Submission date: 10/07/2024
Resubmission date: 04/10/2024
Camera ready submission date: 10/12/2024

1st round notification: 09/25/2024
Available online: 26/12/2024
Publication date: 26/12/2024

Section: Extended versions of papers presented at the XX Brazilian Symposium on Information Systems (SBSI 2024)

Ecore4PROV-DM: A Metamodel for Enhancing Data Provenance Adoption in Information Systems

Marcos Alves Vieira[1,2], **Gislainy Crisostomo Velasco**[2], **Sergio T. Carvalho**[2]

[1]Instituto Federal Goiano (IF Goiano) – Campus Iporá
Iporá, Goiás – Brazil

[2]Universidade Federal de Goiás (UFG) – Instituto de Informática (INF)
Goiânia, Goiás – Brazil

marcos.vieira@ifgoiano.edu.br, gislainycrisostomo@discente.ufg.br,
sergiocarvalho@ufg.br

Abstract. *Effective management of data provenance is essential in Information Systems, particularly for data-intensive applications. Despite the W3C PROV family of documents establishing a standard for representing provenance, integrating this information into software development processes remains a significant challenge. This paper addresses the problem by introducing the Ecore4PROV-DM metamodel, developed using Model-Driven Engineering techniques to align with the W3C PROV data model (PROV-DM). The metamodel's application is demonstrated through real-world scenarios, including the Urban Observatory project at Newcastle University. Evaluated using a subset of the Metamodel Quality Requirements and Evaluation (MQuaRE) framework, focusing on three key quality requirements, Ecore4PROV-DM exhibits high accuracy and completeness, making it a robust tool for provenance modeling. By bridging the gap between the conceptual richness of W3C PROV-DM and practical implementation needs, Ecore4PROV-DM facilitates precise provenance representation and seamless integration into diverse Information Systems.*

Keywords. *Data Provenance; Model-Driven Engineering; W3C PROV; W3C PROV-DM; Metamodel Evaluation.*

1. Introduction

In contemporary Information Systems development, the traceability and understanding of data provenance play a crucial role in ensuring transparency, accountability, and reproducibility [Herschel et al. 2017, Hu et al. 2020]. As these systems become more data-intensive, the need for robust provenance models grows increasingly important. The W3C PROV family of documents has emerged as a standard for representing, serializing, and storing data provenance, providing a foundation for interoperability across diverse domains [Moreau et al. 2013a].

Despite the availability of the W3C PROV standard, developers face challenges when integrating provenance information into Information Systems. These challenges

Cite as: Vieira, M. A., Velasco, G. C. & Carvalho, S. T. (2024). Ecore4PROV-DM: A Metamodel for Enhancing Data Provenance Adoption in Information Systems. iSys: Revista Brasileira de Sistemas de Informação (Brazilian Journal of Information Systems), , Vol 17(1), 13:1-13:31. DOI: 10.5753/isys.2024.4691

include the complexity of mapping abstract provenance concepts to software models, ensuring compatibility with existing systems, and automating the generation and use of provenance data during runtime. Additionally, aligning the conceptual richness of W3C PROV with the technical demands of real-world systems presents challenges in developing efficient and scalable solutions.

Model-Driven Engineering (MDE) offers a promising approach by providing a systematic and standardized framework for representing and managing provenance data [Kinderen et al. 2017]. The Eclipse Modeling Framework (EMF), a core technology for building MDE-based tools, provides model validation, code generation, and integration capabilities, facilitating the automation of model-driven processes, including provenance representation in Information Systems.

The primary research question addressed by this paper is: *How can a metamodel aligned with the W3C PROV standard enhance the integration of provenance information into Information Systems, specifically addressing traceability and reproducibility challenges in data-intensive applications?* The proposed Ecore4PROV-DM metamodel answers this question by providing a structured and compliant approach that meets the needs of developers integrating provenance information into their systems. This research bridges the gap between the conceptual framework of W3C PROV and the practical requirements of Information Systems, offering a solution for accurate modeling and traceability of data provenance. Designed to closely align with the components of the W3C PROV data model (PROV-DM), the metamodel incorporates MDE techniques, providing a formalized and systematic method to manage provenance information within the context of established W3C standards.

The evaluation of Ecore4PROV-DM focuses on three key Metamodel Quality Requirements (MQRs) from the MQuaRE framework [Kudo 2021, Kudo et al. 2020a, Kudo et al. 2020b]. These requirements were selected based on their relevance to assessing the accuracy, completeness, and usability of the metamodel in representing provenance information aligned with the W3C PROV standard.

The Ecore4PROV-DM metamodel allows developers to consistently and automatically incorporate provenance information into their systems. By adhering to W3C PROV standards, this approach enables better alignment between conceptual models and practical implementation needs in data-driven environments. With strict compliance to the W3C PROV-DM standard, the metamodel establishes a structured and standardized framework for representing and managing provenance information. This simplifies the creation of provenance models, ensuring compliance with PROV-DM constraints and facilitating seamless integration into software systems. Aligned with these constraints, the models are well-suited for serialization and computer processing. The integration of MDE further supports the development process, offering a formalized approach that streamlines the implementation of provenance data. This alignment between the conceptual richness of W3C PROV-DM and the practical needs of software development ensures compliance with standards, addressing the challenges of integrating provenance features into data-intensive applications.

The paper is organized as follows: Section 2 presents background details on data provenance, the W3C PROV family of documents, Model-Driven Engineering, and introduces the MQuaRE framework, which served as the basis for the quality evaluation of the metamodel; Section 3 explores related research; Section 4 provides an in-depth overview of the Ecore4PROV-DM metamodel; Section 5 demonstrates the practical application of the metamodel through the presentation of two use cases; Section 6 evaluates the Ecore4PROV-DM in the context of the MQuaRE framework; and finally, Section 7 concludes this work by summarizing key contributions and suggesting future research directions.

2. Background

This section provides essential background information to facilitate the understanding of concepts and technologies discussed in this paper. It encompasses key topics such as data provenance, the W3C PROV family of documents, Model-Driven Engineering (MDE), Eclipse Modeling Framework (EMF), and the MQuaRE framework, which serves as the basis for evaluating the quality of our proposed metamodel.

2.1. Data Provenance

Provenance encompasses information delineating the production process of a product, whether a data entity or a physical object [Herschel et al. 2017]. As defined by the W3C Provenance Working Group[1], it refers to “information about entities, activities, and people involved in producing a piece of data or thing, which can be used to form assessments about its quality, reliability, or trustworthiness.” This term traces its origins to the Latin *prōvenīre*, meaning “coming from.” Specifically, “data provenance” involves mechanisms and techniques used to acquire and record information about the origin of data and the transformations shaping its current state, commonly referred to as “provenance data,” “provenance information,” or “provenance record.”

Data provenance is valuable in numerous industries, as it serves multiple purposes [Glavic 2021, Pérez et al. 2018]. In open information systems, it assists in determining the origin of data and identifying responsible parties. In scientific research, provenance offers insights into the methodology used to obtain results, guaranteeing transparency and reproducibility. In news reporting, it plays a vital role in verifying references and sources, enhancing trustworthiness and credibility. In legal contexts, provenance facilitates licensing, attribution of documents and data, and privacy management. The domain of IoT sees data provenance being utilized in a variety of areas, including supply chains, health monitoring, digital forensics, and intelligent services [Hu et al. 2020]. Provenance is crucial in these fields, ensuring accountability, authenticity, and informed decision-making [Herschel et al. 2017].

Data provenance is crucial for confirming data authenticity, which facilitates data reuse by providing detailed records of the data generation process, addressing how, why, where, when, and by whom the data was created. This metadata is fundamental for effective data management and is essential in areas where data integrity is paramount. In contexts with multiple handlers from creation to consumption, such as data-intensive research, provenance metadata ensures the quality and accuracy of the data for researchers

[1] https://www.w3.org/2011/prov/

who did not collect it. Publishing data with comprehensive provenance metadata allows for broader verification and reuse, helping others assess the quality, reproducibility, and reliability of the data. This documentation gives stakeholders greater confidence in the results derived from such data [ARDC, Australian Research Data Commons 2022].

Methods for capturing and documenting provenance range from simple text annotations to complex metadata schemas. Basic provenance can be recorded in README files that describe data collection and processing details. Internally, software tools and systems like Kepler [Ludäscher et al. 2006], Galaxy [Community 2022], and Taverna [Wolstencroft et al. 2013] automatically document data provenance during computational workflows, which is crucial for scientific computing's reproducibility and transparency. This internally captured provenance is essential for system users and can be exported to dedicated repositories for broader access. Provenance information supports both machine-to-machine interactions and user-friendly interfaces, enhancing data journey understanding. Advanced visualizations from systems like VisTrails [Callahan et al. 2006] improve provenance data interpretability for both specialists and stakeholders. In structured data management environments, provenance may adhere to standards like Dublin Core [Wolf et al. 1998] or ISO 19115-3 [ISO Central Secretary 2023], providing a formalized framework for data documentation. However, these standards are often tailored to specific domains, such as metadata for documents (Dublin Core) or geographic information (ISO 19115-3). In contrast, the W3C's PROV Family of Documents offers a more flexible and domain-agnostic specification for modeling data provenance across various fields. Its detailed structure for representing relationships between entities, activities, and agents makes it particularly suited for complex use cases requiring extensive provenance tracking, which was the focus of this research.

2.2. W3C PROV Family of Documents

The W3C PROV Family of Documents, introduced by the W3C Provenance Working Group, presents a comprehensive framework comprising a model, serializations, and complementary definitions. This framework establishes standardized representations for seamless exchange of provenance data across different platforms. The PROV standards aim to facilitate the easy publication and exchange of provenance information across the Web and information systems, supporting representation and exchange of provenance data in formats such as RDF and XML. Additionally, PROV documents provide guidelines for accessing, validating, and correlating provenance information with standards like Dublin Core [Wolf et al. 1998].

In W3C PROV, provenance is represented by three central figures: *Entity*, *Agent*, and *Activity*, interconnected by relationships like *wasGeneratedBy*, *used*, *wasAssociatedWith*, and *wasAttributedTo*. These relationships, defined in the Provenance Data Model (PROV-DM) [Moreau et al. 2013a], underpin the W3C PROV family of specifications. Figure 1 illustrates these core elements in a provenance graph, adhering to W3C PROV conventions. In this graphical representation, entities, activities, and agents are depicted by yellow ovals, blue rectangles, and orange "pentagon houses", respectively, connected by directed edges symbolizing their relationships.

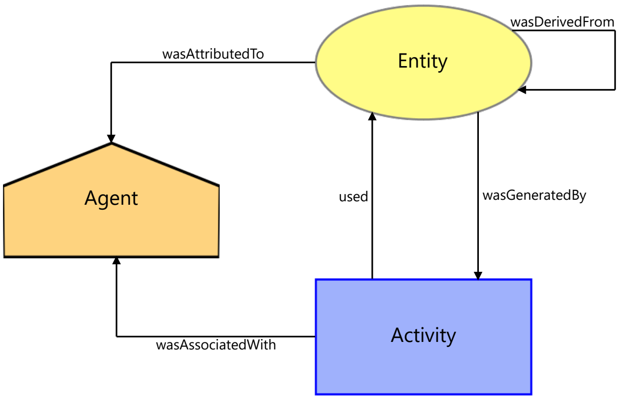


Figure 1. Main components and relationships in the W3C PROV Data Model. Adapted from: [Gil et al. 2013].

