	No	Yes	No
Download Required	No	No	Yes	No	Yes
Open Source	Yes	Yes	Yes	No	No
GitHub/Documentation	Yes	Yes	Yes	No	Yes

Initially, a quantitative analysis was conducted to enumerate the features of each metadata editor. Following this assessment, a more detailed analysis was performed, focusing on their effectiveness in managing metadata, as well as their advantages, disadvantages, and ease of use. Based on these evaluations, the three editors selected for detailed assessment were **DC Generator**[2], **GeoNetwork**[3], and **Omeka**[4]

Subsequently, we conducted manual tests on these editors to assess their simplicity. The testing process involved downloading the editors, if necessary, configuring them, registering metadata for a series of SE experiments, and evaluating the outputs of each editor. Based on these tests, the editors were ranked in terms of simplicity. The DC Generator emerged as the simplest, followed by Omeka and GeoNetwork.

Following this classification, we decided to work with DC Generator. We started an additional technical analysis and an evaluation of the editor, based on the **Software Product Quality Model, SquaRE – ISO/IEC 25010:2011** [ISO 2011, Wazlawick 2013]. The evaluation of DC Generator focused on **Functional Suitability**, **Reliability**, **Usability**, and **Performance Efficiency**. This evaluation is in course by researchers with experience in ESE. Figure 1 illustrates the simple Dublin Core Generator interface.

5.2. Discussion of Results

Following the SM and NSR, we identified several tools that met the criteria for further investigation. Among these, DC Generator, GeoNetwork, and Omeka were selected for a more detailed assessment due to their feature sets and relevance to metadata management.

[2] https://nsteffel.github.io/dublin_core_generator/generator_nq.html
[3] https://geonetwork-opensource.org/downloads.html
[4] https://www.omeka.net

