Software and Systems Modeling
https://doi.org/10.1007/s10270-025-01278-1

SPECIAL SECTION PAPER

# The extended EA ModelSet—a FAIR dataset for researching and reasoning enterprise architecture modeling practices

**Philipp-Lorenz Glaser**[1] · **Emanuel Sallinger**[2] · **Dominik Bork**[1]

Received: 23 March 2024 / Revised: 2 October 2024 / Accepted: 22 January 2025


**Abstract**
Conceptual modeling research is increasingly investigating the application of artificial intelligence (AI) and machine learning (ML) to automate tasks like model creation, completion, analysis, and processing. This trend also applies to enterprise architecture (EA) research. In contrast to its neighboring disciplines, such as business process management, EA lacks proper guidelines, patterns, and best practices to create high-quality EA models. A currently limiting factor for conducting AI-based research to bridge these gaps is the scarcity of openly available models of adequate quality and quantity. With this paper, our aim is to address this limitation by introducing the extended *EA ModelSet*, a curated and FAIR repository of enterprise architecture models represented in the ArchiMate modeling language that can be used by the research and practitioner community. We report on our efforts to build the EA ModelSet and elaborate on exemplary future empirical and ML-based research that can facilitate the dataset. We hope that this paper sparks a community effort toward the further development and maintenance of the EA ModelSet.

**Keywords** Enterprise modeling · Machine learning · FAIR · Enterprise architecture · Dataset · Artificial intelligence · Conceptual modeling · ArchiMate

## 1 Introduction

In recent years, the fields of conceptual modeling and enterprise modeling have seen an increasing interest in exploring the promising applications of artificial intelligence, specifically machine learning (ML) and generative AI, to various tasks such as model creation [1, 35, 37], completion [3, 14], analysis [40, 65], processing [31], and transformation [15, 21]. Selected overviews of the use of AI/ML in conceptual modeling can be found in, e.g., [5, 10, 43, 56, 57] while the general interest of the research community in this topic is discussed in, e.g., [32, 44, 48, 63]. ML has the potential to revolutionize the way enterprise modeling is approached and implemented. However, a significant challenge that hinders progress in this domain is the scarcity of readily available data, specifically high-quality and diverse models in sufficient quantities [12]. As stated by López et al., *"However, current datasets are either too small or not properly curated."* [39, 41] Moreover, even if such datasets are available, their proper curation and maintenance establish further challenges.

The success of ML approaches heavily relies on large and diverse datasets that capture the intricacies and complexities of real-world scenarios. This also holds true for the conceptual modeling community to enable robust and data-driven modeling research. The lack of datasets, moreover, limits our ability to conduct empirical modeling research. Given the wide adoption of enterprise architecture (EA) modeling and enterprise architecture management (EAM) in practice, it is surprising how little is reported in the literature on best practices, modeling patterns, or guidelines. Compared to business process management (BPM) or software and systems modeling, where seminal contributions were made a decade ago [8,

Communicated by Monika Kaczmarek-Heß, João Paulo Almeida, and Erik Proper.

✉ Dominik Bork
dominik.bork@tuwien.ac.at

Philipp-Lorenz Glaser
philipp-lorenz.glaser@tuwien.ac.at

Emanuel Sallinger
emanuel.sallinger@tuwien.ac.at

[1] Business Informatics Group, TU Wien, Favoritenstrasse 9-11, Vienna 1040, Vienna, Austria

[2] Database and Artificial Intelligence Group, TU Wien, Favoritenstrasse 9-11, Vienna 1040, Vienna, Austria

Published online: 26 February 2025

26, 45], there is an unexplored potential for developing such best practices, patterns, and guidelines for EA modeling. Pattern catalogs in the area of EA are primarily based on management patterns, theoretical deliberations, or limited case-based deductions [13, 49, 61]. Existing EA pattern literature is primarily based on case studies, interviews, and theoretical deliberations, lacking empirical grounding [23]. The lack of publicly available, free-to-access datasets establishes a major bottleneck in advancing ML and empirical research in the EA domain. Without access to a substantial collection of models, researchers face significant challenges in developing and evaluating new ideas, concepts, and algorithms. Furthermore, continuous curation and maintenance of such a dataset are crucial for further research advancements.

To address the challenges mentioned at the outset, researchers from all disciplines recognize the importance of adhering to the principles of Findable, Accessible, Interoperable, and Reusable (FAIR) [60, 67, 69, 70] data management. A FAIR dataset ensures that data are discoverable and accessible to all interested researchers, fostering collaboration, and enabling the reproducibility of results. Additionally, an FAIR dataset is designed to be interoperable, facilitating seamless integration with various tools and techniques. In addition, by making the dataset reusable, researchers can build on existing work and accelerate the development of innovative solutions.

A FAIR dataset of EA models is crucial for several reasons. Firstly, it addresses the issue of data scarcity by collating a comprehensive collection of diverse and high-quality EA models from various domains and industries. Secondly, adhering to the principles of FAIR ensures that the dataset is accessible to the conceptual modeling community under well-defined conditions, which may include open or restricted access depending on licensing, and encourages active engagement and contribution from researchers worldwide. The introduction of a FAIR EA ModelSet opens up a plethora of possibilities for ML-based and empirical research. Researchers can leverage this curated repository to train and validate ML models, enabling automated tasks such as generating new EA models, analyzing complex relationships within models, processing large volumes of data efficiently, and transforming models to adapt to evolving business requirements.

Furthermore, the availability of a FAIR dataset fosters the growth of a collaborative and innovative research community dedicated to exploring the potential applications of ML in EA management. By providing a common foundation for experiments and evaluations, the FAIR EA ModelSet empowers researchers to benchmark their methods against existing approaches, driving continuous improvement and development in the field. This research has the potential to hold significant value for the conceptual modeling community, especially if it continues to evolve, is being maintained, gains adoption, and not only addresses data scarcity but also paves the way for a more collaborative and dynamic research landscape. Through this research, our goal is to inspire and encourage a collective effort toward the development, curation, and maintenance of a comprehensive and freely available dataset, sparking new avenues of exploration and innovation at the intersection of artificial intelligence and conceptual modeling [10, 43]).

In this paper, we present an extension of our previous work [24] in which we introduced an open, curated repository of EA models following the FAIR principles—the EA ModelSet. We extend our previous work in several ways, the results of which form part of this paper: (i) we collect more models, also through peers in the enterprise architecture community; (ii) we implement means of improving the quality of the models in the EA ModelSet by realizing a labeling application that is connected to the EA ModelSet and allows efficient investigation and analysis of EA models as well as future maintenance and extension of the dataset; and (iii) we elaborate on current applications of the EA ModelSet, some of which include our own ongoing research efforts. In total, the EA ModelSet now comprises 977 models that we collected, harmonized, integrated, and publicized on an FAIR basis. In the paper at hand, we further detail the means of efficiently exploiting the EA ModelSet by using the Webpage, the Java Command-Line Interface, and the Python library we developed.

In the remainder of this paper, we discuss the approach we followed to collect, process, and manage the EA ModelSet in Sect. 2. Section 3 then introduces the characteristics of the EA ModelSet. An evaluation according to the FAIR principles is presented in Sect. 4. Several of the usage scenarios enabled by our EA ModelSet, reflections, and future research directions are discussed in Sect. 5. Related model repositories are briefly referred to in Sect. 6. Eventually, we conclude this paper in Sect. 7.

## 2 Creating the model set

This section describes the approach we followed while creating the EA ModelSet dataset. The approach (see Fig. 1) consists of three main stages:

1. *Data Collection*: initial EA models are retrieved from different data sources and stored, serving as raw data for the dataset.
2. *Data Processing*: the collected models are processed and transformed into a standardized format to be compatible with advanced analysis and ML tasks.
3. *Dataset Management & Publishing*: execution of data cleansing tasks that involve different curation and stew-

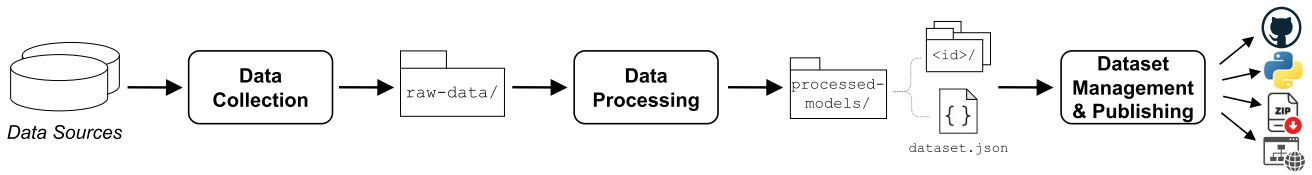


**Fig. 1** Overview of the EA ModelSet creation

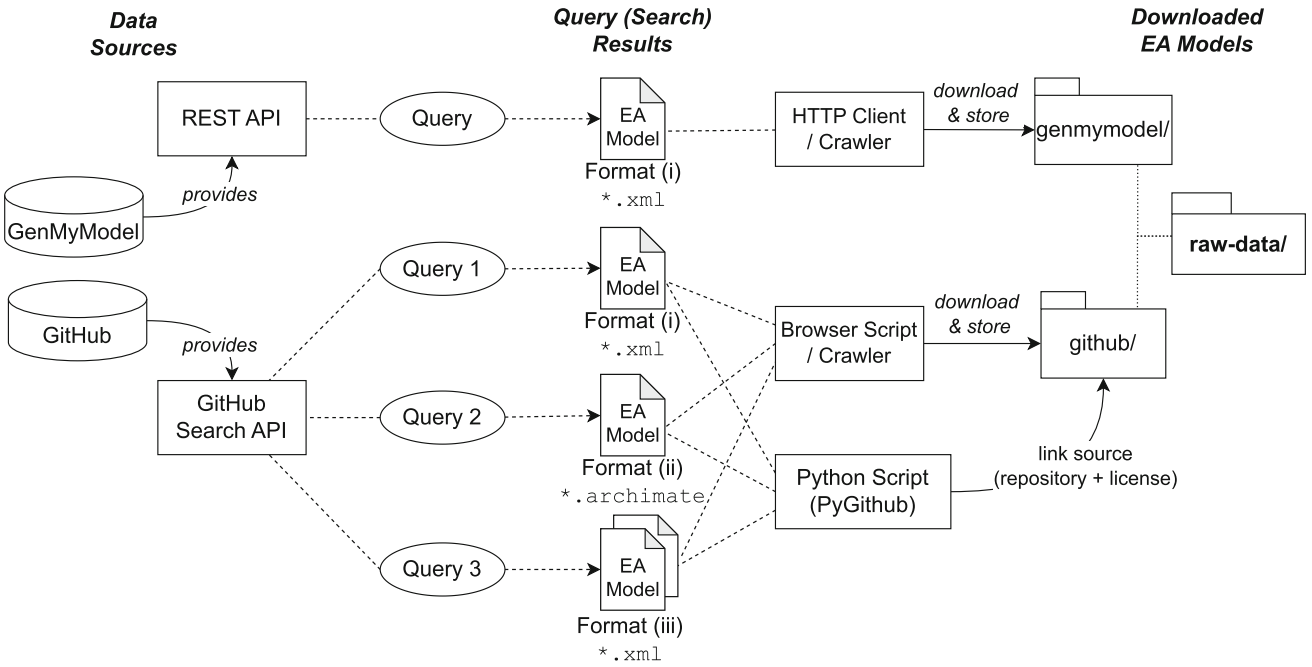


**Fig. 2** Data collection workflow

ardship tasks, and publishing the EA ModelSet dataset with its accompanying services.