The PROV-N notation [Moreau et al. 2013b] simplifies the translation of the PROV data model into a concise format for human understanding, serving as the foundation for formal semantics. Furthermore, other human- and machine-readable representations of PROV, such as PROV-JSON and PROV-XML, are also available.

PROV-DM is a domain-agnostic conceptual data model underlying the W3C PROV Family of specifications. It accommodates diverse applications through extensibility points for domain-specific extensions. The model distinguishes between core structures, capturing essential provenance aspects, and extended structures for specialized applications. Structurally, PROV-DM comprises six key components, addressing various provenance documentation facets [Moreau et al. 2013a].

1. **Entities and Activities**: Define relationships between an *Entity* and an *Activity*, including *used* (Use), *wasGeneratedBy* (Generation), *wasStartedBy* (Start), *wasEndedBy* (End), *wasInvalidatedBy* (Invalidation), and *wasInformedBy* (Communication).
2. **Derivations**: Address the derivations of entities from other entities and subtypes of derivation, indicating that an *Entity* may have been derived (*wasDerivedFrom*) from another entity. Furthermore, this derivation may have been of a specific type, such as a revision (*wasRevisionOf*), a citation (*wasQuotedFrom*), or a primary source (*hadPrimarySource*), each of which can be related to an *Activity*.
3. **Agents, Responsibility, Influence**: Address agents and the relationships representing their responsibility and influence over an entity or activity. It includes *Entity*, *Activity*, and *Agent*, along with the relationships *wasAttributedTo* (Attribution), *wasAssociatedWith* (Association), and *actedOnBehalfOf* (Delegation).
4. **Bundles**: Refer to a mechanism for supporting the provenance of provenance.
5. **Alternatives**: Enable the representation of alternatives or specializations of entities. It includes two binary associations, which are self-relationships of *Entity*: *alternateOf* (Alternative) and *specializationOf* (Specialization).
6. **Collections**: Concern the notion of collections, where a collection is an entity that has some members. The members are entities, and therefore their provenance can be expressed. It includes *Collection*, a specialization of *Entity*, and *EmptyCollection*, a specialization of *Collection*.

2.3. Model-Driven Engineering

Model-Driven Engineering (MDE) is a software development approach centered on models as central artifacts throughout the development lifecycle [Bucchiarone et al. 2020, Schmidt 2006]. The MDE paradigm emphasizes using models to represent various aspects of a system, serving as abstractions to capture essential characteristics, support reasoning, analysis, and transformation.

MDE is centered around creating and manipulating domain-specific models, which act as both blueprints and high-level representations of systems, encapsulating complex behaviors and structures in a more accessible format. Systems are developed by specifying models, which are then transformed into executable code through automated processes. This approach relies on metamodels, which define the language for expressing models and guide the creation of domain-specific models by describing entity types in a domain and their associations, providing a vocabulary and rules for constructing models. At its core, MDE is based on metamodels, which are themselves described by a meta-metamodel at the highest abstraction level, typically represented by the Meta-Object Facility (MOF) standardized by the Object Management Group (OMG). MOF features a four-layered architecture, where each layer is an instance of the layer above it [Völter et al. 2013].

Models are constructed using Domain-Specific Modeling Languages (DSMLs), defined by a metamodel [López-Fernández et al. 2015]. DSMLs offer specialized languages for specific domains, enabling the creation of models capturing essential concepts and relationships. A metamodel is essentially “a model that defines the structure of a modeling language” [Rodrigues da Silva 2015], providing a formal description of the language’s syntax, semantics, and constraints [Bruel et al. 2018].

2.4. Eclipse Modeling Framework

The Eclipse Modeling Framework (EMF) [Steinberg et al. 2008] is a powerful framework within the Eclipse Integrated Development Environment (IDE), providing features for creating, editing, and validating models and metamodels. EMF’s key capability includes code generation, translating metamodels into corresponding Java classes, enabling the instantiation of metaclasses and the creation of models aligned with the metamodel.

EMF consists of three main components. Firstly, the core framework includes the Ecore meta-metamodel, which defines models and provides runtime support features like change notification, XML Metadata Interchange (XMI) serialization for persistence, and a reflective Application Programming Interface (API) for manipulating EMF objects. Secondly, the *EMF.Edit* framework offers generic reusable classes for building editors for EMF models, including content and label providers, property source support, and a command framework for automatic undo and redo operations in JFace[2]-based editors. Lastly, *EMF.Codegen* provides a code generation facility for creating comprehensive editors for EMF models, offering a graphical user interface for setting generation options and invoking generators, and integrating with Eclipse Java Development Tooling (JDT) for enhanced functionality.

[2] https://wiki.eclipse.org/The_Official_Eclipse_FAQs#JFace

Metamodels in EMF are constructed using the Ecore meta-meta-model, which, in turn, is based on the Meta Object Facility (MOF) meta-metamodel [Steinberg et al. 2008]. When defining metamodels with Ecore, developers utilize class instances to specify structure, associations, and properties. An Ecore-based metamodel typically includes different class instances such as `EClass`, `EAttribute`, `EReference`, `ESuperType`, and `EDataType`. This process, conducted using the Ecore language, integrates seamlessly with other Eclipse projects. For example, Eclipse Sirius [Madiot and Paganelli 2015] facilitates the creation of custom model editors, while Eclipse Acceleo [Madiot et al. 2024] supports model-to-text (M2T) transformations.

2.5. Metamodel Quality Requirements and Evaluation (MQuaRE) framework

Proposed by [Kudo 2021, Kudo et al. 2020a, Kudo et al. 2020b], the Metamodel Quality Requirements and Evaluation (MQuaRE) serves as a crucial tool for assessing metamodel quality across diverse applications. The framework ensures metamodel quality through five key characteristics: compliance, conceptual suitability, usability, maintainability, and portability. These characteristics are further subdivided into detailed sub-characteristics, offering a comprehensive guide for assessing metamodel quality.

MQuaRE delineates 19 Metamodel Quality Requirements (MQRs) and 23 associated quality measures, providing a robust foundation for metamodel evaluation. The framework recommends a structured five-phase evaluation process: (1) establish evaluation requirements, (2) specify the evaluation, (3) design the evaluation, (4) execute the evaluation, and (5) conclude the evaluation. Each phase is crucial for maintaining consistency and effectiveness throughout the evaluation process.

3. Related Work

In [Kinderen et al. 2017], the authors introduce metamodel provenance to trace the origins of metamodel elements, such as language concepts, attributes, or constraints. They focus on a goal-driven approach, illustrating how goal models help understand the origins of conceptual modeling language elements. The authors demonstrate this approach with a scenario in the electricity domain, providing a practical illustration of their methodology.

The research in [Bastin et al. 2023] focuses on evaluating data quality in citizen science. The authors emphasize the need for transparent documentation of data processes and introduce a user-friendly prototype authoring tool developed using MDE techniques. This tool provides a text editor to simplify the creation of machine-readable dataset descriptions, based on the ISO 19115-3 [ISO Central Secretary 2023] geographic information provenance model, aiming to enhance provenance information and promote data integration and sharing in the domain.

The paper [Velasco et al. 2023] addresses challenges in smart contract development, focusing on contract immutability and asset storage. It introduces the High-Level Metamodel for Smart Contract (HLM-SC), which uses MDE to declare contract elements at a high level for improved safety and reliability. The authors evaluate HLM-SC's conceptual validity using the MQuaRE framework and external evaluators. The paper also offers a practical guide for developers adopting HLM-SC and demonstrates its application in a real-world NFT industry scenario.

Compared to previous studies, Ecore4PROV-DM specifically addresses the challenge of modeling data provenance to enhance traceability and reproducibility in Information Systems. The goal-driven approach in [Kinderen et al. 2017] focuses on tracing the origins of metamodel elements but lacks a structured, easily integrable solution for Information Systems. In contrast, Ecore4PROV-DM provides a domain-independent metamodel that adheres to the W3C PROV standard. This proposed solution improves the traceability and reproducibility of data-intensive applications by documenting all interactions within the provenance chain, making them easy to trace back, as demonstrated in the use cases. Furthermore, similar to the work by [Velasco et al. 2023], our study employs the MQuaRE framework and external evaluators to rigorously assess the conceptual validity of the proposed metamodel.

The Ecore4PROV-DM metamodel distinguishes itself by providing a clear and structured representation of provenance information, enhancing traceability and reproducibility in data-intensive applications. By leveraging the well-established concepts and structures of the W3C PROV standard, Ecore4PROV-DM offers a practical and standardized approach to modeling provenance, facilitating interoperability and exchange of provenance information across different systems and domains. Additionally, the metamodel's compatibility with existing Eclipse-based MDE tools and frameworks enhances its usability and adoption, making it a valuable asset for researchers and practitioners in the field of provenance management. This compatibility with the W3C PROV standard and MDE also allows for easy integration with existing systems and tools, further enhancing its practical utility in real-world applications.

4. Ecore4PROV-DM Metamodel

Meticulously representing the W3C PROV data model (PROV-DM) using the Ecore metametamodel is key to creating a structured approach that ensures compliance with established standards. Ecore is chosen because it allows for domain-specific models that integrate easily into diverse Information Systems. Unlike other methods, Ecore provides a robust framework that supports Model-Driven Engineering (MDE) practices, facilitating tool and application development for managing provenance information. The Ecore4PROV-DM metamodel enhances traceability and reproducibility in data-intensive applications by offering a consistent and adaptable solution across various domains, addressing gaps left by models lacking this structure. The definitions of the PROV-DM components, as specified in [Moreau et al. 2013a], serve as the basis for this representation. This approach ensures the resulting metamodel closely mirrors the W3C PROV standards, providing a reliable framework for provenance information modeling in Information Systems.

In Figure 2, the structure of the Ecore4PROV-DM metamodel is illustrated, highlighting its construction in the Eclipse Modeling Framework (EMF) to represent the PROV-DM data model accurately. The semantics of the metamodel are elucidated through relationships between metaclasses, depicted by edges, and their multiplicities, indicated by numbering. The metamodel's foundational elements, denoted in blue, consist of the metaclasses: *Entity*, *Activity*, and *Agent*. These elements form the core of the PROV-DM specification, encapsulating the primary components of provenance data.