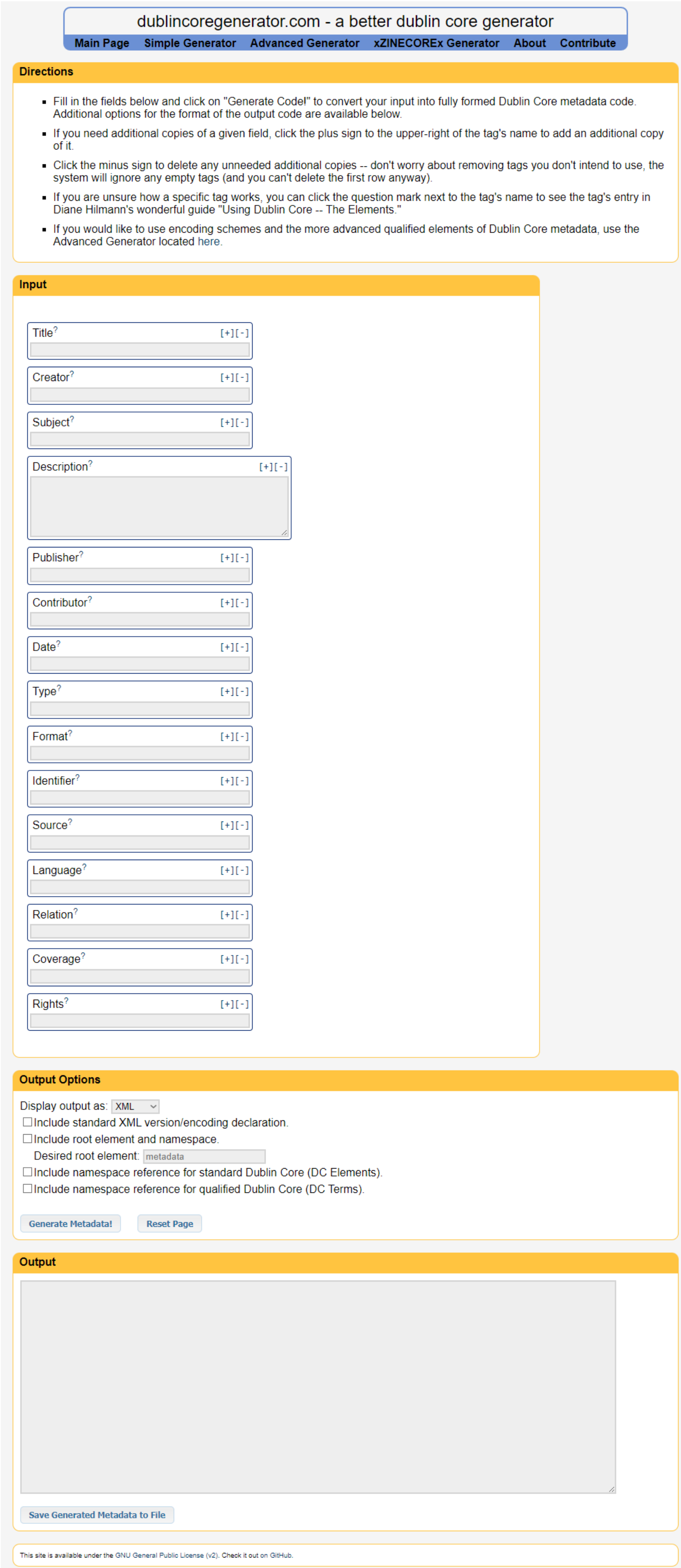

Figure 1. Dublin Core Generator

In terms of usability, GeoNetwork proved to be the most challenging due to the requirement of installing an Apache or Jetty server. Omeka necessitates creating an account on the website and setting up a collection before editing metadata. Additionally, it imposes restrictions related to the website and 500 MB of storage. In contrast, DC Generator does not require downloads or account creation, making it the simplest editor. However, it has the drawback of needing to manually copy the generated output to a blank XML/HTML/XHTML[5] file, as the save button is non-functional.

Despite this inconvenience, DC Generator was the quickest to manipulate among the three editors. Given the complexity of using GeoNetwork (and its high computational resource consumption) and the account creation requirement for Omeka, we opted to proceed with the DC Generator editor. This choice was made as the prerequisites for the other editors could potentially deter researchers from utilizing them.

6. Threats to Validity

During the execution of the studies, especially the SM, the following threats to validity can be identified [Ampatzoglou et al. 2019].

6.1. Systematic Mapping

This study has taken into account and addressed the following categories of threats:

- **Selection Validity:** encompasses threats that could undermine studies' search and evaluation process. Notable threats include the selection of databases and the formulation of search strings. By focusing on databases relevant to CS and conducting iterative searches, the study tried to reduce the risk of selection bias [Nakagawa et al. 2017], and multiple tests were conducted using the established search strings to ensure comprehensiveness.
- **Data Validity:** involves threats associated with data extraction and analysis. Examples include errors during data collection and discrepancies in publication. The methodology for addressing data validity is sound, as it incorporates multiple assessments by different authors to verify the accuracy of the data. This approach enhances the reliability of the data and reduces the potential for individual biases or errors.
- **Research Validity:** addresses threats related to the overall research design, such as the generalizability of results and the adequacy of research coverage. To minimize these issues, the study adhered to established protocols for SM as outlined in the literature [Kitchenham et al. 2007]; [Petersen et al. 2008].

6.2. Non-Systematic Review

The NSR approach introduced threats to validity. Firstly, the manual search could have provided only a partial view of available metadata editors. Relevant editors may have been excluded if they did not appear on the initial pages of Google search results. This limitation is particularly significant because it implies that some potentially valuable editors were overlooked due to the Google search engine ranking.

7. Final Remarks

This study presented two investigations to identify metadata editors to register metadata in experiments in SE. Initially, one SM was conducted. In the final, 31 studies were

[5]More information available at: `https://nsteffel.github.io/dublin_core_generator/index.html`

evaluated, of which 21 studies were selected; however, they proved unsatisfactory. Subsequently, one NSR was conducted to identify existing metadata editors available on the web that could be adapted for our research. Fortunately, we obtained promising results and have been working with them.

The NSR conducted for metadata editors was useful for initial exploration but had inherent limitations that were not minimized in enough way. Future reviews should employ a systematic approach to mitigate these threats and ensure a more comprehensive assessment of available metadata editors.

A survey with software engineering researchers is underway to evaluate the selected editor. We believe that this study has the potential to produce benefits, such as providing a more comprehensive description of datasets generated or reused in SE experiments, facilitating better conditions for reusing this data across various tools and projects, and simplifying the processes of repeating, replicating, or reproducing experiments conducted in SE. It also holds relevance for research investigating the application of metadata standards, such as DC, in ESE.