In the remainder of this section, we describe each of the three stages of the creation of the EA ModelSet in detail.

## 2.1 Dataset collection

The data collection process revolves around the retrieval and storage of EA models from diverse data sources. These models serve as the raw data input for subsequent processing activities in our dataset. In our process, we programmatically retrieved models from *GitHub* and *GenMyModel* (see Fig. 2) and manually collected models from diverse other sources. The models collected from each source and their formats are summarized in Table 1.

The programmatically retrieved data sources, GitHub and GenMyModel, serve as valuable data sources due to their extensive collections of ArchiMate models, which can also be retrieved with reasonable effort. GitHub, a popular platform for hosting and sharing code repositories, hosts numerous open-source projects and provides an API for searching code globally across all indexed repositories. Utilizing the provided search functionality, we formulated specific queries to retrieve ArchiMate models in different commonly used formats. We focused on acquiring models in three specific formats, each encapsulating the same informational content concerning elements, relationships, and views/diagrams, such as names, types, and other relevant attributes. These formats include:

**Table 1** Collected models, their source, and format

| Source | Format | | | |
|---|---|---|---|---|
| | (i) | (ii) | (iii) | Total |
| GitHub | 225 | 704 | 86 | 1,015 |
| GenMyModel | 287 | 0 | 0 | 287 |
| Other | 14 | 42 | 0 | 56 |

(i) *The Open Group Standard ArchiMate Model Exchange File Format* – This format is a standardized XML schema designed to facilitate model exchange among tools, with mandatory adoption by certified tools since

June 1, 2018.[1] The exchange file format is organized into three distinct schemas: model exchange, view exchange, and diagram exchange. Models in this format encapsulate all relevant modeling aspects within a single XML file ending in `*.xml`

(ii) *Archi model storage format* – Utilized by the Archi modeling tool,[2] this format represents another XML-based storage mechanism. Unlike the broader exchange format mentioned previously (i.e., format (i)), the Archi model storage format is a proprietary format, specifically for use with the Archi tool ecosystem, storing a complete model within a single XML file ending in `*.archimate`

(iii) *Git Friendly Archi File Collection (GRAFICO) format* – This format was developed within the Model Collaboration Archi plugin[3] for improved version control and collaboration. The different ArchiMate layers are segregated into separate directories, and each ArchiMate element, relationship, and view is stored in an individual XML file ending in `*.xml`.

To search for specific file formats on GitHub, their respective file extensions (i.e., `*.xml` and `*.archimate`) were included in the search query. It is important to note that GitHub's search index currently does not include large repositories and files larger than 350 kiB, which means that large models may be excluded from search results. However, it is possible to retrieve large ArchiMate models that are stored in the GRAFICO format (i.e., format (iii)) since they are split across several files, which bypasses this limitation.

Initially, we partly automated the collection process by downloading the individual files from the search results through a browser script, and then used the Python library PyGithub[4] to automatically retrieve models from GitHub, associate them with their respective repositories, and if present, link the corresponding license information. Now, our strategy for acquiring these models has been significantly improved by implementing specialized crawlers for fully automated retrieval. In total, we collected 1,015 models from GitHub, stored in the `raw-data/github/` directory. From these 1,015 models, 225 are in format (i), 704 in format (ii), and 86 in format (iii). Models in GRAFICO format (i.e., format (iii)) were transformed into format (ii) using the Archi Command-Line Interface (CLI) tool to not introduce any additional complexity for later activities (e.g., not requiring an additional parser).

GenMyModel [19], an online modeling platform that supports various modeling languages, serves as another data source. Interfaced through its REST API,[5] we filtered for public ArchiMate projects and retrieved the models in the standard model exchange XML format (i.e., format (i)). Similarly to the GitHub retrieval, a specialized crawler has improved the process of collecting models from GenMyModel to reduce the effort previously required. In total, we collected 287 models from GenMyModel, stored in the `raw-data/genmymodel/` directory.

In addition to these primary data sources, from which we programmatically retrieved models, we manually collected models from other sources, including forums, publications, and project/company websites. These models were obtained through targeted web searches. We also reached out to the EA community to receive direct contributions from practitioners and researchers in the field. In total, we collected 56 models from other sources, 14 in format (i) and 42 in format (ii), and stored them in the `raw-data/other/` directory.

## 2.2 Dataset processing

With a substantial collection of more than 1, 000 ArchiMate models in different formats, the subsequent step involves processing these models to transform them into a standardized format suitable for advanced analysis and ML tasks. The data processing phase (see Fig. 3) is initiated by receiving the collected models from the `raw-data/` directory as input, with file duplicates discarded beforehand by comparing their MD5 file hashes. Each raw ArchiMate model is then processed as follows (the numbers refer to Fig. 3):

❶ **Parsing**: The file is parsed to extract relevant information and to create an intermediate `ParsedModel` representation. Since all our input files are either in format (i) or (ii), two separate XML parsers are used. Although the formats differ in their hierarchical structure and naming schemes, they contain the same information and, therefore, can be parsed into a unified representation (i.e., a `ParsedModel`). The file is skipped if any major errors occur during parsing (e.g., invalid XML). Similarly, if the parsed model does not have at least one view or the number of elements is less than 10, the file is also skipped, as such low numbers indicate a model with insufficient complexity/quality.

❷ **Duplicate Detection**: The parsed model's ID is checked against IDs of models that have already been processed. If a duplicate is found, the existing JSON representation of the model is updated by adding the duplicate model's file path to the list of detected duplicates, and the model is tagged with a `DUPLICATE` label. The processing workflow then continues with the next file.

[1] https://www.opengroup.org/open-group-archimate-model-exchange-file-format.

[2] https://www.archimatetool.com/.

[3] https://github.com/archimatetool/archi-modelrepository-plugin.

[4] https://github.com/PyGithub/PyGithub.

[5] https://app.genmymodel.com/api/projects/public.

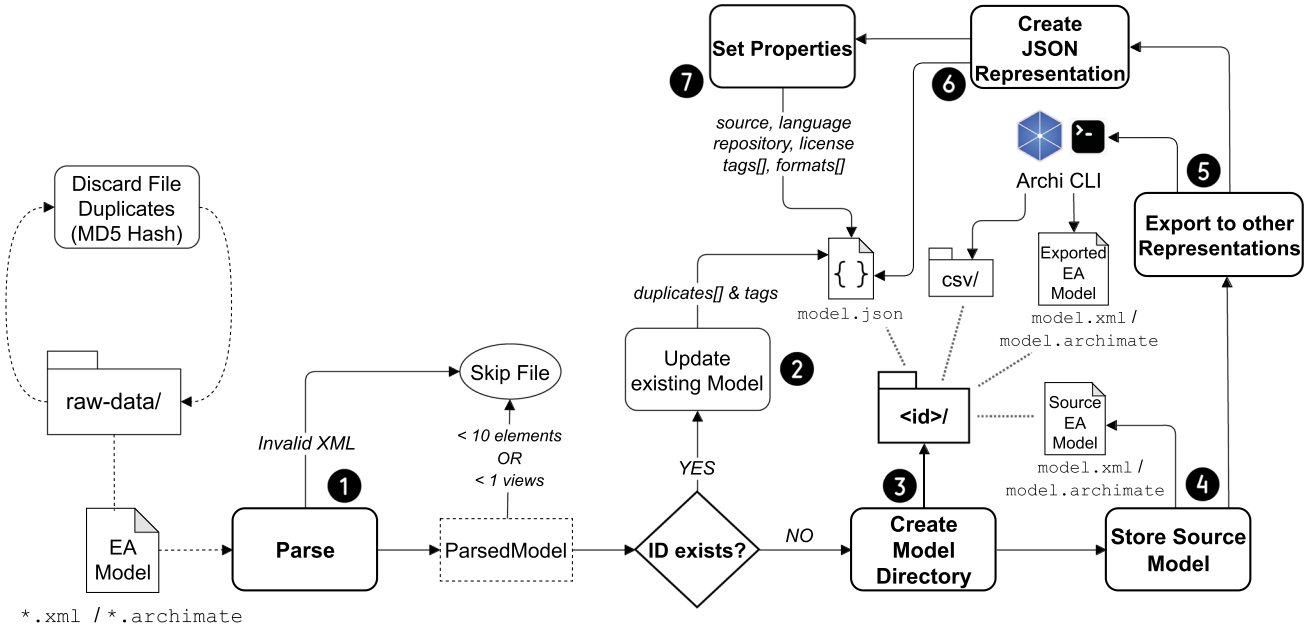


**Fig. 3** Data processing workflow

❸ **Directory Creation**: For each unique model, a new directory is created with the ID as its name. This directory is used to store and locate the model in various formats.

❹ **Storage & ❺ Export of Formats**: The source file from which the model was parsed is stored first, either as `model.xml` or `model.archimate`. Additionally, to improve interoperability, the model is exported into the respective other ArchiMate model format (i.e., as `model.xml` or `model.archimate`) using the Archi CLI tool.[6] Elements, relations, and properties of the model are exported as separate CSV files (named `elements.csv`, `relations.csv`, and `properties.csv`, respectively) and stored within a directory named `csv/`.

❻ **JSON Representation**: The last file that is created in the model's directory is a JSON representation of the model, named `model.json` and conforming to a defined JSON schema in `ea-model.schema.json`. The JSON representation includes additional properties to further classify certain model characteristics in the dataset, in addition to common ArchiMate model properties already present in the parsed source file (see Sect. 3.1 for more information regarding the JSON schema).

❼ **Set Properties**: For the first release of the dataset, we relied on simple mechanisms to set the properties: The `source` property is set to the path of the parsed source file, for warnings during the parsing process (e.g., a relationship could not be parsed due to invalid source/target ID) we added a `WARNING` label to the list of `tags`, a corresponding `repository` URL and `license` is linked, the list of `formats` is based on the successfully exported formats of the previous step, and at last we set the `language` property by merging the names of a model's elements into a single textual representation to serve as input for the language detection Java library Lingua[7] that provides us an estimate of the used language in a model.

After data processing, a total of 977 unique models remained, which are stored in the `processed-models/` directory. Each model has its own subdirectory, denoted by the model's ID, and contains the different representations of the model created during the processing stage (i.e., JSON, two different XML, and CSV).

## 2.3 Dataset management and publishing

The final stage of our approach focuses on managing and publishing the EA ModelSet dataset with its accompanying services (see Fig. 4). The dataset is stored within the `EAModelSet` GitHub repository[10] in a central directory called `dataset/`. This directory includes the `processed-models/` directory from the previous stage and a `dataset.json` file which adheres to the JSON schema specified in `ea-dataset.schema.json` (see Sect. 3.1) containing metadata and computed data about the dataset itself. It also includes brief information about each model and a subset of its characteristics, facilitating model search. The

[6] https://github.com/archimatetool/archi/wiki/Archi-Command-Line-Interface.

[7] https://github.com/pemistahl/lingua.