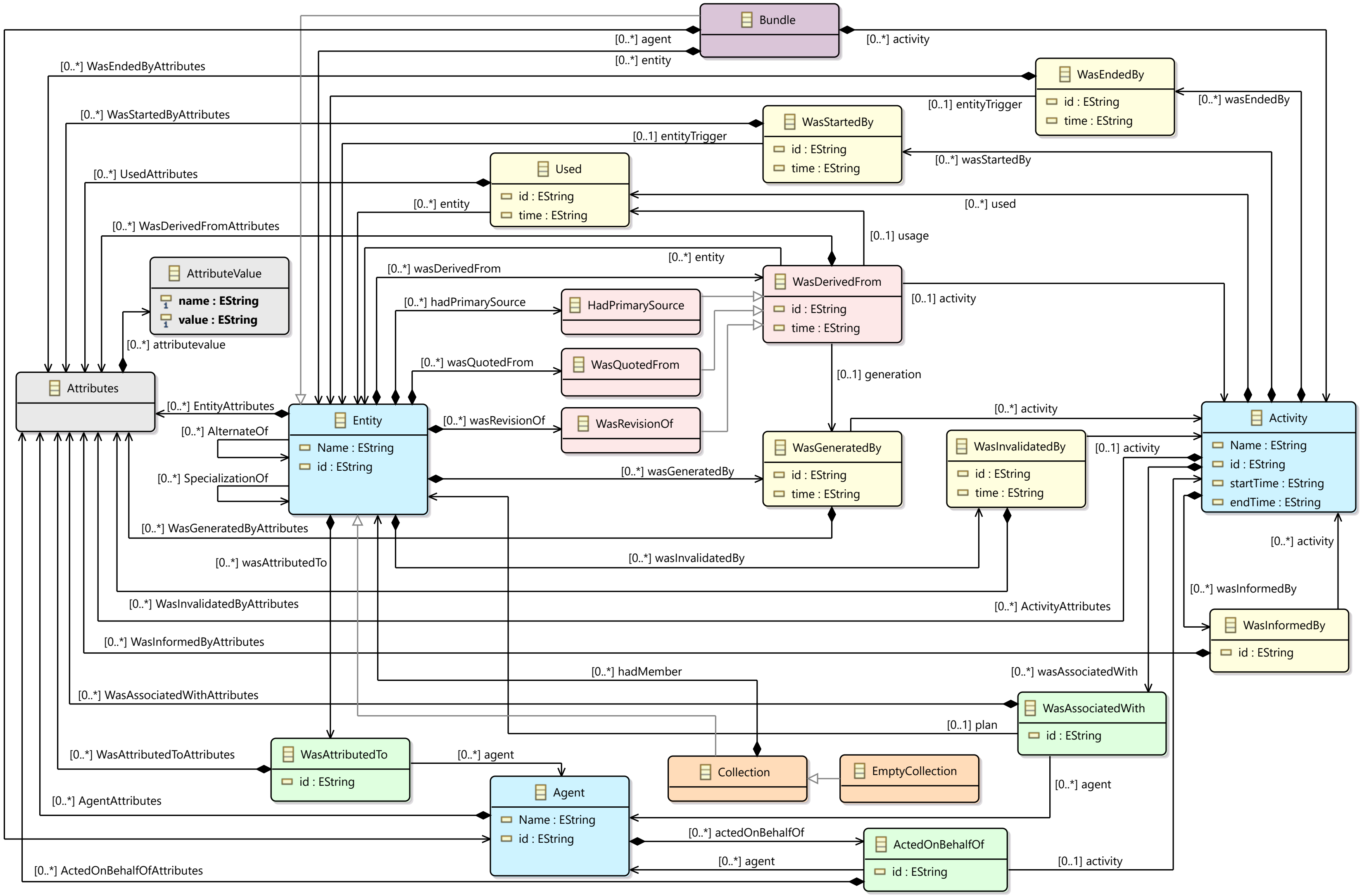


Figure 2. Ecore4PROV-DM: Ecore metamodel depicting the W3C PROV Data Model.

To implement relationships between metaclasses in EMF, Ecore4PROV-DM uses two techniques. The first method, “element-based edges,” employs a metaclass as a mediator between two related metaclasses, which is advantageous for representing relationships as actual classes, simplifying model component navigation, and enhancing clarity and usability. The second method, “relation-based edges,” directly connects metaclasses through an edge signifying their relationship, creating a more straightforward and less cluttered model by defining relationships without intermediary classes.

Additionally, the metaclasses *Attributes* and *AttributeValue*, illustrated in gray in Figure 2, instantiate attribute-value pairs that detail the properties of provenance entities, activities, and agents, adding depth and specificity to the provenance model.

To simplify the explanation of the PROV-DM, its creators categorized the elements into six components, each grouping members for specific purposes:

- Component 1 (Entities and Activities), detailed in Subsection 4.1 and represented in Ecore4PROV-DM in yellow.
- Component 2 (Derivations), detailed in Subsection 4.2 and represented in Ecore4PROV-DM in red.
- Component 3 (Agents, Responsibility, Influence), detailed in Subsection 4.3 and represented in Ecore4PROV-DM in green.
- Component 4 (Bundles), detailed in Subsection 4.4 and represented in Ecore4PROV-DM in purple.
- Component 5 (Alternatives), detailed in Subsection 4.5.
- Component 6 (Collections), detailed in Subsection 4.6 and represented in Ecore4PROV-DM in orange.

Table 1 provides a comprehensive summary of the core elements that comprise the PROV-DM data model. The primary elements, represented by the rows in bold, delineate the foundational components of the model and their potential relationships. The table also illustrates the corresponding syntax in the PROV-N language for representing these relationships. The notation used in the PROV-N language is explained as follows: `id` stands for identifier; `attr` denotes attribute; `val` signifies value; `st` represents start time; `et` indicates end time; `e` is entity; `a` refers to activity; `t` stands for timestamp; `ag` denotes agent; `g` signifies generation; `u` represents usage; `pl` is plan; `alt` indicates alternative; and `c` refers to collection. This detailed breakdown helps understand the structure and syntax of the PROV-N language, facilitating its use in modeling provenance information.

Each component of the PROV-DM is presented in the following sections, based on the data model specification available in [Moreau et al. 2013a], along with the specific details of its implementation within the Ecore4PROV-DM metamodel. It is important to note that the figures included in the subsequent subsections are excerpts from the Ecore4PROV-DM metamodel. These excerpts provide insights into the implementation of individual components, whereas the complete metamodel is illustrated in Figure 2.

Table 1. Types and relationships of the PROV-DM data model. Adapted from [Moreau et al. 2013a].

Comp.	Element	Representation in the PROV-N notation
(1)	**Entity**	**entity(id, [attr1=val1, ...])**
	Activity	**activity(id, st, et, [attr1=val1, ...])**
	Generation	wasGeneratedBy(id;e,a,t,attrs)
	Usage	used(id;a,e,t,attrs)
	Communication	wasInformedBy(id;a2,a1,attrs)
	Start	wasStartedBy(id;a2,e,a1,t,attrs)
	End	wasEndedBy(id;a2,e,a1,t,attrs)
	Invalidation	wasInvalidatedBy(id;e,a,t,attrs)
(2)	Derivation	wasDerivedFrom(id; e2, e1, a, g2, u1, attrs)
	Revision	... prov:type='prov:Revision' ...
	Quotation	... prov:type='prov:Quotation' ...
	Primary Source	... prov:type='prov:PrimarySource' ...
(3)	**Agent**	**agent(id, [attr1=val1, ...])**
	Attribution	wasAttributedTo(id;e,ag,attr)
	Association	wasAssociatedWith(id;a,ag,pl,attrs)
	Delegation	actedOnBehalfOf(id;ag2,ag1,a,attrs)
	Plan	... prov:type='prov:Plan' ...
	Person	... prov:type='prov:Person' ...
	Organization	... prov:type='prov:Organization' ...
	SoftwareAgent	... prov:type='prov:SoftwareAgent' ...
	Influence	wasInfluencedBy(id;e2,e1,attrs)
(4)	**Bundle constructor**	**bundle id description_1 ... description_n endBundle**
	Bundle type	**... prov:type='prov:Bundle' ...**
(5)	Alternate	alternateOf(alt1, alt2)
	Specialization	specializationOf(infra, supra)
(6)	**Collection**	**... prov:type='prov:Collection' ...**
	EmptyCollection	**... prov:type='prov:EmptyCollection' ...**
	Membership	hadMember(c,e)

4.1. Component 1: Entities and Activities

Depicted in Figure 3, this component delineates the relationships between an *Entity* and an *Activity*. These relationships include *used* (Use), where an activity utilizes an entity; *wasGeneratedBy* (Generation), indicating that an entity was produced by an activity; *wasStartedBy* (Start), denoting that an activity was initiated by an entity; *wasEndedBy* (End), signifying that an activity was concluded by an entity; *wasInvalidatedBy* (Invalidation), meaning that an entity was rendered invalid by an activity; and *wasInformedBy* (Communication), where an activity was informed by another activity. These relationships are crucial for representing the interactions between entities and activities within a provenance model. The metaclasses that constitute this component are visually represented in yellow within the Ecore4PROV-DM metamodel.

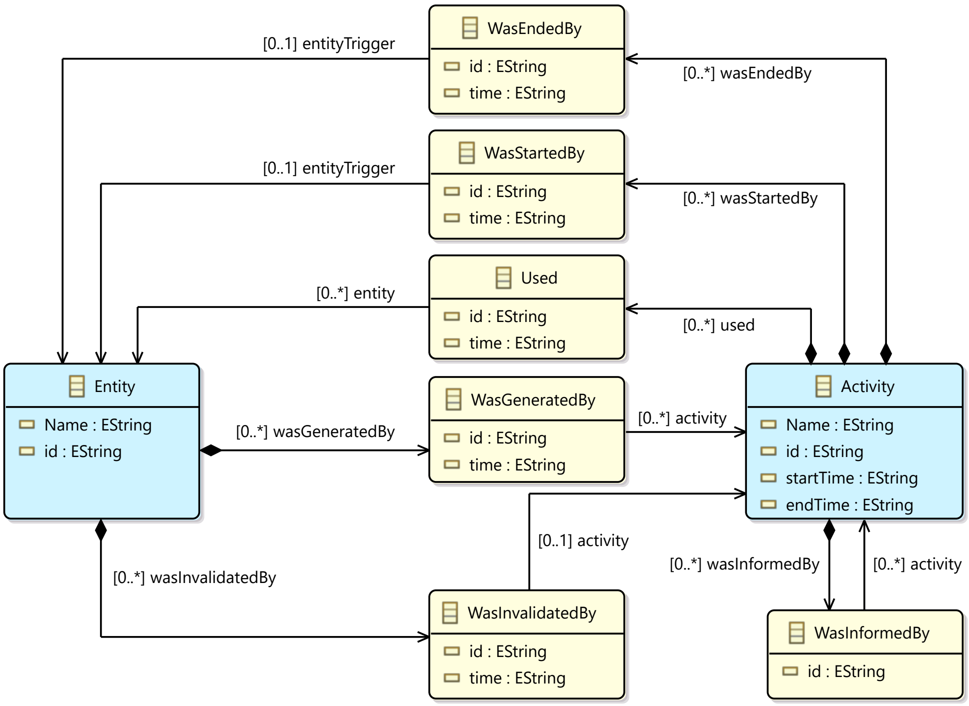


Figure 3. Extract from the Ecore4PROV-DM metamodel showing the implementation of PROV-DM Component 1.

4.1.1. Entity

An entity is defined as a physical, digital, conceptual, or other type of object that possesses certain fixed characteristics. This broad definition encompasses a wide variety of items, whether they exist in the real world or not. It is implemented by the *Entity* metaclass. This notion enables the capturing and documenting of the origins and lifecycle of objects within a provenance record.

4.1.2. Activity

An activity is a process or event that occurs over a span of time and interacts with various entities. This interaction can encompass a wide range of actions, including the consumption, processing, transformation, modification, reallocation, utilization, or generation of these entities. It is implemented by the metaclass *Activity*. This concept is pivotal for accurately documenting provenance, as it captures the dynamic interactions and transformations that entities undergo within a system.

4.1.3. Generation

Generation refers to the process by which a new entity is brought into existence through the completion of an activity. Prior to this act of generation, the entity does not exist; it is through the activity's culmination that the entity is produced and subsequently becomes available for use within the system. This concept is implemented in the provenance model by the relationship denoted as *wasGeneratedBy*. Specifically, this relationship connects an entity, represented by the metaclass *Entity*, with an activity, represented by the metaclass *Activity*. The connection is mediated by the metaclass *WasGeneratedBy*, which formalizes the association between the entity and the activity responsible for its creation. This concept is crucial for accurately documenting the origins of data, thereby enhancing the reliability of provenance records.