Acknowledgements

Edson OliveiraJr thanks CNPq/Brazil Grant #311503/2022-5.

References

Al-Khalifa, H. S. and Davis, H. C. (2006). The evolution of metadata from standards to semantics in e-learning applications.

Ampatzoglou, A. et al. (2019). Identifying, categorizing and mitigating threats to validity in software engineering secondary studies. *Information and Software Technology*, 106:201–230.

Bhanuse, S. S., Kamble, S. D., and Kakde, S. M. (2016). Text mining using metadata for generation of side information. *Procedia Computer Science*, 78:807–814.

Cao, S., Gao, Y., Gao, X., and Chen, G. (2019). Adam.

Cha, M.-H., Kim, D.-O., Kim, H.-Y., and Kim, Y.-K. (2016). Adaptive metadata rebalance in exascale file system. *The Journal of Supercomputing*, 73(4):1337–1359.

Cha, M.-H., Lee, S.-M., Kim, H.-Y., and Kim, Y.-K. (2019). Effective metadata management in exascale file system. *The Journal of Supercomputing*, 75(11):7665–7689.

Chen, Z.-G., Liu, Y.-B., Wang, Y.-F., and Lu, Y.-T. (2021). A gpu-accelerated in-memory metadata management scheme for large-scale parallel file systems. *Journal of Computer Science and Technology*, 36(1):44–55.

Colace, F., Santo, M. D., Molinara, M., and Percannella, G. (2003). An automatic learning contents selector based on metadata standards. In *Proceedings of the ITRE*, volume 1, pages 431–435, Newark, USA. IEEE.

Cristina, R. (2010). Metadados como elementos do processo de catalogação. *Aleph UCLA Undergraduate Research Journal for the Humanities and Social Sciences*.

Dai, M. and Zhu, D. (2018). An effective grouping method for unstructured data based on swift.

Dong, D., Carns, P., Ross, R., Jenkins, J., Blauer, K., and Chen, Y. (2015). Graphtrek: Asynchronous graph traversal for property graph-based metadata management.

Dong, D., Carns, P., Ross, R., Jenkins, J., Muirhead, N., and Chen, Y. (2016). An asynchronous traversal engine for graph-based rich metadata management. *Parallel Computing*, 58:140–156.

Feng, L. and Jonker, W. (2003). Efficient processing of secured xml metadata. In Meersman, R. and Tari, Z., editors, *On The Move to Meaningful Internet Systems 2003: OTM 2003 Workshops*, pages 704–717, Berlin, Heidelberg. Springer Berlin Heidelberg.

Formenton, D. et al. (2018). Os padrões de metadados como recursos tecnológicos para a garantia da preservação digital. *Biblios: Journal of Librarianship and Information Science*, (68):82–95.

Gao, Y., Gao, X., Yang, X., Liu, J., and Chen, G. (2019). An efficient ring-based metadata management policy for large-scale distributed file systems. *IEEE Transactions on Parallel and Distributed Systems*, 30(9):1962–1974.

Guerra, E., Alves, F., Kulesza, U., and Fernandes, C. (2013). A reference architecture for organizing the internal structure of metadata-based frameworks. *Journal of Systems and Software*, 86(5):1239–1256.

Ha, Y.-g. and Jang, B.-s. (2015). Efficient tva metadata encoding for mobile and ubiquitous content services. *Pervasive and Mobile Computing*, 24:91–100.

Hayslett, M. (2023). Libguides: Metadata for data management: A tutorial: Intro. `https://guides.lib.unc.edu/metadata`.

Hong, K., Hu, J., and Chen, X. (2010). Research on information metadata standards of knowledge organization - a case study of chinese digital library. *Proceedings of the International Conference on Computer and Information Science (ICIS)*.

Huang, X., Gao, Y., Zhou, X., Gao, X., and Chen, G. (2023). An adaptive metadata management scheme based on deep reinforcement learning for large-scale distributed file systems. *IEEE/ACM Transactions on Networking*, 31(6):2840–2853.

ISO (2011). Iso/iec 25010:2011 - systems and software engineering – systems and software quality requirements and evaluation (square) – system and software quality models. Accessed: 2024-08-30.