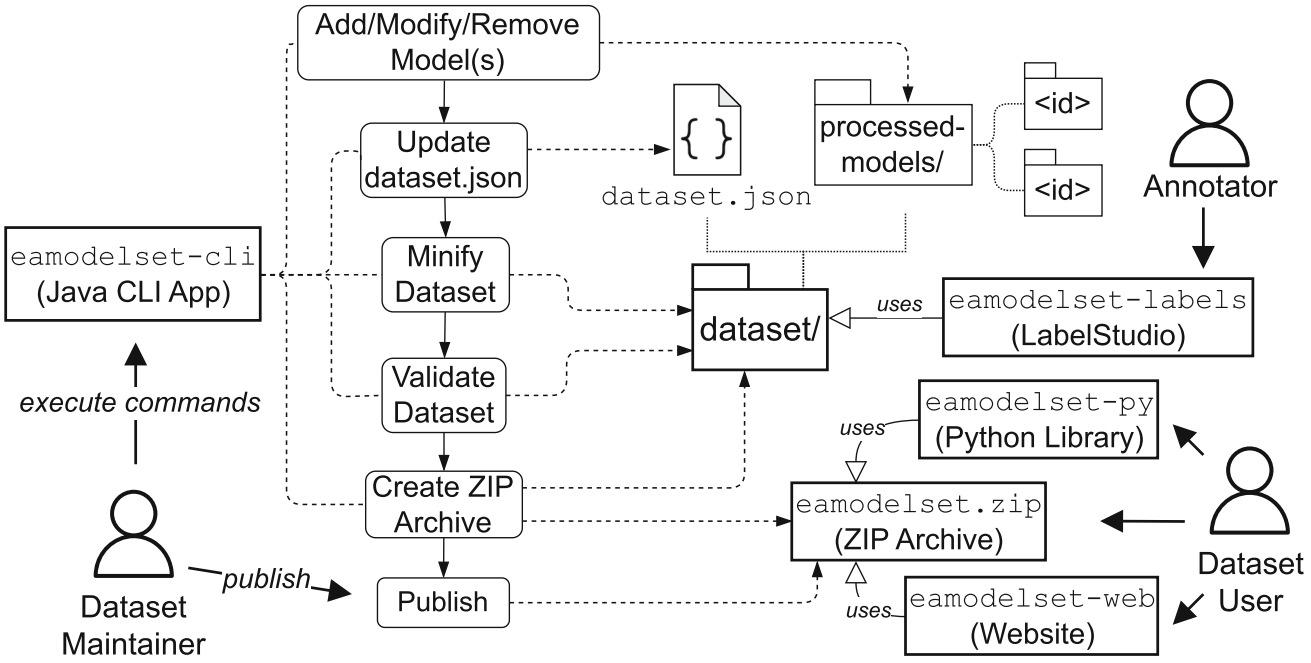


**Fig. 4** Dataset management and publishing workflow

data in the `dataset.json` are further used by the website for model search and the Python library for searching within the pandas dataframe.

Dataset management activities are primarily performed using the accompanying *Java CLI application*,[10] enabling maintainers to add, modify, or remove models from the dataset. When preparing for a new release, the `dataset.json` file is updated to reflect the changes made to the dataset. Following the update, the processed models undergo a minification process to reduce their file size and optimize storage efficiency. Minification involves removing unnecessary white spaces, comments, and other non-essential elements from the model files, improving their compactness. The dataset then undergoes a validation process to ensure its quality and consistency. During the transformation process, unforeseen errors may occur due to inconsistencies in the raw data or schema evolution. The validation process checks each individual model and the dataset as a whole against the defined JSON schemas to ensure conformity and verifies the file structure and presence of all required files. Any models that do not adhere to the defined schema or have missing files are flagged for further manual investigation or correction, ensuring the overall integrity of the data.

Once the dataset is prepared and validated, it is compressed into a single file archive named `ea-modelset.zip`. Compression further reduces the overall file size, making it easier to distribute and download the dataset while preserving its content and structure. After following the described stages, the EA ModelSet dataset is effectively organized, summarized, validated, compressed, and ready to be made publicly accessible. It is published as a new GitHub release to ensure the dataset's availability, version control, and visibility to the wider community. The prepared ZIP archive is also utilized by the accompanying applications, such as the website and Python library (see Sect. 3.3). Besides users, annotators can collaboratively utilize the EA ModelSet dataset through the `eamodelset-labels` service, which integrates with LabelStudio. Using the Archi CLI, the diagrams (i.e., views) of models in the dataset are exported to PNG images and made available in the LabelStudio interface for annotation (see Fig. 5). Currently, these annotations include domain categories (e.g., insurance, banking, etc.) to help classify models according to their industry or usage context, and labels to mark incomplete or erroneous models. The annotated data can later be exported for use in empirical research and supervised machine learning tasks (e.g., automated domain classification, outlier detection) requiring labeled data.

In addition to the above-mentioned management and publishing activities depicted in Fig. 4, the provided Java CLI application can generate PNG images of the model's views/diagrams (using the Archi CLI in the background), which can further enhance the versatility of the EA ModelSet. However, these diagrams are not included in the distributed ZIP archive yet, as they significantly increase the file size and more efficient storage means must be established first.

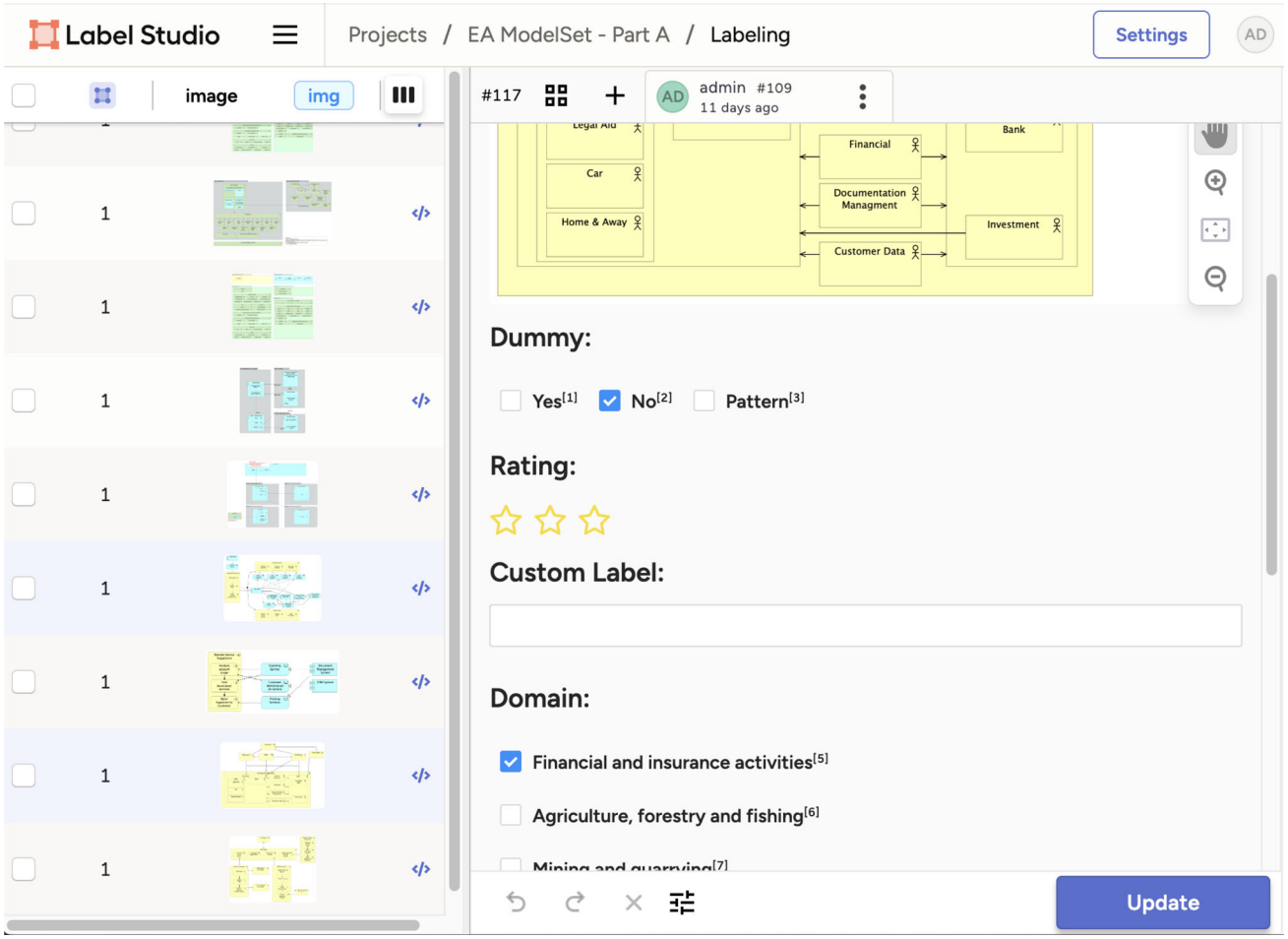


**Fig. 5** The developed LabelStudio UI that can be used to label the models in the EA ModelSet

To improve the dataset's utility, we incorporated LabelStudio,[8] an open-source data labeling tool, in which we imported the generated diagrams to provide efficient mechanisms for collaboratively annotating the models in the dataset. The graphical interface of LabelStudio, as depicted in Fig. 5, illustrates the practical application of this tool, allowing it to annotate domain information (e.g., insurance, hotel, banking) in a model's view. These annotations can be exported and utilized, e.g., in empirical analysis or machine learning algorithms, to perform tasks such as automatic domain classification [40]. Moreover, dummy models (i.e., simple sketches that do not represent actual enterprise architectures in any form) can easily be marked and later filtered out to improve the quality of the models in the dataset.

## 3 EA ModelSet

We now introduce the curated and FAIR EA ModelSet—a dataset of ArchiMate models.

[8] https://labelstud.io/.

```
EAModelSet/
├── dataset/
│   ├── dataset.json
│   └── processed-models/
└── raw-data/
    ├── github/
    │   ├── archimate/
    │   ├── grafico/
    │   └── xml/
    ├── genmymodel/
    └── other/
```

**Fig. 6** Root Directory Structure

### 3.1 Dataset structure and schema

The EA ModelSet dataset is organized according to a well-defined structure and leverages JSON Schemas [50] to facilitate efficient data management and to provide a FAIR dataset of EA models.

**Root Directory Structure** (see Fig. 6): The `raw-data/` directory holds the collected raw data models that were

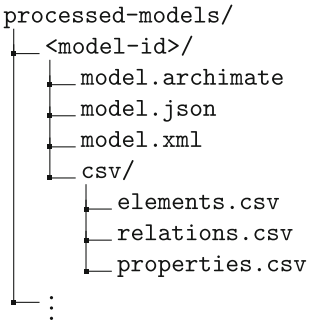


**Fig. 7** Model Directory Structure

used for data processing. It includes subdirectories for different data sources, such as `github/` (i.e., from GitHub), `genmymodel/` (i.e., from GenMyModel), and `other/` (i.e., from miscellaneous sources). The models from GitHub are further organized in three sub-directories `archimate/`, `grafico/`, and `xml/` based on their respective file format. The main directory for the dataset is the `dataset/` directory, which contains the `dataset.json` file. Within the `processed-models/` directory, each processed model has its own subdirectory and follows a consistent format.

**Model Directory Structure** (see Fig. 7): A model directory contains the primary JSON model file (`model.json`) and two ArchiMate XML model files (`model.archimate` and `model.xml`). Additionally, models and their contents are stored in separate CSV files within the `csv/` directory.

Two principal JSON files are used to encapsulate the dataset's information: the `model.json` for individual models and the `dataset.json` for the dataset as a whole. Figure 8 illustrates how the JSON schemas are positioned in relation to the dataset to ensure consistency of metadata and data. The `ea-modelset.schema.json` and `ea-model.schema.json` schema files define the structure and validation rules for content in the `dataset.json` and `model.json` files, respectively.

The *Dataset* object contains the dataset metadata and includes information such as the title, version, lastUpdated date, repository URL, homepage URL, distribution details (including distribution title, download URL, media type, and byte size), model count, and an array of *ModelInfo* objects that provide a reduced subset of metadata and computed properties of each individual model. The *EA Model* object provides comprehensive information about each model including its elements, relationships, and views.