4.1.4. Use

The concept of “Use” refers to the initiation of an activity’s interaction with a particular entity. This marks the point at which the activity begins to utilize the entity, and prior to this moment, the activity had no interaction with – nor was it influenced by – the entity in question. This concept is implemented through the *used* relationship, which connects an activity, represented by the metaclass *Activity*, with an entity, represented by the metaclass *Entity*. This connection is mediated by the metaclass *Used*. The relevance of this concept lies in its ability to precisely capture the dependencies and interactions between activities and entities, which is crucial for accurate provenance recording.

4.1.5. Communication

Communication involves the exchange of an unspecified entity between two activities, where one activity utilizes an entity generated by the other. This exchange indicates a dependency of activity a_2 on activity a_1, facilitated by an entity produced by a_1 and subsequently used by a_2. This is represented by the *wasInformedBy* relationship between the two activities, instantiated by the metaclass *Activity* and mediated by the metaclass *WasInformedBy*. This concept is crucial in provenance representation as it captures the flow of information and dependencies between activities, ensuring accurate documentation of how data and processes are interconnected.

4.1.6. Start

The concept of “Start” refers to the point in time when an activity is initiated by an entity, which is known as a *trigger*. Prior to this start event, the activity does not exist. Following the start of the activity, any subsequent use, generation, or invalidation involving the activity takes place. The start can be attributed to a triggering entity that directly initiates the activity or to another activity, termed as a *starter*, that produces the trigger. This relationship is represented by the *wasStartedBy* linkage between an activity (represented by the metaclass *Activity*) and an entity (represented by the metaclass *Entity*), and is facilitated through the metaclass *WasStartedBy*. This concept is crucial in provenance representation as it allows to precisely document the initiation of activities, thereby ensuring accurate tracking and accountability of processes.

4.1.7. End

End refers to the point at which an activity is considered to have concluded, triggered by an entity known as the *trigger*. At this moment, the activity ceases to exist. It is crucial to note that any usage, generation, or invalidation related to an activity must occur before the activity's termination. The "End" can reference either the triggering entity that caused the activity to end or an activity, referred to as the *ender*, that generated the trigger. This relationship is implemented through the *wasEndedBy* association, which connects an activity (represented by the metaclass *Activity*) and an entity (represented by the metaclass *Entity*), mediated by the metaclass *WasEndedBy*. This concept is fundamental in provenance representation as it precisely delineates the lifecycle of activities, ensuring accurate documentation of when and why activities conclude within the provenance data.

4.1.8. Invalidation

Invalidation marks the commencement of the destruction, cessation, or expiration of an existing entity by an activity. After invalidation, the entity is no longer available for use or subject to further invalidation. It is important to note that any generation or use of an entity must precede its invalidation. This concept is implemented by the relationship *wasInvalidatedBy* which exists between an entity (represented by the metaclass *Entity*) and an activity (represented by the metaclass *Activity*). This relationship is mediated by the metaclass *WasInvalidatedBy*, which serves as a connector defining the interaction where the activity invalidates the entity.

4.2. Component 2: Derivations

Depicted in Figure 4, this component addresses the derivations of entities from other entities and the subtypes of these derivations. It indicates that an instance of the *Entity* metaclass may have been derived (*wasDerivedFrom*) from another entity. This derivation can be of a specific type, such as a revision (*wasRevisionOf*), a citation (*wasQuotedFrom*), or a primary source (*hadPrimarySource*). Each of these derivation types can be associated with an *Activity*. These relationships are crucial for representing the derivation processes and the lineage of entities within a provenance model. In the Ecore4PROV-DM metamodel, the metaclasses that implement these relationships are highlighted in red.

4.2.1. Derivation

A derivation refers to the transformation of one entity into another, the update of an entity that results in a new version, or the construction of a new entity based on a pre-existing one. This process is implemented through the relationship *wasDerivedFrom*, which connects two entities (instances of the metaclass *Entity*). The relationship is mediated by the metaclass *WasDerivedFrom*, which can potentially have some form of association with an activity (represented by the metaclass *Activity*). This relationship ensures that the provenance of derived entities is accurately tracked and documented, providing a comprehensive view of how new entities are generated from existing ones.

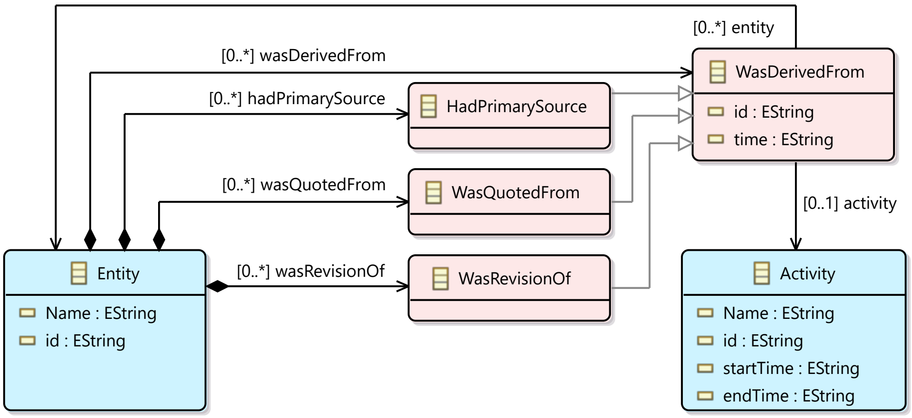


Figure 4. Extract from the Ecore4PROV-DM metamodel showing the implementation of PROV-DM Component 2.

4.2.2. Revision

A revision represents a derivation in which the resulting entity is a revised version of an original entity. This concept is implemented through the relationship *wasRevisionOf* between two entities, represented by the metaclass *Entity*. The relationship itself is mediated by the metaclass *WasRevisionOf*, which is a specialization of the metaclass *WasDerivedFrom*. Additionally, this relationship may involve some form of interaction with an activity, as represented by the metaclass *Activity*. This structure ensures that revisions are accurately captured within the provenance model, reflecting the detailed and hierarchical nature of the provenance data.

4.2.3. Quotation

A citation involves the repetition of (part or all of) an entity, which can be a text, image, or other forms of content, by an individual who may or may not be the original author. This concept is implemented through the relationship *wasQuotedFrom* between two entities, represented by the metaclass *Entity*. This relationship is mediated by the metaclass *WasQuotedFrom*, which is itself a specialization of the metaclass *WasDerivedFrom*. The *WasDerivedFrom* metaclass signifies that one entity is derived from another, and it can have some form of association with an activity, represented by the metaclass *Activity*. This hierarchical and relational structure allows for a detailed and nuanced representation of how citations and derivations are interconnected within the provenance model.

4.2.4. Primary Source

A primary source for a given topic is defined as a document, artifact, or any piece of information created by an individual or entity that possesses direct experience and knowledge of the subject matter at the time it was studied. This concept is implemented through the relationship *hadPrimarySource* which links two entities (instances of the metaclass

Entity). This relationship is mediated by the metaclass *HadPrimarySource*, which is a specialization of the more general metaclass *WasDerivedFrom*. The *WasDerivedFrom* metaclass itself may have an associated relationship with an activity (instance of the metaclass *Activity*), thereby providing a comprehensive framework for tracing the origins and derivations of primary sources within the provenance model.

4.3. Component 3: Agents, Responsibility, Influence

Illustrated in Figure 5, this component focuses on agents and the relationships that represent their responsibility and influence over entities or activities. It encompasses the metaclasses *Entity*, *Activity*, and *Agent*, along with the relationships *wasAttributedTo* (Attribution), *wasAssociatedWith* (Association), and *actedOnBehalfOf* (Delegation). These relationships are essential for modeling the interactions and dependencies between agents and other components in the provenance model. In Ecore4PROV-DM metamodel, the metaclasses and their corresponding relationships are visually distinguished in green.

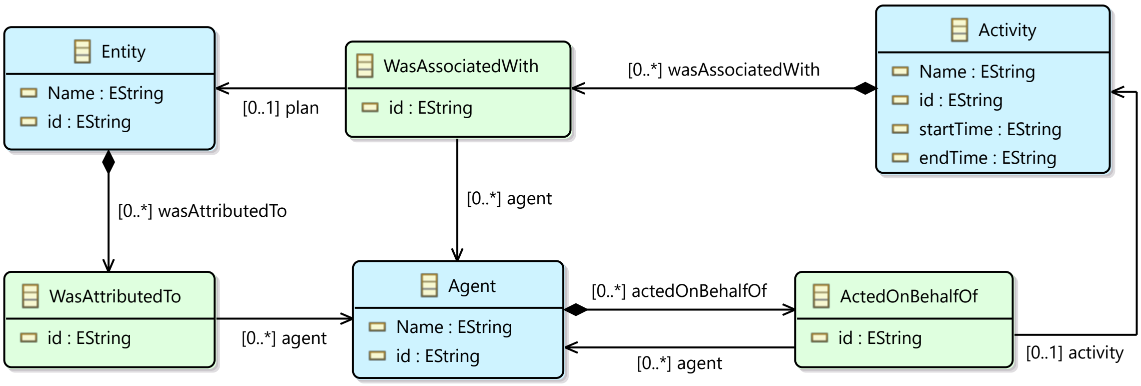


Figure 5. Extract from the Ecore4PROV-DM metamodel showing the implementation of PROV-DM Component 3.

4.3.1. Agent

An agent refers to something that holds some level of responsibility for an activity that is occurring, for the existence of a particular entity, or for the actions carried out by another agent. This level of accountability and influence over activities and entities is encapsulated in the concept of the *Agent* metaclass, which serves to implement and formalize this notion within the metamodel.

4.3.2. Attribution

Attribution refers to the process of associating an entity with an agent, establishing a clear connection between the two. This is realized through a specific relationship mediated by the metaclass *WasAttributedTo*, where an entity, represented by the metaclass *Entity*, is linked to an agent, represented by the metaclass *Agent*. This relationship ensures that the provenance of the entity is traceable to the responsible agent, providing transparency and accountability.

4.3.3. Association

An activity association involves assigning responsibility to an agent for a particular activity, thereby indicating that the agent played a role in the execution of that activity. This concept also permits the specification of a plan. A plan is defined as an entity that encapsulates a series of actions or steps devised by one or more agents with the intention of achieving specific goals. This concept is implemented through the *wasAssociatedWith* relationship, which connects an activity (represented by the *Activity* metaclass) with an agent (represented by the *Agent* metaclass). This relationship is mediated by the *WasAssociatedWith* metaclass. Furthermore, the plan can be articulated through the *plan* relationship, which links the *WasAssociatedWith* metaclass to an entity, represented by the *Entity* metaclass. This structure allows for a detailed representation of the involvement of agents in activities, along with the plans they implement to achieve their objectives.

4.3.4. Delegation

Delegation involves the assignment of authority and responsibility to an agent, either by itself or by another agent, to perform a specific activity as a delegate or representative. Despite this delegation, the agent on whose behalf the delegate acts retains some level of responsibility for the outcome of the delegated work. This concept is implemented through the relationship *actedOnBehalfOf* between two agents, represented by the metaclass *Agent* and mediated by the metaclass *ActedOnBehalfOf*. The specific activity for which the delegation is valid can be represented by the *activity* relationship, which links the *ActedOnBehalfOf* metaclass to an activity, represented by the *Activity* metaclass.

4.4. Component 4: Bundles

Illustrated in Figure 6, this component refers to a mechanism designed to support the provenance of provenance. This component includes the *Bundle* metaclass, which is depicted in purple within the Ecore4PROV-DM model. The *Bundle* metaclass is defined as a specialization of the *Entity* metaclass, providing a structured way to encapsulate and manage the provenance information related to other provenance records.