Kitchenham, B. et al. (2007). Guidelines for performing systematic literature reviews in software engineering.

Lorist, H. H. J. and van der Meer, K. (2001). Standards for digital libraries and archives: Digital longevity. pages 89–98.

Motelet, O., Baloian, N., and Pino, J. A. (2008). Taking advantage of metadata semantics: the case of learning-object-based lesson graphs. *Knowledge and Information Systems*, 20(3):323–348.

Nakagawa, E. Y. et al. (2017). Revisão sistemática da literatura em engenharia de software: teoria e prática.

Nguyen, M. C., Won, H., Son, S., Gil, M.-S., and Moon, Y.-S. (2017). Prefetching-based metadata management in advanced multitenant hadoop. *The Journal of Supercomputing*, 75(2):533–553.

NISO (2017). Understanding metadata: What is metadata, and what is it for?: A primer — niso website. `https://www.niso.org/publications/understanding-metadata-2017`.

Petersen, K. et al. (2008). Systematic mapping studies in software engineering. In *12th International Conference on Evaluation and Assessment in Software Engineering (EASE) 12*, pages 1–10.

Pfeiffer, R.-H. (2020). What constitutes software? an empirical, descriptive study of artifacts. In *Proceedings of the 17th International Conference on Mining Software Repos-*

itories, MSR '20, page 481–491, New York, NY, USA. Association for Computing Machinery.

Pöttker, L. M. V., Ferneda, E., and Moreiro-González, J. A. (2018). Mapeamento relacional entre padrões de metadados educacionais. *Perspectivas em Ciência da Informação*, 23(3):25–38.

Rao, J., Ao, T., Dai, K., and Zou, X. (2019). Arce: Towards code pointer integrity on embedded processors using architecture-assisted run-time metadata management. *IEEE Computer Architecture Letters*, 18(2):115–118.

Rehak, D. R. and Mason, R. (2003). Engaging with the learning object economy. In Littlejohn, A., editor, *Reusing online resources: a sustainable approach to e-learning*, pages 22–30. Kogan Page, London.

Rocha, L., Sales, L., and Sayão, L. (2017). Descrever para preservar: metadados como ferramenta para gestão de dados de pesquisa. *ISKO Brasil*, 5(2):194–201.

Santana, F., Cordeiro, A., and OliveiraJr, E. (2023a). Dublin core for recording metadata of experiments in software engineering: A survey. In *Anais da VII Escola Regional de Engenharia de Software*, pages 169–177, Porto Alegre, RS, Brasil. SBC.

Santana, F., Cordeiro, A., and OliveiraJr, E. (2023b). Metadata standards: a review towards modeling experiments. In *Anais da VII Escola Regional de Engenharia de Software*, pages 159–168, Porto Alegre, RS, Brasil. SBC.

Santana, F., Cordeiro, A., and OliveiraJr, E. (2023c). Use of the dublin core standard to express open metadata related to software engineering experiments. In *Anais do III Workshop de Práticas de Ciência Aberta para Engenharia de Software*, pages 1–5, Porto Alegre, RS, Brasil. SBC.

Varga, J., Romero, O., Pedersen, T. B., and Thomsen, C. (2018). Analytical metadata modeling for next generation bi systems. *Journal of Systems and Software*, 144:240–254.

Wazlawick, R. S. (2013). *Engenharia de Software*. Elsevier, Rio de Janeiro, Brazil, 1st edition.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M., Regnell, B., and Wesslén, A. (2012). *Experimentation in Software Engineering*. Computer Science. Springer Berlin Heidelberg.

Xiong, J., Hu, Y., Li, G., Tang, R., and Fan, Z. (2011). Metadata distribution and consistency techniques for large-scale cluster file systems. *IEEE Transactions on Parallel and Distributed Systems*, 22(5):803–816.

Zeng, Y. and Veeravalli, B. (2014). Optimal metadata replications and request balancing strategy on cloud data centers. *The Journal of Supercomputing*, 74(11):4512–4525.

Zhou, C., Li, X., and Zhou, C. (2019). Efficient metadata management and intelligent scheduling in exascale systems. *Future Generation Computer Systems*, 82:100–108.