## 3.2 Dataset description and statistics

The final EA ModelSet dataset is composed of 977 unique ArchiMate models. Table 2 provides some descriptive statistics of the dataset, including the sum, average, minimal, and maximum number of elements, relationships, and views. Figure 10 further shows the distribution of the models by means of relating the number of model elements on the x-axis to the number of model views on the y-axis. It can be derived from these statistics that the dataset features models of varying size (from 10 up to 4,003 elements, from zero to 5,773 relationships) and the number of views (from one to 357). Figure 9 shows how often each ArchiMate element type occurs in the models. We can derive, that there is a majority of models in the EA ModelSet dataset with elements from the ArchiMate Business (yellow color) and Application layers (cyan color). Investigating the individual ArchiMate concepts, we can derive that the `BusinessProcess`, `ApplicationComponent`, and `BusinessObject` element types are used the most often, while other types (e.g., `TechnologyInteraction` or `DistributionNetwork`) are rarely present. Similarly, Fig. 11 shows the occurrence frequency of each ArchiMate relationship type, with `Composition`, `Association` and `Realization` types being the most prominent, while `Specialization` and `Influence` relationship types being used least often. Further statistics are provided at the EA ModelSet homepage.[9]

## 3.3 Dataset usage

The EA ModelSet facilitates various usage scenarios by providing accompanying services and applications. In this section, we describe the support we provided to efficiently access and utilize the dataset. The dataset and all its related services and applications can be found in the central EA ModelSet GitHub repository, accessible through the assigned pURL.[10]

**Download Dataset:** The dataset can be downloaded as a compressed ZIP file from the GitHub repository's release section[10] with a Git tag introduced for each new version. The ZIP file contains all the necessary files and directories to access and explore the dataset locally (except `raw-data/` files). It serves as the primary method for obtaining the dataset and forms the basis for the accompanying services and applications.

**Python Library:** A dedicated Python library is provided to facilitate programmatic access and analysis of the dataset within a Python environment. This library provides an interface for interacting with the dataset as a pandas dataframe, presenting the data in a structured tabular format that can be easily manipulated for various analytical tasks. Users can use additional filtering functionality to filter the dataset on different attributes such as the model's source, language, or the minimum/maximum number of elements, relationships, or views. Additionally, one can retrieve complete JSON or CSV

[9] https://me-big-tuwien-ac-at.github.io/EAModelSet/home.

[10] https://purl.org/eamodelset.

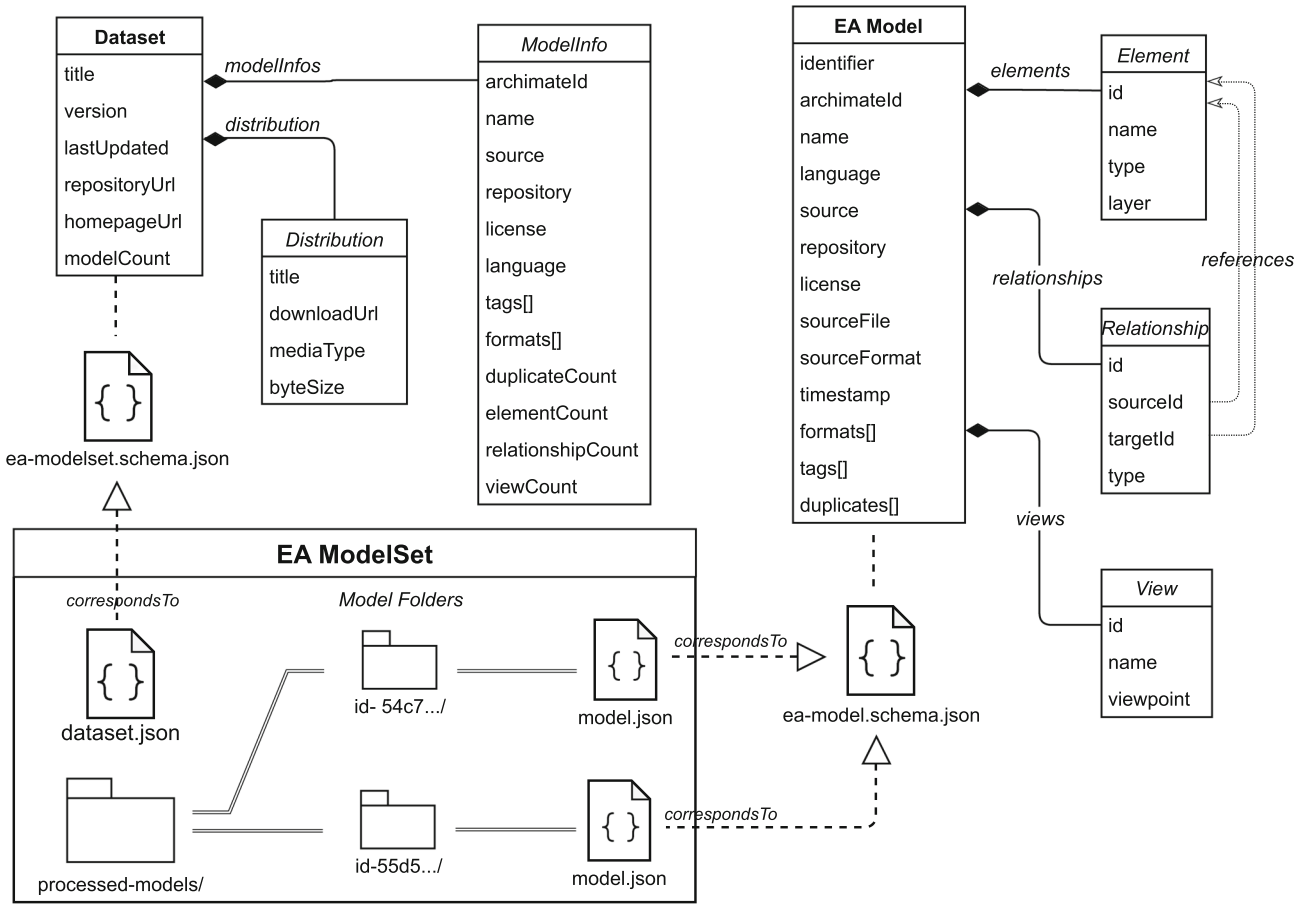


**Fig. 8** JSON Schema

**Table 2** Descriptive statistics of the EA ModelSet models

| | Elements | Relationships | Views |
|---|---|---|---|
| Sum | 104,196 | 136,539 | 7,150 |
| Avg | 106.65 | 139.75 | 7.32 |
| Min | 10 | 0 | 1 |
| Max | 4,003 | 5,773 | 357 |
| Mode | 16 | 20 | 1 |
| Median | 50 | 57 | 2 |
| Stdev | 252.4 | 393.11 | 22.87 |

data representations of a model using the model's unique `ID`, which can be obtained from the dataframe.

In the following, we provide a basic usage example of the library. A more detailed example can be found in the provided Jupyter Notebook.[11] As can be seen, it is very easy and efficient to use our library to load the dataset into a pandas dataset and to query, filter, and analyze the data.

[11] https://github.com/me-big-tuwien-ac-at/EAModelSet/blob/main/python-lib/examples/python-example.ipynb.

```python
# Import the library
from eamodelset.dataset import EAModelSet

# Create dataset instance
dataset = EAModelSet()

# Access pandas dataframe
df = dataset.data

# Retrieve all models
models = dataset.filter_models()

# Basic filtering for models using English
models = dataset.filter_models(lang='en')

# Combination of multiple filters
models = dataset.filter_models(
    name='archi',
    lang='en',
    source='github',
    min_elems=100,
```

```
    max_views=10
)

# Retrieve JSON/CSV of a specific model
model = dataset.get_model('id-123')
model2 = dataset.get_csv_model('id-123')

# Access model properties
name = model['name']
language = model['language']
elements = models['elements']
```

**Java CLI:** For managing and maintaining the dataset, a Java Command-Line Interface (CLI) was realized. The CLI enables users to issue command line commands (see Table 3) to perform operations on the dataset like adding or removing models, updating metadata, generating statistics, or validating the dataset's integrity (cf. Sect. 2.3). The Java CLI also allows connecting and loading the data into a MongoDB document database or a Neo4j Graph Database for advanced querying and analysis. The use of the functionality of the Java CLI is now briefly demonstrated, a more detailed demonstration is provided in the GitHub repository.[12]

[12] https://github.com/me-big-tuwien-ac-at/EAModelSet/tree/main/cli-app.

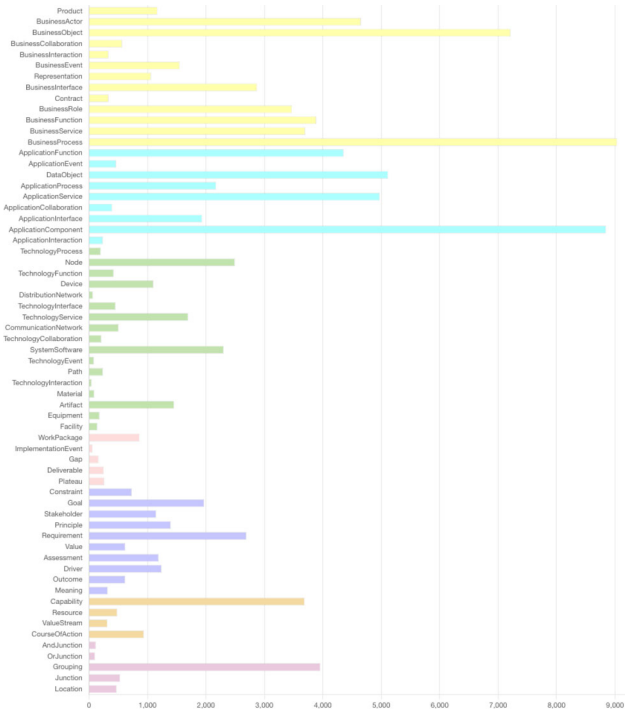


**Fig. 9** ArchiMate Element Type Frequency

The following listing of Java CLI commands illustrates the process for managing the dataset (cf. Fig. 4). The `load` command initializes the environment by loading the models of the dataset into a MongoDB instance. Once the dataset is loaded, the remaining commands become enabled. The dataset can then be dynamically added or removed through the `addModel` or `removeModel` commands. The `datasetJson` command updates the central `dataset.json` file, ensuring that metadata remains synchronized with the dataset's current state.

```
# load dataset
load /path/to/dataset/

# add model
addModel /path/to/model.archimate
addModel /path/to/model.xml

# remove model
removeModel id-1234

# update dataset.json file
datasetJson

# validate the whole dataset
validate

# create zip archive
zip /path/to/target.zip
```

The `validate` operation checks for the file structure and JSON schema of each model and the dataset itself. Lastly, the `zip` command creates a compressed archive of the whole dataset, making it ready for release and distribution. Table 3 lists all commands of our developed Java CLI with a brief description.

**Website:** The EA ModelSet has a dedicated website[9] (Fig. 12) that also serves as the landing page for the dataset, offering a user-friendly interface for easy exploration of the models. The website is divided into four sections:

*(i) Home:* The home section serves as the dataset's landing page and as a starting point for users to get acquainted with the dataset. It lists the dataset's metadata, which is read from the `dataset.json` file, ensuring that the information can be easily updated in subsequent releases. The home section also includes a button to download the dataset as a ZIP file (also linked through the JSON file to the released distribution on GitHub).