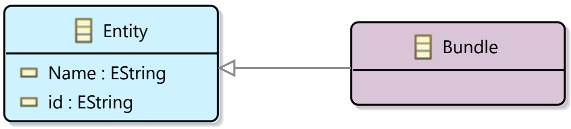


Figure 6. Extract from the Ecore4PROV-DM metamodel showing the implementation of PROV-DM Component 4.

4.4.1. Bundle

A bundle is a named set of provenance descriptions and is itself considered an entity, thereby allowing the provenance of provenance to be expressed. In the Ecore4PROV-DM metamodel, this is implemented by the metaclass *Bundle*, which is a specialization of the entity metaclass *Entity*. The *Bundle* class is the central component of the Ecore4PROV-DM metamodel. Consequently, in any instance of the metamodel, there will be at least one bundle, which can contain agents (metaclass *Agent*), entities (metaclass *Entity*), activities (metaclass *Activity*), and all the relationships between these elements. This structure ensures that the metamodel can comprehensively represent complex provenance scenarios, including the relationships and interactions among various provenance elements.

4.5. Component 5: Alternatives

Depicted in Figure 7, this component facilitates the expression of alternatives or specializations of entities. It includes the *Entity* metaclass and features two binary associations, both of which are self-relationships of the *Entity* metaclass itself. These associations are *alternateOf* (Alternative) and *specializationOf* (Specialization), which allow for the representation of alternative instances of an entity and the specialization of an entity into more specific forms, respectively. This component is crucial as it provides the flexibility needed to model complex relationships and variations within the provenance data, enhancing the ability to capture nuanced differences and hierarchies among entities.

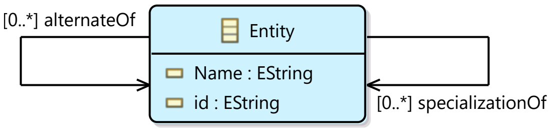


Figure 7. Extract from the Ecore4PROV-DM metamodel showing the implementation of PROV-DM Component 5.

4.5.1. Alternative

Two alternative entities represent different aspects of the same underlying thing. These aspects may be identical or distinct, and the alternative entities might have overlapping or non-overlapping timeframes. This concept is implemented by the relationship *alternateOf*, which connects two entities (instances of the *Entity* metaclass). This relationship allows for the expression of the notion that the entities, while different, are considered alternatives to one another, providing a flexible way to model varying perspectives or states of the same object within a provenance record.

4.5.2. Specialization

An entity that is a specialization of another inherits all the characteristics and attributes of the original entity and, furthermore, introduces additional specific attributes or properties that provide a more detailed representation of the same concept or thing. This relationship implemented through the *specializationOf* relationship, which connects two entities (metaclass *Entity*). This connection indicates that the specialized entity is an extension or more detailed version of the original entity, encompassing all its properties while also incorporating more specific aspects.

4.6. Component 6: Collections

Depicted in Figure 8, this component addresses the concept of collections. A collection is defined as an entity comprising multiple members, each of which is also an entity whose provenance can be documented. This component includes the metaclass *Collection*, which extends the *Entity* metaclass, allowing for the grouping of entities while maintaining their individual provenance information. Additionally, it encompasses the *EmptyCollection* metaclass, a specialization of the *Collection* metaclass, representing collections that currently have no members. In the Ecore4PROV-DM metamodel, these metaclasses are depicted in orange. This component is crucial as it enables the modeling of complex structures within provenance data, facilitating comprehensive and accurate provenance tracking for collections of related entities.

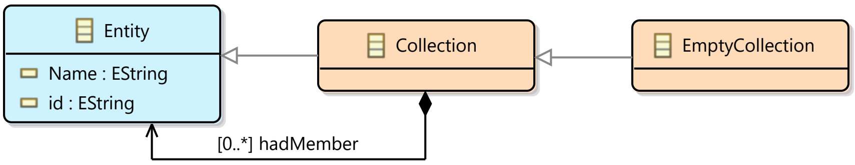


Figure 8. Extract from the Ecore4PROV-DM metamodel showing the implementation of PROV-DM Component 6.

4.6.1. Collection

A collection is an entity that provides an organizational structure for a group of elements, each of which must also be entities. These elements are considered members of the collection. The concept of a collection is implemented by the metaclass *Collection*, which is a specialization of the more general metaclass *Entity*. This specialization enables the *Collection* to be treated as an entity itself, thereby allowing its provenance to be expressed. Moreover, this approach offers a refined and structured method for managing groups of related entities.

4.6.2. Empty Collection

An empty collection refers to a collection that contains no members. It is implemented by the *EmptyCollection* metaclass, which is a specialized form of the *Collection* metaclass. By specializing the *Collection* metaclass, the *EmptyCollection* metaclass inherits its properties and behaviors, while explicitly defining a collection that is devoid of any elements. This allows for a clear and structured representation of collections that are intentionally empty within the provenance document.

4.6.3. Membership

Indicates the belonging of an entity within a collection. This relationship is implemented through the *hadMember* relationship, which connects a collection, represented by the metaclass *Collection*, to an entity, represented by the metaclass *Entity*.

5. Modeling Provenance: Ecore4PROV-DM Use Cases

In this section, we demonstrate the practical application of the Ecore4PROV-DM metamodel by modeling two distinct scenarios, each providing a tangible example of how Ecore4PROV-DM can be used to model real-world provenance data.

5.1. Urban Observatory Project

Here, we illustrate the application of the Ecore4PROV-DM metamodel to model a real-world provenance dataset from the ProvStore platform[3], a cloud-based repository for provenance data. We focus on a specific provenance model associated with the Urban Observatory project at Newcastle University in the United Kingdom. This project monitors urban indicators like environmental conditions and infrastructure performance using an extensive network of active sensors and CCTV cameras installed throughout the city.

The provenance model in question comprises four distinct bundles: *vargen:templ _trainset*, *vargen: templ_model*, *vargen:templ_count*, and *vargen: templ_image*. Each of these bundles represents a specific activity within the Urban Observatory project, and they utilize various entities to generate new data and insights. Every entity within the model has a set of associated attribute-value pairs, which serve to describe the properties of that entity once the provenance model is fully instantiated as a provenance record.

Figure 9 provides an the visual representation of the provenance graph for the *vargen:templ _trainset* bundle within this model, as it is hosted on the ProvStore platform. Furthermore, Figure 10 (a) presents a segment of the actual instance of this provenance model, as displayed within the Ecore4PROV-DM tree editor. Figure 10 (b) presents a code snippet of the XML file generated by EMF, which serves to represent this provenance model using the Ecore4PROV-DM metamodel.

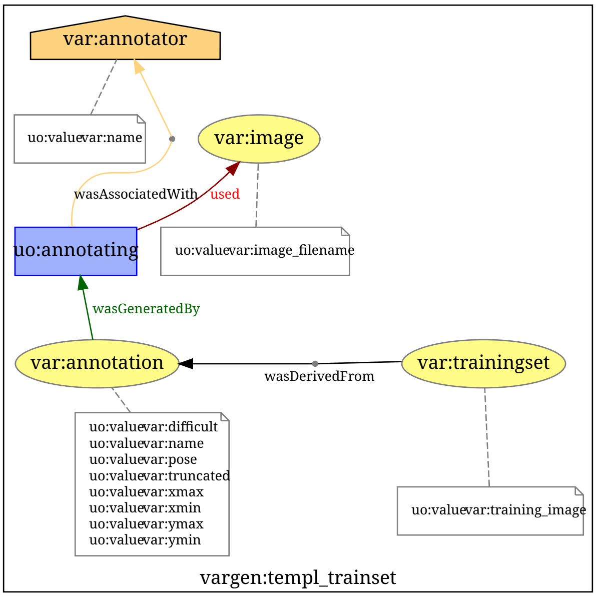
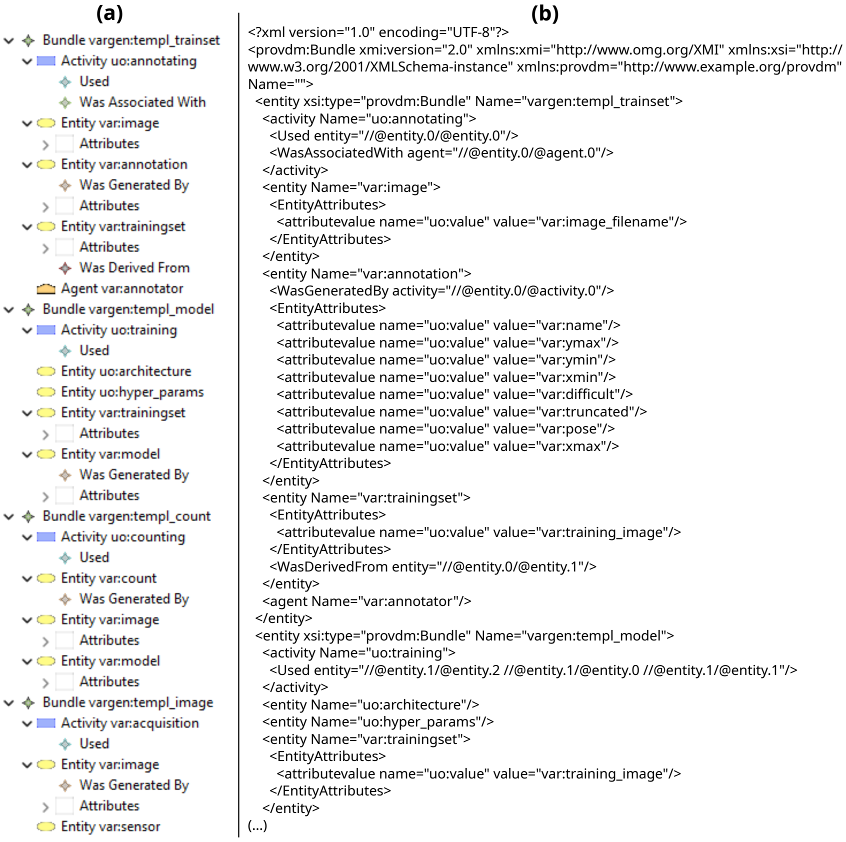


Figure 9. Excerpt of the provenance model graph from the Urban Observatory.