*(ii) Search:* The search interface enables efficient exploration and retrieval of relevant models in the dataset (see

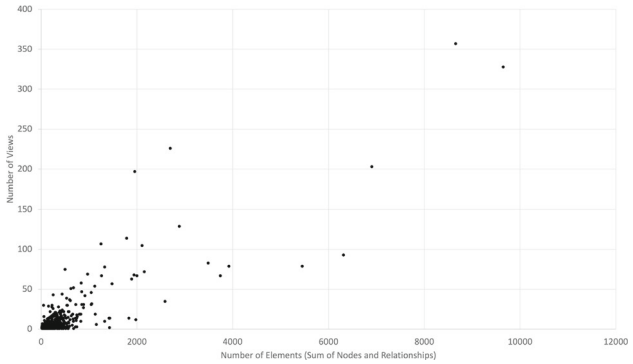


**Fig. 10** Distribution of the models with respect to the number of model elements and views

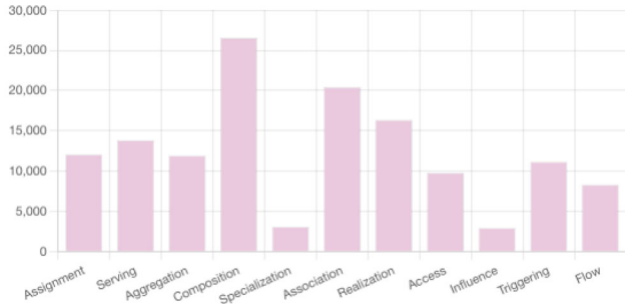


**Fig. 11** ArchiMate relationship type frequency

Fig. 12). Users can search for specific models based on various criteria, such as model ID, name, tags, language, source, license, or the minimum/maximum number of elements, relationships, or views. The search functionality supports arbitrary combinations of filtering criteria, sorting columns, and a "global search" feature to filter all fields.

*(iii) Model Details:* This page allows in-depth analysis of each model through the data extracted from the respective `model.json` file. It can be accessed by navigating from the search section or by following the URL of the model's identifier (https://me-big-tuwien-ac-at.github.io/EAModelSet/model/<id>). The upper part of the details page (see Fig. 13) provides a direct download of the associated file formats and lists the metadata and data related to a specific model. The lower part (see Fig. 14) includes images of the model's views that can be enlarged for better visibility and lists the data of elements, relationships, and views in tabular form.

*(iv) Statistics:* The statistics page provides insights into the dataset's composition, complexity, and characteristics by presenting key statistics and metrics. Users can explore charts showing the usage of specific languages, layers, element/relationship types, or concrete values for the total number of models, as well as the total, minimum, maximum, and average number of elements, relationships, and views.

## 4 Evaluation against the FAIR principles

The FAIR principles provide guidelines to improve the **F**indability, **A**ccessibility, **I**nteroperability, and **R**euse of digital assets [70]. The FAIR principles further emphasize machine-actionability in scientific data management to support dealing with increased volume, complexity, and creation speed of data. In the following, we evaluate the compliance of the EA ModelSet dataset regarding each FAIR principle.

### 4.1 Findability

*F1: "(meta)data are assigned a globally unique and persistent identifier."* The EA ModelSet meets this requirement by assigning a Persistent Uniform Resource Locator (pURL)

**Table 3** Java CLI commands (excerpt)

| Command | Description |
|---|---|
| `help` | Display help about available commands |
| `history` | Display or save the history of previously run commands |
| `version` | Show version information about the CLI application |
| `script<path>` | Read and execute commands from a file |
| `load<path>` | Load the given dataset directory into MongoDB |
| `loadNeo4j<path>` | Load the given dataset directory into Neo4j |
| `addModel<path>` | Process and add the given model file to the loaded dataset |
| `removeModel<id>` | Remove a model from the loaded dataset through the given ID |
| `validate` | Validate the structure and schema of the loaded dataset, including all its models |
| `validateModel<id>` | Validate a specific model in the dataset through the given ID |
| `datasetJson` | Create a dataset.json file for the loaded dataset |
| `datasetStats` | Create dataset statistics file for the loaded dataset |
| `zip<path>` | Create a ZIP archive of the loaded dataset in the given directory |

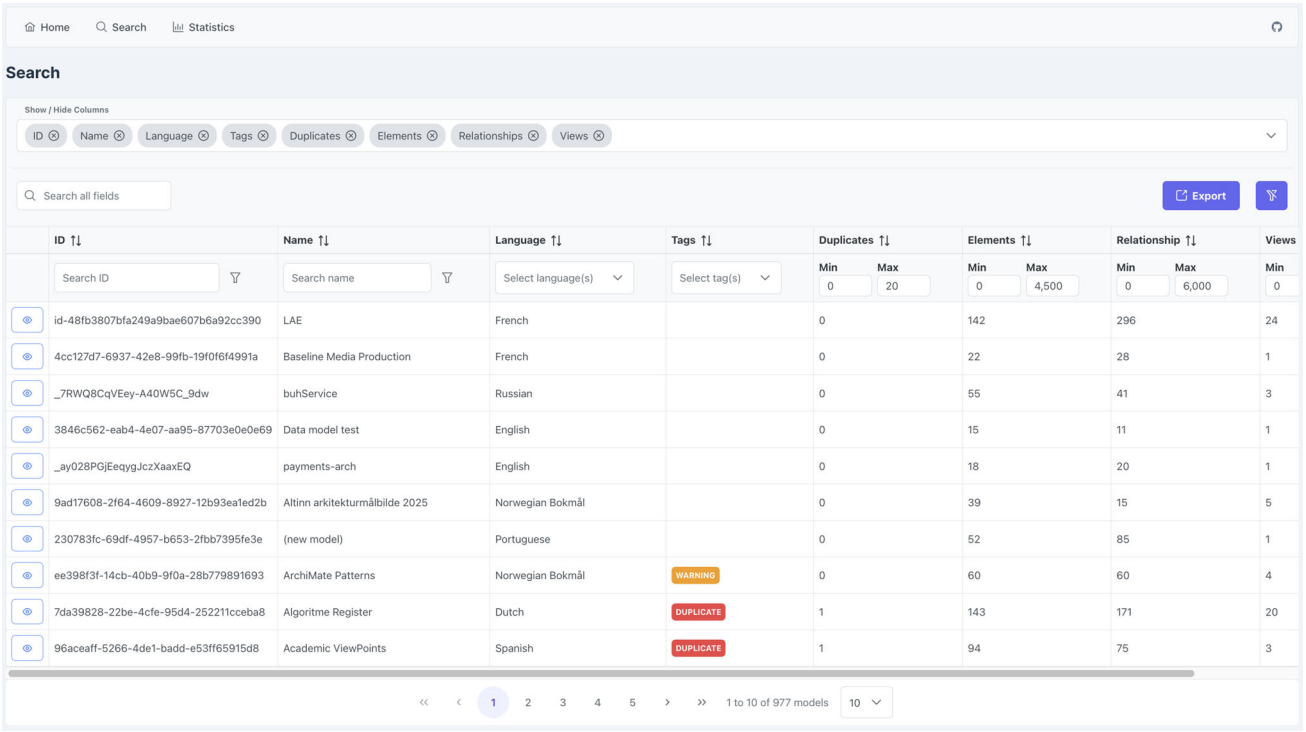


**Fig. 12** Search page of the EA ModelSet website

to access the (meta)data stored in the GitHub repository.[10] Furthermore, the dataset is accessible via a globally unique DOI (10.5281/*zenodo*.8192011) and uses ORCID for author identification. Within the dataset, each model has a unique URI, in the form of https://me-big-tuwien-ac-at.github.io/EAModelSet/model/<id>, where `<id>` represents a tool-generated Universally Unique Identifier (UUID) or a similar type of identifier for the model. The unique identifier allows direct access to each model and guarantees global uniqueness and unambiguous identification.

*F2: "data are described with rich metadata."* The dataset provides comprehensive information about each model, capturing e.g., its name, description, source, license, language, and various other attributes (see Fig. 8). The metadata defined in the JSON schema richly describes the data through additional characteristics.

*F3: "metadata clearly and explicitly include the identifier of the data it describes."* In the JSON representation of the EA model, the metadata explicitly includes the identifier of the data it describes. Each model is associated with a unique URI identifier that incorporates its ID, providing a clear reference to a model. The ID is based on the `archimateId` property which is also included in the metadata and is an auto-generated UUID already present in the collected data, which is reused.

*F4: "(meta)data are registered or indexed in a searchable resource."* The EA ModelSet is hosted in a public GitHub repository, providing e.g., search functionality and version control to locate and access the dataset. The dedicated website and Python library offer additional functionalities, including search and filter capabilities to find models based on certain characteristics (e.g., language, views, number of elements).

## 4.2 Accessibility

*A1: "(meta)data are retrievable by their identifier using a standardized communications protocol."* Metadata and data are retrievable on GitHub, and also using the identifier URI leading to the website, which is accessible using an open, free, and universally implementable communications protocol (*A1.1*), e.g., through HTTP(S) and common web browsers. The protocol thereby enables free access for use but requires an authentication and authorization procedure (i.e., a GitHub account with the required permissions on the repository) for updating the dataset (*A1.2*).

*A2: "metadata are accessible, even when the data are no longer available."* The dataset includes an additional JSON file for each model, providing descriptive metadata for each model. This metadata remains accessible even if the actual data associated with the model are no longer available. We further publish the repository releases on persistent data storage via Zenodo [25] to ensure accessibility even if the GitHub repository becomes unavailable.