[3] https://openprovenance.org/store/documents/497



(a)

```
Bundle vargen:templ_trainset
  Activity uo:annotating
    Used
    Was Associated With
  Entity var:image
    Attributes
  Entity var:annotation
    Was Generated By
    Attributes
  Entity var:trainingset
    Attributes
    Was Derived From
  Agent var:annotator
Bundle vargen:templ_model
  Activity uo:training
    Used
  Entity uo:architecture
  Entity uo:hyper_params
  Entity var:trainingset
    Attributes
  Entity var:model
    Was Generated By
    Attributes
Bundle vargen:templ_count
  Activity uo:counting
    Used
  Entity var:count
    Was Generated By
  Entity var:image
    Attributes
  Entity var:model
    Attributes
Bundle vargen:templ_image
  Activity var:acquisition
    Used
  Entity var:image
    Was Generated By
    Attributes
  Entity var:sensor
```

(b)

```
<?xml version="1.0" encoding="UTF-8"?>
<provdm:Bundle xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance" xmlns:provdm="http://www.example.org/provdm"
Name="">
 <entity xsi:type="provdm:Bundle" Name="vargen:templ_trainset">
  <activity Name="uo:annotating">
   <Used entity="//@entity.0/@entity.0"/>
   <WasAssociatedWith agent="//@entity.0/@agent.0"/>
  </activity>
  <entity Name="var:image">
   <EntityAttributes>
    <attributevalue name="uo:value" value="var:image_filename"/>
   </EntityAttributes>
  </entity>
  <entity Name="var:annotation">
   <WasGeneratedBy activity="//@entity.0/@activity.0"/>
   <EntityAttributes>
    <attributevalue name="uo:value" value="var:name"/>
    <attributevalue name="uo:value" value="var:ymax"/>
    <attributevalue name="uo:value" value="var:ymin"/>
    <attributevalue name="uo:value" value="var:xmin"/>
    <attributevalue name="uo:value" value="var:difficult"/>
    <attributevalue name="uo:value" value="var:truncated"/>
    <attributevalue name="uo:value" value="var:pose"/>
    <attributevalue name="uo:value" value="var:xmax"/>
   </EntityAttributes>
  </entity>
  <entity Name="var:trainingset">
   <EntityAttributes>
    <attributevalue name="uo:value" value="var:training_image"/>
   </EntityAttributes>
   <WasDerivedFrom entity="//@entity.0/@entity.1"/>
  </entity>
  <agent Name="var:annotator"/>
 </entity>
 <entity xsi:type="provdm:Bundle" Name="vargen:templ_model">
  <activity Name="uo:training">
   <Used entity="//@entity.1/@entity.2 //@entity.1/@entity.0 //@entity.1/@entity.1"/>
  </activity>
  <entity Name="uo:architecture"/>
  <entity Name="uo:hyper_params"/>
  <entity Name="var:trainingset">
   <EntityAttributes>
    <attributevalue name="uo:value" value="var:training_image"/>
   </EntityAttributes>
  </entity>
(...)
```

Figure 10. The Urban Observatory provenance model as an instance of the Ecore4PROV-DM metamodel.

5.2. Provenance Templates

A provenance template, as defined in PROV-Template [Moreau et al. 2018], is a PROV document that outlines the desired provenance to be generated. This template uses variables as placeholders for values, serving as a declarative specification of the intended provenance to be produced by an application. Bindings in the template associate variables with corresponding values. The PROV-Template expansion algorithm takes a template and bindings as input, generating a provenance document with variables substituted by their corresponding values.

The initial step in this methodology involves the creation of a "provenance model" that outlines the structure of the intended provenance. Traditionally, this process entails drafting the model manually and then translating it into PROV-N notation, which is prone to errors. However, utilizing Ecore4PROV-DM reduces the likelihood of errors, as the resulting provenance models adhere to the W3C PROV data model, ensuring compliance with its definitions and constraints. This approach enables interactive construction of the provenance model, streamlining the overall process. Once the model is finalized, the generation of its PROV-N code can be achieved through model-to-text (M2T) transformations. Furthermore, by leveraging Ecore4PROV-DM, developers can efficiently manage the relationships and attributes associated with their data provenance, enhancing traceability and reproducibility in their applications. This capability ultimately supports the stated goal of facilitating effective provenance integration within data-intensive applications.

Figure 11 illustrates a provenance model from [Moreau 2017], showing an arithmetic operation like the sum of two values resulting in a third value. Here, the activity *var:operation* represents the arithmetic operation, consuming two input values denoted by the entities *var:consumed1* and *var:consumed2*, and producing a result represented by the entity *var:produced*, triggered by an *var:agent*. Furthermore, Figure 12 (a) presents a segment of the actual instance of this provenance model, as displayed within the Ecore4PROV-DM tree editor. Figure 12 (b) presents a code snippet of the XML file generated by EMF, which serves to represent this provenance model using the Ecore4PROV-DM metamodel.

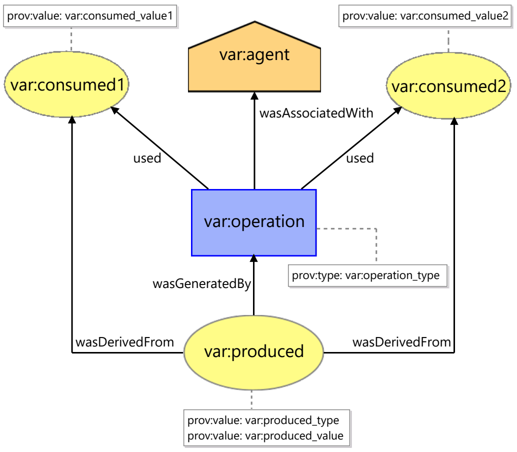
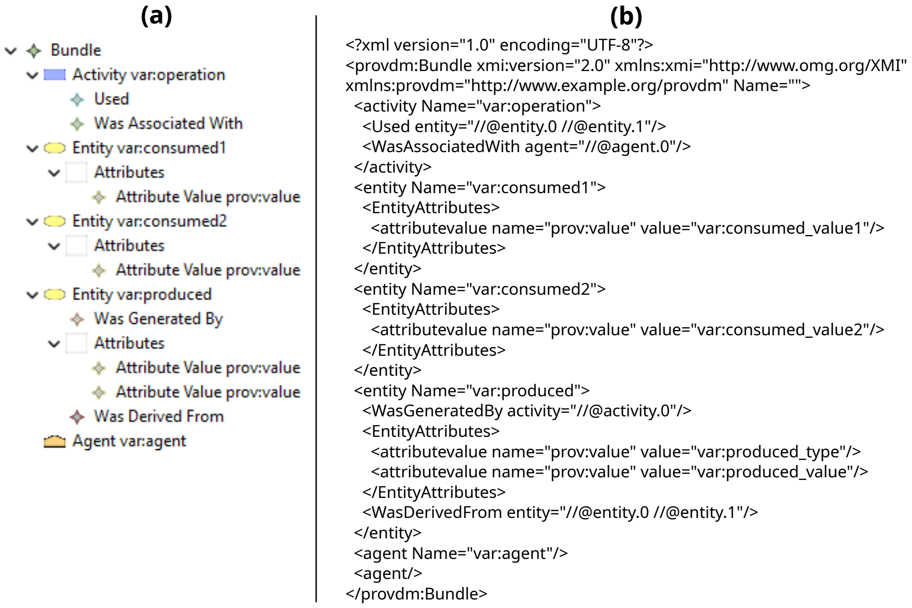

prov:value: var:consumed_value1
prov:value: var:consumed_value2
var:agent
var:consumed1
var:consumed2
wasAssociatedWith
used
used
var:operation
prov:type: var:operation_type
wasGeneratedBy
wasDerivedFrom
var:produced
wasDerivedFrom
prov:value: var:produced_type
prov:value: var:produced_value

Figure 11. Provenance graph depicting the PROV-Template for an arithmetic operation. Adapted from: [Moreau 2017].


(a)

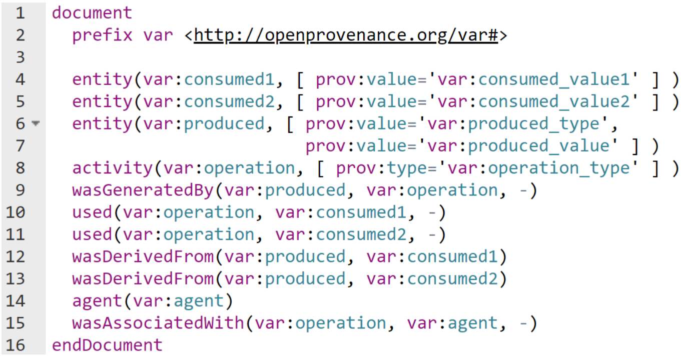
- Bundle
  - Activity var:operation
    - Used
    - Was Associated With
  - Entity var:consumed1
    - Attributes
      - Attribute Value prov:value
  - Entity var:consumed2
    - Attributes
      - Attribute Value prov:value
  - Entity var:produced
    - Was Generated By
    - Attributes
      - Attribute Value prov:value
      - Attribute Value prov:value
    - Was Derived From
  - Agent var:agent

(b)

```
<?xml version="1.0" encoding="UTF-8"?>
<provdm:Bundle xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:provdm="http://www.example.org/provdm" Name="">
 <activity Name="var:operation">
  <Used entity="//@entity.0 //@entity.1"/>
  <WasAssociatedWith agent="//@agent.0"/>
 </activity>
 <entity Name="var:consumed1">
  <EntityAttributes>
   <attributevalue name="prov:value" value="var:consumed_value1"/>
  </EntityAttributes>
 </entity>
 <entity Name="var:consumed2">
  <EntityAttributes>
   <attributevalue name="prov:value" value="var:consumed_value2"/>
  </EntityAttributes>
 </entity>
 <entity Name="var:produced">
  <WasGeneratedBy activity="//@activity.0"/>
  <EntityAttributes>
   <attributevalue name="prov:value" value="var:produced_type"/>
   <attributevalue name="prov:value" value="var:produced_value"/>
  </EntityAttributes>
  <WasDerivedFrom entity="//@entity.0 //@entity.1"/>
 </entity>
 <agent Name="var:agent"/>
 <agent/>
</provdm:Bundle>
```

Figure 12. The arithmetic operation PROV-Template as an instance of the Ecore4PROV-DM metamodel.

Figure 13 illustrates the automatically generated PROV-N code for the provenance model. This code generation is achieved through M2T transformations implemented using Eclipse Acceleo [Madiot et al. 2024]. These transformations convert the elements within the model into PROV-N notation, adhering to the definitions specified in the PROV-DM standard, as outlined in Table 1.



```
document
  prefix var <http://openprovenance.org/var#>

  entity(var:consumed1, [ prov:value='var:consumed_value1' ] )
  entity(var:consumed2, [ prov:value='var:consumed_value2' ] )
  entity(var:produced, [ prov:value='var:produced_type',
                         prov:value='var:produced_value' ] )
  activity(var:operation, [ prov:type='var:operation_type' ] )
  wasGeneratedBy(var:produced, var:operation, -)
  used(var:operation, var:consumed1, -)
  used(var:operation, var:consumed2, -)
  wasDerivedFrom(var:produced, var:consumed1)
  wasDerivedFrom(var:produced, var:consumed2)
  agent(var:agent)
  wasAssociatedWith(var:operation, var:agent, -)
endDocument
```

Figure 13. PROV-N code of the PROV-Template presented in Figures 11 and 12.

6. Ecore4PROV-DM Quality Evaluation

A team of metamodeling experts conducted an evaluation to assess the completeness, accuracy, and usability of Ecore4PROV-DM in relation to the PROV-DM data model. The specialists meticulously examined each PROV-DM component element to verify its accurate representation in Ecore4PROV-DM. This evaluation process adhered to the Metamodel Quality Requirements and Evaluation (MQuaRE) framework, as introduced by [Kudo 2021, Kudo et al. 2020a, Kudo et al. 2020b]. MQuaRE delineates 19 Metamodel Quality Requirements (MQRs) and associates them with 23 quality measures corresponding to the mentioned characteristics, providing a comprehensive set of criteria for consideration during metamodel evaluation.

In addition to confirming Ecore4PROV-DM's alignment with the W3C PROV data model, this evaluation emphasizes the metamodel's contribution to addressing challenges related to data provenance in Information Systems. By employing a structured framework, Ecore4PROV-DM enhances the capabilities of developers in managing provenance information, which in turn supports improved traceability and reproducibility in data-intensive applications.

The evaluation method employed in this study is a customized version of the MQuaRE framework. Subsection 6.1 outlines the evaluation planning following the MQuaRE usage guide. Subsection 6.2 elaborates on the execution process of the evaluation, while Subsection 6.3 provides the evaluation results for each evaluator, along with their profiles and discussion of the evaluation purpose. Lastly, Subsection 6.4 addresses potential threats to this evaluation process validity.

6.1. Planning

The evaluation of Ecore4PROV-DM aimed to gauge its alignment with the W3C PROV data model (PROV-DM). Within this context, the evaluation questionnaire focused on assessing three Metamodel Quality Requirements (MQRs) derived from the MQuaRE framework. Specifically, it targeted MQR02, evaluating the completeness of the metamodel by assessing its coverage of concepts specified in the PROV-DM. Additionally, it addressed MQR03, ensuring the correctness of the metamodel in accurately representing PROV-DM concepts. Finally, MQR07, a usability requirement, gauged the proportion of the metamodel's concepts that are easily identifiable and evident to users. The selection of these MQRs was guided by their close alignment with the overarching research objectives, which revolve around assessing the accuracy, comprehensiveness, and user-friendliness of the metamodel concerning the correct representation of the twenty-two PROV-DM concepts distributed across its six components, as meticulously elucidated in Section 4. By focusing on these specific MQRs, the evaluation process was designed to effectively measure the metamodel's fidelity to the PROV-DM standards, ensuring that it not only captures the breadth of specified concepts but also maintains correctness and accessibility in a manner conducive to users' understanding and utilization. Table 2 outlines the chosen Metamodel Quality Requirements (MQRs), illustrating the relationships between MQRs, selected characteristics and subcharacteristics, and corresponding measures.

Table 2. A detailed breakdown of the selected Metamodel Quality Requirements (MQRs). Adapted from [Kudo et al. 2020b].

Quality Requirements	Characteristics	Subcharacteristics	Measures
MQR02 - The metamodel must cover the concepts found in its specifications.	Conceptual Suitability	Conceptual Completeness	Conceptual Coverage
MQR03 - The metamodel must represent the concepts found in its specifications correctly.	Conceptual Suitability	Conceptual Correctness	Conceptual correcteness
MQR07 - The users must be able to recognize whether a metamodel is appropriate for their needs accordingly the evident concepts to the user in the metamodel specifications.	Usability	Appropriateness recognizability	Evident concepts

To facilitate the evaluation of the selected Metamodel Quality Requirements (MQRs), the MQuaRE framework recommends the use of specific artifacts. For this evaluation's scope, two foundational documents were employed to establish a context for assessing the metamodel domain. The first document introduced fundamental data provenance concepts and briefly outlined PROV-DM, while the second provided comprehensive insights into the implementation of Ecore4PROV-DM within the Eclipse Modeling Framework (EMF). In addition to these artifacts, external evaluators received a high-resolution image of the metamodel's class diagram.

Quality measures were selected based on the previously identified Metamodel Quality Requirements (MQRs). The MQuaRE framework [Kudo et al. 2020b] provides detailed information on the quality measures, including their descriptions, measurement functions, and interpretation of measured values. As further detailed in Table 3, for the three selected quality requirements, the measurement function is given by the formula $X = 1 - A/B$, where A is the number of concepts not modeled, incorrectly modeled or evident to the user, respectively for MQR02, MQR03 and MQR07; and B is the number of concepts described in the metamodel. The target value for these measures (*i.e.*, the value of X in the previous formula) is set at 1, indicating full satisfaction of the corresponding quality measure. In our evaluation, an acceptable tolerance value of 0.8 were established, allowing for some imperfection while still deeming the artifact acceptable in terms of quality. Therefore, the decision criteria for the evaluation, along with the formulas used to calculate scores for characteristics and subcharacteristics, were defined. In each case, the measurement functions recommended by the authors of the MQuaRE framework were employed.

Table 3. Measurement function for the selected MQRs. Adapted from [Kudo et al. 2020b].

MQR	Measures	Description	Measurement function
MQR02	CCp-1 - Conceptual coverage	What proportion of the specified concepts has been modeled?	X = 1 - A / B A = Number of missing concepts. B = Number of concepts described in the metamodel specifications. 0 <= X <= 1. The closer to 1, the more complete.
MQR03	CCr-1 - Conceptual correctness	What proportion of metamodel concepts are modeled correctly?	X = 1 - A / B A = Number of incorrectly modeled concepts. B = Number of concepts considered in the evaluation. 0 <= X <= 1. The closer to 1, the more correct.
MQR07	UAp-3 - Evident concepts	What proportion of metamodel concepts are evident to the user?	X = 1 - A / B A = Number of concepts evident to the user. B = Number of concepts described in the metamodel specification. 0 <= X <= 1. The closer to 1, the better.

6.2. Execution

Following the planning phase of the evaluation, the process of selecting external evaluators was conducted according to specific criteria. It was imperative that the chosen evaluators possessed expertise in software modeling or metamodeling. This decision was driven by the metamodel's current state, which demands technical proficiency for effective utilization. Consequently, the strategic selection of evaluators with these skills aligns with the research objective, which focuses on assessing the metamodel's adherence to the W3C PROV data model.

After reviewing the two artifacts and examining the Ecore4PROV-DM class diagram, evaluators participated in a Google Forms questionnaire, which covered personal details, educational background, prior knowledge of data provenance, and metamodeling, in addition to inquiries about the provided support documents. The questionnaire also included specific questions related to Ecore4PROV-DM, designed to assess the three selected Metamodel Quality Requirements (MQRs) for each of the twenty-two PROV-DM concepts within Ecore4PROV-DM: metamodel completeness, accuracy concerning alignment with the PROV-DM, and precision in representing its conceptual evidence.

6.3. Results and Discussion

Fifteen experts, all possessing degrees in Computer Science, participated in the evaluation process. Among them, only one evaluator held an undergraduate degree, while the others had completed master's degrees, PhDs, or were pursuing a PhD. The experts' responses to the evaluation questionnaire (in Portuguese) are accessible online[4]. Regarding their prior knowledge, 73.3% were unfamiliar with data provenance, 20% lacked knowledge of Model-Driven Engineering (MDE), and 26.7% were not acquainted with the Eclipse Modeling Framework (EMF).

In terms of the clarity of the provided artifacts, 93.4% of respondents found the document on data provenance and PROV-DM to be easily comprehensible, reflecting an average Likert scale rating of 4.2. Additionally, 80% of participants reported that the document elucidating the implementation of Ecore4PROV-DM in EMF was easy to understand, registering an average Likert scale rating of 4.1. The remaining evaluators affirmed that both documents were sufficiently clear, and none indicated encountering any difficulty in comprehension.

The evaluation results revealed unanimous agreement among participating experts regarding the seamless alignment of Ecore4-PROV-DM with the PROV-DM specification (MQR02 and MQR03). However, one specialist expressed reservations about the modeling of the "End" concept (*wasEndedBy*) within Component 1, while another specialist raised concerns about the implementation of Component 4 (*Bundles*).

Figure 14 depicts the score distribution assigned by evaluators for quality requirements MQR02 and MQR03. Notably, only one evaluator (E2) recorded a score below the tolerance threshold of 0.8. However, the mean score across all evaluators surpassed the acceptable tolerance (0.97) and neared the maximum score of 1.0 for both MQRs. This suggests that the experts involved in the evaluation process consider that Ecore4PROV-DM correctly and completely represents the concepts of the PROV-DM data model. The similarity in scores for both quality requirements aligns with expectations. Given the absence of assigned weights to the concepts and recognizing that their effective application in modeling could lead to accurate representations, the obtained scores suggest an overall acceptable level of quality for the metamodel.

To provide additional detail, we analyzed the responses of the four evaluators identified as experts in the provenance field. Their assessments were consistent with the overall trends observed in the evaluation, with all four experts rating MQR02 and MQR03 close to the maximum score. This suggests strong agreement among provenance experts regarding the metamodel's completeness and correctness in representing PROV-DM concepts. Notably, none of the expert evaluators gave scores below the tolerance threshold for MQR02 and MQR03, reinforcing the robustness of the evaluation results.

[4] `https://tinyurl.com/Ecore4PROVDM-Evaluation`

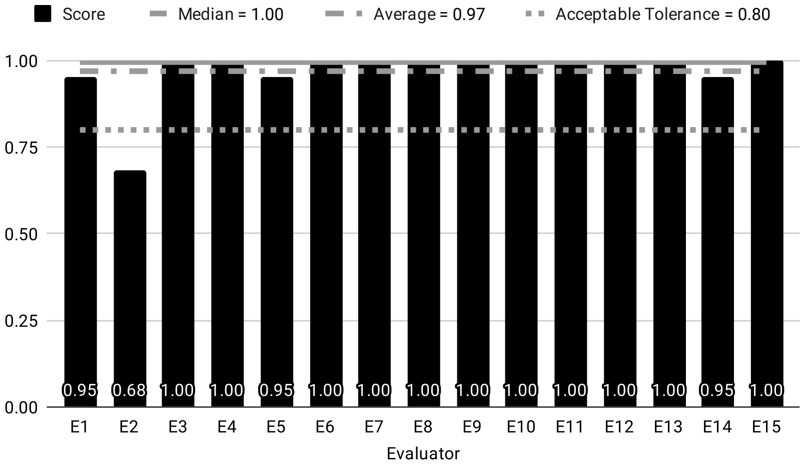


Figure 14. Distribution of evaluators scores for quality requirements MQR02 and MQR03.

In the context of MQR07, evaluators observed that the "Specialization" (*specializationOf*) concept within Component 5, while correctly modeled, is less apparent in the metamodel, with 33.3% of evaluators finding it not easily identifiable. Following this, the "Member" (*hadMember*) concept within Component 6 and the "Alternative" (*alternateOf*) concept within Component 5 were identified as not evident by 26.6% of specialists. However, the majority of the twenty-two concepts across the six components of PROV-DM were deemed clearly evident in Ecore4PROV-DM by evaluators. Figure 15 offers an overview of concepts indicated as less evident in the metamodel.

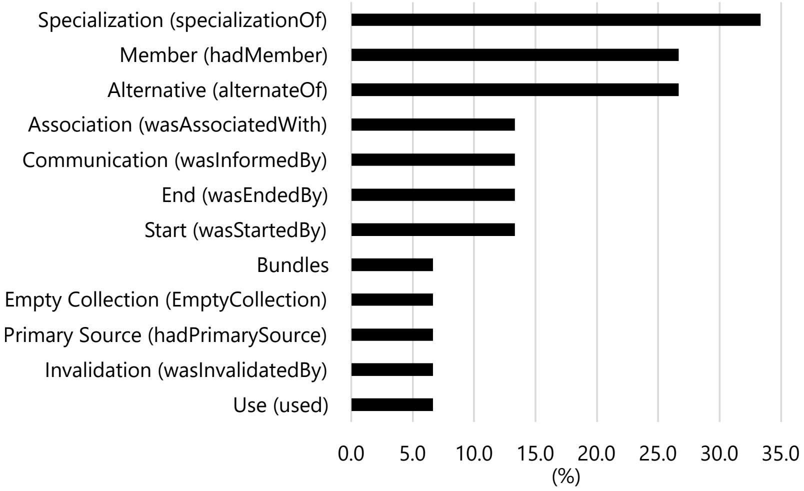


Figure 15. PROV-DM concepts indicated by the evaluators as not evident (MQR07).

In addition to validating Ecore4PROV-DM's alignment with the W3C PROV data model, this evaluation underscores its relevance in the context of Information Systems. By demonstrating how the metamodel facilitates the representation of provenance information, we illustrate its potential to assist developers in overcoming challenges related to data management. Specifically, Ecore4PROV-DM supports the effective tracking of data lineage and access controls, thereby contributing to improved traceability and reproducibility in data-intensive environments.