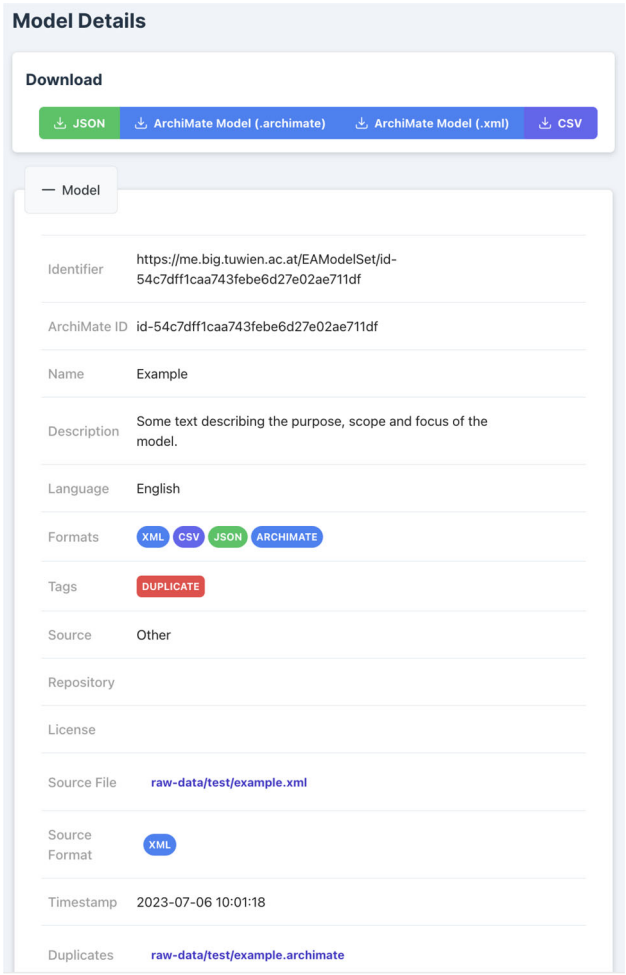


**Fig. 13** Model details page of the EA ModelSet website

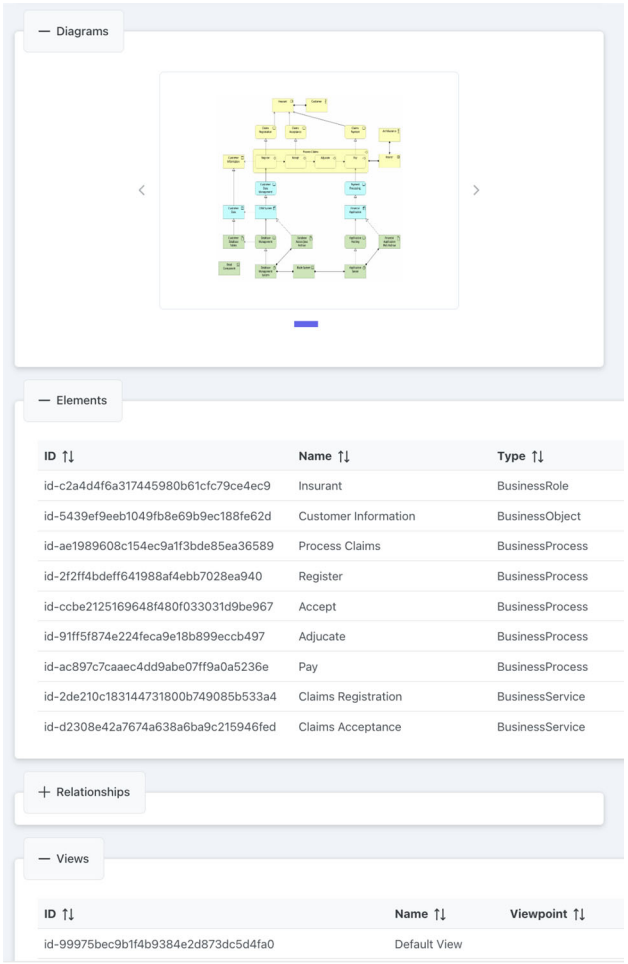


**Fig. 14** Model details page of the EA ModelSet website

## 4.3 Interoperability

*I1: "(meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation."* The metadata and data are stored in JSON files as the main method for knowledge representation. JSON is a widely adopted format for structuring data in a human-readable and machine-readable manner, and the files correspond to a JSON Schema that provides a formal and standardized syntax. Furthermore, we enable additional data formats, including XML and CSV.

*I2: "(meta)data use vocabularies that follow FAIR principles."* The dataset employs a customized (meta)data description, incorporating elements from established FAIR vocabularies. While we have reused vocabularies such as the Data Catalog Vocabulary (DCAT)[13] and Dublin Core Terms (DCT),[14] this is currently limited to translating relevant properties from RDF into JSON schema.[15] Relevant data types are also translated, e.g., dates are formatted according to JSON schema data types (i.e., `date` and `date-time`), and language codes use the two-letter ISO-6391-1 format. Directly linking our schema with existing vocabularies that provide JSON schemas (e.g., DCAT[16]) would enhance semantic interoperability and external referencing but requires careful mapping and validation to ensure equivalence. We recognize this as an important aspect of following FAIR principles and will prioritize this in future extensions of the dataset.

*I3: "(meta)data include qualified references to other (meta)data."* The dataset itself includes `ModelInfo` objects, which are a lightweight representation of models and include an explicit reference to the actual model. Also, the metadata of each model contains explicit references to related models (e.g., duplicates) or internal (e.g., source file) and external resources (e.g., repository).

[13] www.w3.org/TR/vocab-dcat-2/.

[14] www.dublincore.org/specifications/dublin-core/dcmi-terms/.

[15] https://json-schema.org/specification.html.

[16] https://resources.data.gov/schemas/dcat-us/v1.1/schema/catalog.json.

### 4.4 Reusability

*R1: "(meta)data are richly described with a plurality of accurate and relevant attributes."* Each JSON file contains the (meta)data derived from the source model, together with other relevant properties to richly describe a model (e.g., source, timestamp, language, tags). While the dataset already includes many relevant attributes, there is still room for improvement in terms of enriching the metadata. For instance, additional properties such as categories or more descriptive tags could be incorporated to enhance the richness of the metadata, precise filtering, and analysis.

*R1.1: "(meta)data are released with a clear and accessible data usage license."* In building the EA ModelSet, we made an effort to include all available license information from the sources. The majority of models in the dataset have their source repository and corresponding license linked as an entry in the JSON file. The repository and license were automatically retrieved when explicitly provided by the source during data collection (see Sect. 2), and the results were manually re-checked for accuracy. Our approach relies on the assumption that models shared publicly with an attached license are intended for reuse under those terms. However, for models without a clear license, the lack of explicit author awareness raises concerns about permission and usage rights. In cases where no license information was found, we assigned the label `Unspecified` to the license field and the website provides a link to GitHub's documentation on licensing a repository[17] to remind users of possible usage limitations.

*R1.2: "(meta)data are associated with detailed provenance."* The JSON files in each model's directory include properties to present the original source and associated information. The properties provide a level of provenance and include, e.g., source, repository, license, or the parsed source file, allowing traceability to the model's origin. While the current provenance information offers valuable insights, there is potential for more detailed provenance to be included. For example, associating publications or providing diagrams could further enhance the dataset's provenance.

*R1.3: "(meta)data meet domain-relevant community standards."* The EA ModelSet provides models in domain-relevant formats such as ArchiMate XML (two different formats) and CSV. The formats are widely accepted and align with the community's standards, promoting interoperability with existing tools. Furthermore, the newly introduced JSON schema maintains well-established structures and adheres to recognized naming conventions. The introduction of the JSON schema does not add unnecessary complexity but rather provides clarity and consistency to ensure the metadata is understandable within the EA domain. The CSV formats further ease the execution of ML techniques on the EA ModelSet.

[17] https://docs.github.com/en/repositories/managing-your-repositorys-settings-and-features/customizing-your-repository/licensing-a-repository.

## 5 EA ModelSet applications, reflection, and future work

The EA ModelSet dataset provides a rich collection of EA models, unlocking new possibilities for research and practical applications. Researchers can explore the dataset to gain insights into different modeling approaches, applications, and patterns. By analyzing the models within the dataset, researchers can identify best practices and discover common modeling patterns, which could contribute to advancing the field of EA. In this section, we briefly sketch machine learning-based and empirical research applications to the EA ModelSet and discuss directions for future research.

### 5.1 Machine learning applications

The dataset's availability in different formats, including JSON, XML, and CSV, makes it applicable for a variety of ML tasks that extract valuable insights from the data. Some potential applications of ML using the EA ModelSet include *Natural Language Processing* (NLP), *Pattern Detection*, and *Recommender Systems*.

The dataset's textual information, e.g., names, documentation, languages, or tags, can be used to develop NLP models that extract meaningful information from unstructured text. This can support tasks such as *automatic model annotation* [4] and *semantic search* [2, 42].

The EA ModelSet now also allows the application of machine learning-based modeling pattern detection. These modeling patterns, in contrast to existing works where the patterns emerge from the business domain or management principles [13, 23, 49, 61], can be derived from the semantics and the structural aspects of actual models, i.e., ex-post. This approach has been heavily used in other modeling domains with significant contributions, improving the creation of models of high quality [8, 22, 26, 45].

The EA ModelSet dataset can also assist in building recommender systems tailored to EA [51, 55, 58, 72]. By analyzing the repository of EA models, ML algorithms can provide context-based recommendations for specific modeling scenarios. These recommendations can guide enterprise architects by suggesting architectural decisions based on historical data, which can enhance productivity and support informed decision-making [9].

### 5.2 Empirical applications

The EA ModelSet, of course, also allows for empirical research in the practice of enterprise architecture modeling. Similar to machine learning-based pattern detection, also a manual analysis of the EA ModelSet can yield interesting insights. By using the features to filter the models based on, e.g., the ArchiMate layer or the domain, layer-specific and domain-specific patterns can be investigated.

The empirical analysis can also zoom in on the actual use of the language, similarly to, e.g., [18, 45]. Such a quantitative analysis can yield interesting insights into which language concepts are frequently used and which combinations of language concepts frequently appear in models. A qualitative analysis could moreover aim to identify flaws and anti-patterns of enterprise architecture modeling (cf. [34, 59, 65]).

### 5.3 Future work

While the current EA ModelSet dataset is valuable, there are some limitations and areas for future improvement. One current limitation is its ability to only process ArchiMate models in the `*.archimate` and `*.xml` formats. To broaden its applicability in future, we aim to incorporate EA models that (i) conform to other EA modeling languages like 4EM [33], Business Engineering Navigator [71], and which were (ii) created with different EA modeling tools like Dragon1,[18] Atlas[68],[19] Ardoq,[20] Simplified[21] [46, 47], Sparx Systems Enterprise Architect,[22] or LeanIX.[23] Of course, such an extension will require additional research with respect to data harmonization and integration. Even transforming images of models created with other tools to our format is an interesting research challenge (cf. [16]).

An additional current shortcoming we aim to address in future is the fully automated data collection process and to ensure correct record linkage of the source. The current process involving GitHub downloads poses challenges due to authentication, rate limits, and API constraints (e.g., limited file size). We started mitigating these challenges by presenting in this paper our automated pipelines for GenMyModel and GitHub using their current APIs. This allows us to regularly crawl these repositories and update the EA ModelSet with the new models that pass the duplicate and quality checks.

Maintaining data quality and integrity is essential for the EA ModelSet's adoption. Aside from our initial efforts to detect and flag duplicates (based on identifiers and MD5 file hashes), we plan to research and develop more advanced similarity metrics [11, 53] that would help to further clean the data. In terms of data maintenance and publishing, we aim to enhance the dataset's interoperability and operationalizability using an RDF ontology (e.g. [29]).

We also aspire to enrich the classification of models with additional metadata. To support this, we have integrated LabelStudio (see Fig. 5), an open-source data labeling tool, to enable collaborative annotation of models with domain-specific information (e.g., industry sectors) and other relevant metadata. Through structured annotation, we (i) improve the quality of the existing EA ModelSet by identifying incomplete, inaccurate, or duplicate models, ensuring higher consistency, and (ii) enable future extensions by using the labeled data to support processing and automated classification (e.g., through ML) of new models.

Future work also needs to explore the potential to integrate other architecture artifacts into the dataset and link their content to the models. Many artifacts are created in enterprise architecture practice, like architecture documentation, spreadsheets, application portfolios, glossaries, and many more. Semantically analyzing these artifacts and linking them to the model would go a long way toward improved research and reasoning on enterprise architecture practice.

A final direction for future research targets the generalization of our approach. We believe the pipelines for dataset collection, dataset processing, and dataset management & publishing, together with the applications for the labeling of the models and the deployment of the entire dataset, yield potential to be used also in other domains for arbitrary modeling languages [30]. Future work thus needs to further generalize the scripts and the implementation of the applications for being re-used efficiently.

We invite and hope to actively engage the modeling research community for all future considerations. The EA ModelSet is open source, and we plan to realize functionalities that enable efficient contributions from the community, especially with respect to curating the existing dataset and extending the dataset with new models. For example, we aim to offer a model upload service to the EA ModelSet webpage. Users could then simply upload model files in the valid formats, our system would then automatically create a new pull request to the EA ModelSet Github repository. The community could then investigate and discuss the proposed additions, and, in case the addition is meaningful and not a duplicate, the new models could be merged into the dataset. We aim to explore means of automating parts of this process and enrich the process with notification messages sent to registered persons, notifying them about potential additions. With continuous community engagement and improvement

[18] https://www.dragon1.com/resources/enterprise-architecture.

[19] https://linkconsulting.com/what-we-do/products/atlas/.

[20] https://www.ardoq.com/.

[21] https://simplified.engineering/.

[22] https://sparxsystems.com/products/ea/.

[23] https://www.leanix.net/de/enterprise-architecture.

efforts, we aspire to make the EA ModelSet a valuable and comprehensive resource for researchers in the enterprise modeling domain.

## 6 Related ModelSets

FAIR datasets have garnered significant attention in various research domains. In the field of **conceptual and enterprise modeling**, researchers have focused on creating FAIR datasets that encompass various domain-specific models, such as data models, ontology models [6], and domain models. These datasets aim to enhance the accessibility and reusability of conceptual models for research and practical applications and to enable insights into the actual use of modeling languages. Additionally, efforts have been made to standardize metadata annotation and representation to improve the findability and interoperability of the datasets [7, 64].

In **software engineering and software modeling** research, the development of FAIR datasets has been crucial for advancing the state of software development, testing, and maintenance [27, 30]. Researchers have built datasets that comprise software architecture models, UML diagrams, and source code representations [36, 38, 39, 52, 54]. These datasets enable software engineers to leverage ML, data-driven, and empirical techniques to automate and/or improve software development tasks.

Within the **process modeling** community, there have been efforts to curate datasets containing various types of process models [17, 28, 62, 66]. The sub-discipline of **process mining** is also heavily engaged in the creation and use of publicly available datasets (see [20]). These datasets facilitate the empirical analysis of business process management and the evaluation and comparison of process mining algorithms and tools. Notably, also tool vendors started recently to develop ModelSets, like the SAP Signavio Academic Models dataset (SAP-SAM) [66]. The SAP-SAM dataset [66] contains 1,021,471 models in different modeling languages, mainly BPMN process models, but also models conforming to other modeling languages like UML and Petri Net. Moreover, the SAP-SAM dataset contains 10,956 ArchiMate models. All models were created over multiple years through the http://academic.signavio.com platform by researchers, teachers, and students.

## 7 Conclusion

The scarcity of models in adequate quantity and quality is a huge barrier to conducting cutting-edge data-driven and empirical research in modeling. In this paper, we proposed the EA ModelSet, a FAIR dataset of enterprise architecture models in the ArchiMate language that allows these kinds of research in enterprise architecture.

The EA ModelSet is a curated dataset that currently contains 977 ArchiMate models. We believe the dataset has the potential to spark research at the intersection of machine learning and enterprise modeling. Moreover, it enables a deep dive into empirical research in enterprise architecture modeling. We invite the modeling research community to help further curate, maintain, and extend the dataset, and also tool vendors to explore their interest in sharing some of their models.

To improve the quality of the EA ModelSet, we propose a graphical interface, enabled through LabelStudio, that allows efficient exploration and labeling of the models in our dataset, a crucial step for improving the quality of the models and, therefore, the potential insights gained from research using our dataset. The next release of the EA ModelSet will thus focus on high-quality, semantically labeled models.

We hope that the EA ModelSet becomes a valuable asset and helps EA researchers derive theoretical and practical insights into enterprise architecture modeling and management. Aside from the automated solutions for collecting, processing, and managing the EA ModelSet, a community effort is required to ensure its further maintenance and development. In an initial effort, we currently use the developed LabelStudio to semantically classify the models and filter out meaningless models.

**Acknowledgements** This work has been partially funded through the Erasmus+ KA220-HED project "Digital Platform Enterprise" (Project No. 2021-1-RO01-KA220-HED-000027576) and the Vienna Science and Technology Fund (WWTF) (10.47379/VRG18013).

**Funding** Open access funding provided by TU Wien (TUW).



## References

1. Abdelnabi, E.A., Maatuk, A.M., Hagal, M.: Generating uml class diagram from natural language requirements: a survey of approaches and techniques. In: 2021 IEEE 1st International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering MI-STA, pp. 288–293. IEEE (2021)

2. Ali, S.J.: Knowledge graph-based conceptual models search. In: Link, S., Reinhartz-Berger, I., Zdravkovic, J., et al. (eds.) Proceedings of the ER Forum and PhD Symposium 2022 Co-located with 41st International Conference on Conceptual Modeling (ER 2022), Virtual Event, Hyderabad, India, October 17, 2022, CEUR Workshop Proceedings, vol. 3211. CEUR-WS.org, https://ceur-ws.org/Vol-3211/CR_100.pdf (2022)
3. Ali, S.J., Bork, D.: A graph language modeling framework for the ontological enrichment of conceptual models. In: Guizzardi, G., Santoro, F.M., Mouratidis, H., et al. (eds.) Advanced Information Systems Engineering—36th International Conference, CAiSE 2024, Limassol, Cyprus, June 3–7, 2024, Proceedings, Lecture Notes in Computer Science, vol. 14663, pp. 107–123. Springer. https://doi.org/10.1007/978-3-031-61057-8_7 (2024)
4. Ali, S.J., Guizzardi, G., Bork, D.: Enabling representation learning in ontology-driven conceptual modeling using graph neural networks. In: Indulska, M., Reinhartz-Berger, I., Cetina, C., et al. (eds.) Advanced Information Systems Engineering—35th International Conference, CAiSE 2023, Zaragoza, Spain, June 12–16, 2023, Proceedings, Lecture Notes in Computer Science, vol. 13901, pp. 278–294. Springer. https://doi.org/10.1007/978-3-031-34560-9_17 (2023)
5. Almonte, L., Guerra, E., Cantador, I., et al.: Recommender systems in model-driven engineering. Softw. Syst. Model. **21**(1), 249–280 (2022). https://doi.org/10.1007/S10270-021-00905-X
6. Barcelos, P.P.F., Sales, T.P., Fumagalli, M., et al.: A FAIR model catalog for ontology-driven conceptual modeling research. In: 41st International Conference on Conceptual Modeling, ER 2022, pp. 3–17. Springer. https://doi.org/10.1007/978-3-031-17995-2_1 (2022)
7. Bernabé, C.H., Sales, T.P., Schultes, E., et al.: A goal-oriented method for fairification planning. In: Fonseca, C.M., Borbinha, J., Guizzardi, G., et al. (eds.) Companion Proceedings of the 42nd International Conference on Conceptual Modeling: ER Forum, 7th SCME, Project Exhibitions, Posters and Demos, and Doctoral Consortium co-located with ER 2023, Lisbon, Portugal, November 06–09, 2023, CEUR Workshop Proceedings, vol. 3618. CEUR-WS.org, https://ceur-ws.org/Vol-3618/forum_paper_7.pdf (2023)
8. Blaha, M.: Patterns of Data Modeling, vol. 1. CRC Press, Boca Raton (2010)
9. Bork, D., Ali, S.J., Dinev, G.M.: Ai-enhanced hybrid decision management. Bus. Inf. Syst. Eng. **65**(2), 179–199 (2023). https://doi.org/10.1007/s12599-023-00790-2
10. Bork, D., Ali, S.J., Roelens, B.: Conceptual modeling and artificial intelligence: a systematic mapping study. CoRR arXiv:2303.06758. https://doi.org/10.48550/arXiv.2303.06758 (2023)
11. Borozanov, V., Hacks, S., Silva, N.: Using machine learning techniques for evaluating the similarity of enterprise architecture models—technical paper. In: Advanced Information Systems Engineering—31st International Conference, pp. 563–578 (2019)
12. Bucchiarone, A., Cabot, J., Paige, R.F., et al.: Grand challenges in model-driven engineering: an analysis of the state of the research. Softw. Syst. Model. **19**(1), 5–13 (2020). https://doi.org/10.1007/S10270-019-00773-6
13. Buckl, S., Ernst, A.M., Matthes, F. et al: Using enterprise architecture management patterns to complement togaf. In: 2009 IEEE International Enterprise Distributed Object Computing Conference, pp. 34–41. https://doi.org/10.1109/EDOC.2009.30 (2009)
14. Burgueño, L., Clarisó, R., Gérard, S., et al.: An nlp-based architecture for the autocompletion of partial domain models. In: Rosa, M.L., Sadiq S.W., Teniente, E. (eds.) Advanced Information Systems Engineering—33rd International Conference, CAiSE 2021, Melbourne, VIC, Australia, June 28–July 2, 2021, Proceedings, Lecture Notes in Computer Science, vol. 12751, pp. 91–106. Springer. https://doi.org/10.1007/978-3-030-79382-1_6 (2021)
15. Burgueño, L., Cabot, J., Li, S., et al.: A generic LSTM neural network architecture to infer heterogeneous model transformations. Softw. Syst. Model. **21**(1), 139–156 (2022). https://doi.org/10.1007/S10270-021-00893-Y
16. Chen, F., Zhang, L., Lian, X., et al.: Automatically recognizing the semantic elements from UML class diagram images. J. Syst. Softw. **193**, 111431 (2022). https://doi.org/10.1016/J.JSS.2022.111431
17. Corradini, F., Fornari, F., Polini, A., et al.: Reprository: a repository platform for sharing business process models and logs. In: Proceedings of the 1st Italian Forum on Business Process Management. CEUR-WS.org, pp. 13–18 (2021)
18. Corradini, F., Fornari, F., Polini, A., et al.: A formal approach for the analysis of BPMN collaboration models. J. Syst. Softw. **180**, 111007 (2021). https://doi.org/10.1016/J.JSS.2021.111007
19. Dirix, M., Muller, A., Aranega, V.: GenMyModel: an online UML case tool. In: ECOOP 2013, https://hal.science/hal-01251417 (2013)
20. Dumas, M., Rosa, M.L., Mendling, J., et al.: Fundamentals of bpm: model collections. http://fundamentals-of-bpm.org/process-model-collections/, Last Accessed: 24 Jylu 2023 (2024)
21. Eisenberg, M., Sahay, A., Ruscio, D.D., et al.: Multi-objective model transformation chain exploration with momot. Inf. Softw. Technol. **174**, 107500 (2024). https://doi.org/10.1016/J.INFSOF.2024.107500
22. Fumagalli, M., Sales, T.P., Guizzardi, G.: Pattern discovery in conceptual models using frequent itemset mining. In: Ralyté, J., Chakravarthy, S., Mohani,a M.K., et al. (eds.) Conceptual Modeling - 41st International Conference, ER 2022, Hyderabad, India, October 17–20, 2022, Proceedings, Lecture Notes in Computer Science, vol. 13607. Springer, pp. 52–62. https://doi.org/10.1007/978-3-031-17995-2_4 (2022)
23. García-Escallón, R.R., Aldea, A.: On enterprise architecture patterns: a systematic literature review. In: Filipe, J., Smialek, M., Brodsky, A., et al. (eds.) Proceedings of the 22nd International Conference on Enterprise Information Systems, ICEIS 2020, Prague, Czech Republic, May 5–7, 2020, vol. 2. SCITEPRESS, pp. 666–678. https://doi.org/10.5220/0009392306660678 (2020)
24. Glaser, P., Sallinger, E., Bork, D.: EA modelset—A FAIR dataset for machine learning in enterprise modeling. In: Almeida, J.P.A., Kaczmarek-Heß, M., Koschmider, A., et al. (eds.) The Practice of Enterprise Modeling—16th IFIP Working Conference, PoEM 2023, Vienna, Austria, November 28–December 1, 2023, Proceedings, Lecture Notes in Business Information Processing, vol. 497, pp. 19–36. Springer. https://doi.org/10.1007/978-3-031-48583-1_2 (2023a)
25. Glaser, P.L., Sallinger, E., Bork, D.: EA ModelSet. (2023). https://doi.org/10.5281/zenodo.8192011
26. Gomaa, H.: Software Modeling and Design: UML, Use Cases, Patterns, and Software Architectures. Cambridge University Press, London (2011)
27. Hebig, R., Ho-Quang, T., Chaudron, M.R.V., et al.: The quest for open source projects that use UML: mining github. In: Baudry, B., Combemale, B. (eds.) Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems, Saint-Malo, France, October 2–7, 2016, pp. 173–183. ACM, http://dl.acm.org/citation.cfm?id=2976778 (2016)
28. Heinze, T.S., Stefanko, V., Amme, W.: BPMN in the wild: BPMN on github.com. In: Manner J, Haarmann, S., Kolb, S., et al (eds.) Proceedings of the 12th ZEUS Workshop on Services and their Composition, Potsdam, Germany, February 20–21, 2020, CEUR Workshop Proceedings, vol. 2575, pp. 26–29. CEUR-WS.org, https://ceur-ws.org/Vol-2575/paper5.pdf (2020)
29. Hinkelmann, K., Laurenzi, E., Martin, A., et al.: Archimeo: a standardized enterprise ontology based on the archimate conceptual model. In: Proceedings of the 8th International Conference on Model-Driven Engineering and Software Development, MOD-