6.4. Validity Threats

In evaluating the quality of the Ecore4PROV-DM metamodel, certain validity threats were identified that could impact the accuracy of the findings. One significant concern revolves around the non-stratified sample used in the evaluation, resulting in an uneven distribution of evaluators with varying metamodeling experience. To mitigate this threat, future evaluations could employ a more balanced and stratified sample, ensuring representation across diverse expertise levels. This approach would provide a comprehensive and nuanced understanding of the metamodel's effectiveness, minimizing the risk of biased outcomes influenced by the evaluators' varying levels of experience.

Additionally, the absence of a weighting mechanism in the current summation technique, as employed by the MQuaRE framework, poses a potential threat to the significance assessment of individual metamodel concepts. To address this concern, future evaluations could incorporate a refined measurement approach that includes weighted assessments. Introducing a mechanism to assign varying degrees of importance to different metamodel concepts would enhance the accuracy of the significance assessment, ensuring a more nuanced and precise evaluation of their respective contributions. This refinement would contribute to a more robust and comprehensive evaluation, minimizing the risk of undervaluing or overvaluing specific concepts within the Ecore4PROV-DM metamodel.

7. Concluding Remarks

We presented Ecore4PROV-DM, a metamodel designed to adhere to the W3C PROV standard for capturing and representing provenance information in Information Systems. This metamodel specifically addresses the significant challenges developers face in integrating provenance information into Information Systems development processes. By offering a structured approach that aligns with the PROV data model (PROV-DM) across its six components, Ecore4PROV-DM not only facilitates the management of provenance data but also promotes the adoption of standardized representations in software development. Using the Metamodel Quality Requirements and Evaluation (MQuaRE) framework, we meticulously assessed Ecore4PROV-DM, ensuring its fidelity in representing PROV-DM concepts, and evaluating completeness, correctness, and usability. The evaluation results show a high consensus among metamodeling experts on the accuracy and completeness of Ecore4PROV-DM in capturing PROV-DM nuances. While specific concerns were raised about certain concepts, the overall assessment underscores Ecore4PROV-DM as a robust and reliable metamodel for modeling provenance information.

The practical application of Ecore4PROV-DM was demonstrated through two scenarios: (1) modeling data provenance for the Urban Observatory project at Newcastle University, which monitors urban indicators using sensors and CCTV cameras; and (2) creating a PROV-Template for an arithmetic operation. The PROV-N code for this template was automatically generated using model-to-text transformations.

In a recent follow-up to this work [Vieira and Carvalho 2024], Ecore4PROV-DM served as the foundation for developing a Model-Driven Engineering (MDE) tool aimed at facilitating the graphical modeling of provenance information. This tool incorporates model-to-text transformations, automating the conversion of Ecore4PROV-DM instances into PROV-N textual representation. The metamodel establishes a well-structured basis for developing such tools, empowering users to intuitively design and visualize provenance relationships.

Future work should explore integrating Ecore4PROV-DM with existing data management and analysis platforms to foster interoperability and enhance the adoption of standardized provenance representation. Additionally, research efforts can extend the metamodel to accommodate evolving standards or domain-specific requirements, ensuring its relevance and applicability in dynamic research landscapes.

This work provides a well-defined and evaluated Ecore metamodel for PROV-DM, offering a practical tool for integrating provenance data into Information Systems. By supporting consistent modeling of provenance information, Ecore4PROV-DM helps address challenges in ensuring traceability and transparency in software development processes. The systematic structure of Ecore4PROV-DM enhances its utility and reliability, providing a solid foundation for developing systems that require precise provenance representation. Adhering to rigorous quality standards, Ecore4PROV-DM serves as a cornerstone for tool development and a catalyst for ongoing exploration in the Information Systems field, improving the accessibility and usability of provenance information and promoting transparency and reproducibility in scientific and computational endeavors.

References

[ARDC, Australian Research Data Commons 2022] ARDC, Australian Research Data Commons (2022). Data Provenance. Available online: `https://ardc.edu.au/resource/data-provenance/`.

[Bastin et al. 2023] Bastin, L., Reynolds, O., Garcia-Dominguez, A., and Sprinks, J. (2023). Facilitating provenance documentation with a model-driven-engineering approach. In *EGU General Assembly 2023*, pages 24–28, Vienna, Austria. EGU23-8321.

[Bruel et al. 2018] Bruel, J.-M., Combemale, B., Guerra, E., Jézéquel, J.-M., Kienzle, J., de Lara, J., Mussbacher, G., Syriani, E., and Vangheluwe, H. (2018). Model transformation reuse across metamodels. In Rensink, A. and Sánchez Cuadrado, J., editors, *Theory and Practice of Model Transformation*, pages 92–109, Cham. Springer International Publishing.

[Bucchiarone et al. 2020] Bucchiarone, A., Cabot, J., Paige, R. F., and Pierantonio, A. (2020). Grand challenges in model-driven engineering: an analysis of the state of the research. *Software and Systems Modeling*, 19(1):5–13.

[Callahan et al. 2006] Callahan, S. P., Freire, J., Santos, E., Scheidegger, C. E., Silva, C. T., and Vo, H. T. (2006). Vistrails: visualization meets data management. In *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, SIGMOD '06, page 745–747, New York, NY, USA. Association for Computing Machinery.

[Community 2022] Community, T. G. (2022). The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2022 update. *Nucleic Acids Res.*, 50(W1):W345–W351.

[Gil et al. 2013] Gil, Y., Miles, S., Belhajjame, K., Deus, H., Garijo, D., Klyne, G., Missier, P., Soiland-Reyes, S., and Zednik, S. (2013). PROV Model Primer.

[Glavic 2021] Glavic, B. (2021). Data provenance. *Foundations and Trends® in Databases*, 9(3-4):209–441.

[Herschel et al. 2017] Herschel, M., Diestelkämper, R., and Ben Lahmar, H. (2017). A survey on provenance: What for? What form? What from? *The VLDB Journal*, 26(6):881–906.

[Hu et al. 2020] Hu, R., Yan, Z., Ding, W., and Yang, L. T. (2020). A survey on data provenance in IoT. *World Wide Web*, 23(2):1441–1463.

[ISO Central Secretary 2023] ISO Central Secretary (2023). Geographic information – Metadata – Part 3: XML schema implementation for fundamental concepts. Standard, International Organization for Standardization, Geneva, CH.

[Kinderen et al. 2017] Kinderen, S. D., Kaczmarek-Hess, M., Ma, Q., and Razo-Zapata, I. S. (2017). Towards Meta Model Provenance: A Goal-Driven Approach to Document the Provenance of Meta Models. In Poels, G., Gailly, F., Asensio, E. S., and Snoeck, M., editors, *10th IFIP Working Conference on The Practice of Enterprise Modeling (PoEM)*, volume LNBIP-305 of *The Practice of Enterprise Modeling*, pages 49–64, Leuven, Belgium. Springer International Publishing. Part 1: Regular Papers.

[Kudo 2021] Kudo, T. N. (2021). *A metamodel for aligning requirements standards and testing standards and a framework for evaluating metamodels [in Portuguese].* PhD thesis, Universidade Federal de São Carlos, São Carlos – SP, Brazil.

[Kudo et al. 2020a] Kudo, T. N., Bulcão Neto, R. F., and Vincenzi, A. M. R. (2020a). Toward a Metamodel Quality Evaluation Framework: Requirements, Model, Measures, and Process. In *Proceedings of the XXXIV Brazilian Symposium on Software Engineering*, SBES '20, page 102–107, New York, NY, USA. Association for Computing Machinery.

[Kudo et al. 2020b] Kudo, T. N., Bulcão-Neto, R. F., and Vincenzi, A. M. R. (2020b). Metamodel Quality Requirements and Evaluation (MQuaRE). Technical report, Departamento de Computação, UFScar, São Carlos-SP, Brazil. v 2.0.

[López-Fernández et al. 2015] López-Fernández, J. J., Cuadrado, J. S., Guerra, E., and de Lara, J. (2015). Example-driven meta-model development. *Software & Systems Modeling*, 14(4):1323–1347.

[Ludäscher et al. 2006] Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., Lee, E. A., Tao, J., and Zhao, Y. (2006). Scientific workflow management and the kepler system. *Concurr. Comput.*, 18(10):1039–1065.

[Madiot et al. 2024] Madiot, F., Goubet, L., Begaudeau, S., Chauvin, M., Musset, J., and Pupier, A. (2024). Eclipse Acceleo Wiki. Available online: `https://wiki.eclipse.org/Acceleo/`.

[Madiot and Paganelli 2015] Madiot, F. and Paganelli, M. (2015). Eclipse sirius demonstration. *P&D@ MoDELS*, 1554:9–11.

[Moreau 2017] Moreau, L. (2017). PROV-Template: A Quick Start.

[Moreau et al. 2018] Moreau, L., Batlajery, B. V., Huynh, T. D., Michaelides, D., and Packer, H. (2018). A templating system to generate provenance. *IEEE Transactions on Software Engineering*, 44(2):103–121.

[Moreau et al. 2013a] Moreau, L., Missier, P., Belhajjame, K., B'Far, R., Cheney, J., Coppens, S., Cresswell, S., Gil, Y., Groth, P., Lebo, G. K. T., McCusker, J., Miles, S., Myers, J., and Sahoo, S. (2013a). PROV-DM: The PROV Data Model.

[Moreau et al. 2013b] Moreau, L., Missier, P., Cheney, J., and Soiland-Reyes, S. (2013b). PROV-N: The Provenance Notation.

[Pérez et al. 2018] Pérez, B., Rubio, J., and Sáenz-Adán, C. (2018). A systematic review of provenance systems. *Knowledge and Information Systems*, 57(3):495–543.

[Rodrigues da Silva 2015] Rodrigues da Silva, A. (2015). Model-driven engineering: A survey supported by the unified conceptual model. *Computer Languages, Systems & Structures*, 43:139–155.

[Schmidt 2006] Schmidt, D. C. (2006). Guest editor's introduction: Model-driven engineering. *Computer*, 39(2):0025–31.

[Steinberg et al. 2008] Steinberg, D., Budinsky, F., Merks, E., and Paternostro, M. (2008). *EMF: Eclipse Modeling Framework*. Pearson Education, Boston.

[Velasco et al. 2023] Velasco, G. C., Vieira, M. A., and Carvalho, S. T. (2023). Evaluation of a high-level metamodel for developing smart contracts on the ethereum virtual machine. In *Anais do VI Workshop em Blockchain: Teoria, Tecnologias e Aplicações*, pages 29–42, Porto Alegre, RS, Brasil. SBC.

[Vieira and Carvalho 2024] Vieira, M. A. and Carvalho, S. T. (2024). MDE-Based Graphical Tool for Modeling Data Provenance According to the W3C PROV Standard. In *Proceedings of the 12th International Conference on Model-Based Software and Systems Engineering - MODELSWARD*, pages 141–148. INSTICC, SciTePress.

[Völter et al. 2013] Völter, M., Stahl, T., Bettin, J., Haase, A., and Helsen, S. (2013). *Model-driven software development: technology, engineering, management*. John Wiley & Sons.

[Wolf et al. 1998] Wolf, M., Kunze, J. A., Lagoze, C., and Weibel, D. S. (1998). Dublin Core Metadata for Resource Discovery. RFC 2413.

[Wolstencroft et al. 2013] Wolstencroft, K., Haines, R., Fellows, D., Williams, A., Withers, D., Owen, S., Soiland-Reyes, S., Dunlop, I., Nenadic, A., Fisher, P., Bhagat, J., Belhajjame, K., Bacall, F., Hardisty, A., Nieva de la Hidalga, A., Balcazar Vargas, M. P., Sufi, S., and Goble, C. (2013). The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud. *Nucleic Acids Research*, 41(W1):W557–W561.