ELSWARD 2020. SCITEPRESS, pp. 417–424. https://doi.org/10.5220/0009000204170424 (2020)
30. Ho-Quang, T., Chaudron, M.R.V., Robles, G., et al.: Towards an infrastructure for empirical research into software architecture: challenges and directions. In: Medvidovic, N., Mirakhorli, M., Malek, S., et al. (eds.) Proceedings of the 2nd International Workshop on Establishing a Community-Wide Infrastructure for Architecture-Based Software Engineering, ECASE@ICSE 2019, May 27, 2019, Montreal, Quebec, Canada. IEEE/ACM, pp. 34–41. https://doi.org/10.1109/ECASE.2019.00014 (2019)
31. Iovino, L., Barriga, A., Rutle, A., et al.: Model repair with quality-based reinforcement learning. J. Object Technol. **19**(2), 1–21 (2020). https://doi.org/10.5381/JOT.2020.19.2.A17
32. Kampik, T., Warmuth, C., Rebmann, A., et al.: Large process models: a vision for business process management in the age of generative ai. KI-Künstliche Intelligenz, pp. 1–15 (2024)
33. Lantow, B., Sandkuhl, K., Stirna, J.: Enterprise modeling with 4em: perspectives and method. In: Karagiannis, D., Lee, M., Hinkelmann, K., et al. (eds.) Domain-Specific Conceptual Modeling—Concepts, Methods and ADOxx Tools, pp. 95–120. Springer. https://doi.org/10.1007/978-3-030-93547-4_5 (2022)
34. Leopold, H., Mendling, J., Günther, O.: Learning from quality issues of BPMN models from industry. IEEE Softw. **33**(4), 26–33 (2016). https://doi.org/10.1109/MS.2015.81
35. Lopes, R., Araújo, J., da Silveira, D.S., et al.: A systematic approach to derive conceptual models from BPMN models. In: Shishkov, B. (ed.) Business Modeling and Software Design—14th International Symposium, BMSD 2024, Luxembourg City, Luxembourg, July 1–3, 2024, Proceedings, Lecture Notes in Business Information Processing, vol. 523, pp. 83–96. Springer. https://doi.org/10.1007/978-3-031-64073-5_6 (2024)
36. López, J.A.H., Cuadrado, J.S.: An efficient and scalable search engine for models. Softw. Syst. Model. **21**(5), 1715–1737 (2022). https://doi.org/10.1007/s10270-021-00960-4
37. López, J.A.H., Cuadrado, J.S.: Generating structurally realistic models with deep autoregressive networks. IEEE Trans. Softw. Eng. **49**(4), 2661–2676 (2023). https://doi.org/10.1109/TSE.2022.3228630
38. López, J.A.H., Izquierdo, J.L.C., Cuadrado, J.S.: Modelset: a dataset for machine learning in model-driven engineering. Softw. Syst. Model. **21**(3), 967–986 (2022). https://doi.org/10.1007/s10270-021-00929-3
39. López, J.A.H., Izquierdo, J.L.C., Cuadrado, J.S.: Using the modelset dataset to support machine learning in model-driven engineering. In: Kühn, T., Sousa, V. (eds.) 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings, MODELS 2022, pp. 66–70. ACM. https://doi.org/10.1145/3550356.3559096 (2022)
40. López, J.A.H., Rubei, R., Cuadrado, J.S., et al.: Machine learning methods for model classification: a comparative study. In: Syriani, E., Sahraoui, H.A., Bencomo, N., et al. (eds.) Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems, MODELS 2022, Montreal, Quebec, Canada, October 23–28, 2022, pp. 165–175. ACM. https://doi.org/10.1145/3550355.3552461 (2022c)
41. López, J.A.H., Izquierdo, J.L.C., Cuadrado, J.S.: Modelset: a labelled dataset of software models for machine learning. Sci. Comput. Program. **231**, 103009 (2024). https://doi.org/10.1016/J.SCICO.2023.103009
42. Lucrédio, D., de Mattos Fortes, R.P., Whittle, J.: MOOGLE: a metamodel-based model search engine. Softw. Syst. Model. **11**(2), 183–208 (2012). https://doi.org/10.1007/S10270-010-0167-7
43. Marcén, A.C., Iglesias, A., Lapeña, R., et al.: A systematic literature review of model-driven engineering using machine learning. IEEE Trans. Softw. Eng. **50**(9), 2269–2293 (2024). https://doi.org/10.1109/TSE.2024.3430514
44. Michael, J., Bork, D., Wimmer, M., et al.: Quo vadis modeling? Softw. Syst. Model. **23**(1), 7–28 (2024). https://doi.org/10.1007/S10270-023-01128-Y
45. Muehlen, M.Z., Recker, J.: How much language is enough? Theoretical and practical use of the business process modeling notation. In: Seminal Contributions to Information Systems Engineering: 25 Years of CAiSE, pp. 429–443 (2013)
46. Mulder, M.A.T., Mulder, R., Bodnar, F., et al.: The simplified platform, an overview. In: Michael, J., Pfeiffer, J., Wortmann, A. (eds.) Modellierung 2022—Workshop Proceedings, Hamburg, Germany, June 27–July 1, 2022. Gesellschaft für Informatik e.V., pp. 223–234. https://doi.org/10.18420/MODELLIERUNG2022WS-031 (2022)
47. Mulder, M.A.T., Proper, H.A., Bodnar, F., et al.: Simplified enterprise modelling platform architecture. In: Clark, T., Zschaler, S., Barn, B., et al (eds.) Proceedings of the Forum at Practice of Enterprise Modeling 2022 (PoEM-Forum 2022) co-located with PoEM 2022, London, UK, November 23–25, 2022, CEUR Workshop Proceedings, vol. 3327. CEUR-WS.org, pp. 16–30. http://ceur-ws.org/Vol-3327/paper03.pdf (2022)
48. Mussbacher, G., Combemale, B., Kienzle, J., et al.: Opportunities in intelligent modeling assistance. Softw. Syst. Model. **19**(5), 1045–1053 (2020). https://doi.org/10.1007/S10270-020-00814-5
49. Perroud, T., Inversini, R.: Enterprise architecture patterns: practical solutions for recurring IT-architecture problems. Springer (2013). https://doi.org/10.1007/978-3-642-37561-3
50. Pezoa, F., Reutter, J.L., Suárez, F., et al.: Foundations of JSON schema. In: 25th International Conference on World Wide Web, WWW 2016, pp. 263–273 . ACM (2016)
51. Raavikanti, S., Hacks, S., Katsikeas, S.: A recommender plug-in for enterprise architecture models. In: 25th International Conference on Enterprise Information Systems, ICEIS 2023. SCITEPRESS, pp. 474–480, https://doi.org/10.5220/0011709000003467 (2023)
52. Rahman, M.I., Panichella, S., Taibi, D.: A curated dataset of microservices-based systems. CoRR arXiv:1909.03249 (2019)
53. Ralf, L., Martin, L.: Beobachtungen und einsichten zu repositorys von bpmn-modellen. In: Modellierung 2024, Gesellschaft für Informatik eV, pp. 157–173 (2024)
54. Robles, G., Ho-Quang, T., Hebig, R., et al.: An extensive dataset of UML models in github. In: 14th International Conference on Mining Software Repositories, MSR 2017, pp. 519–522. IEEE Computer Society. https://doi.org/10.1109/MSR.2017.48 (2017)
55. Rocco, J.D., Ruscio, D.D., Sipio, C.D., et al.: Memorec: a recommender system for assisting modelers in specifying metamodels. Softw. Syst. Model. **22**(1), 203–223 (2023). https://doi.org/10.1007/S10270-022-00994-2
56. Romero, J.R., Medina-Bulo, I., Chicano, F. (eds.): Optimising the Software Development Process with Artificial Intelligence. Natural Computing Series. Springer, Berlin (2023). https://doi.org/10.1007/978-981-19-9948-2
57. Rädler, S., Berardinelli, L., Winter, K., et al.: Bridging MMDE and AI: a systematic review of domain-specific languages and model-driven practices in AI software systems engineering. Softw. Syst. Model. (2024). https://doi.org/10.1007/s10270-024-01211-y
58. Saini, R., Mussbacher, G., Guo, J.L.C., et al.: Domobot: a bot for automated and interactive domain modelling. In: Guerra, E., Iovino, L. (eds.) MODELS'20: ACM/IEEE 23rd International Conference on Model Driven Engineering Languages and Systems, Virtual Event, Canada, 18–23 October, 2020, Companion Proceedings, pp. 45:1–45. ACM. https://doi.org/10.1145/3417990.3421385 (2020)
59. Salentin, J., Hacks, S.: Towards a catalog of enterprise architecture smells. In: Gronau, N., Heine, M., Krasnova, H., et al. (eds.) Entwicklungen, Chancen und Herausforderungen der Digitalisierung: Proceedings der 15. Internationalen Tagung Wirtschaftsinformatik, WI 2020, Potsdam, Germany, March 9–11,

2020. Community Tracks, pp. 276–290. GITO Verlag. https://doi.org/10.30844/WI_2020_Y1-SALENTIN (2020)
60. Sales, T.P., Barcelos, P.P.F., Fonseca, C.M., et al.: A FAIR catalog of ontology-driven conceptual models. Data Knowl. Eng. **147**, 102210 (2023). https://doi.org/10.1016/J.DATAK.2023.102210
61. Sasa, A., Krisper, M.: Enterprise architecture patterns for business process support analysis. J. Syst. Softw. **84**(9), 1480–1506 (2011). https://doi.org/10.1016/J.JSS.2011.02.043
62. Schäfer, B., van der Aa, H., Leopold, H., et al.: Sketch2bpmn: automatic recognition of hand-drawn BPMN models. In: 33rd International Conference Advanced Information Systems Engineering. Springer, pp. 344–360 (2021)
63. Shilov, N., Othman, W., Fellmann, M., et al.: Machine learning for enterprise modeling assistance: an investigation of the potential and proof of concept. Softw. Syst. Model. **22**(2), 619–646 (2023). https://doi.org/10.1007/s10270-022-01077-y
64. da Silva Santos, L.O.B., Sales, T.P., Fonseca, C.M., et al.: Towards a conceptual model for the FAIR digital object framework. CoRR arXiv:2302.11894 https://doi.org/10.48550/arXiv.2302.11894 (2023)
65. Smajevic, M., Hacks, S., Bork, D.: Using knowledge graphs to detect enterprise architecture smells. In: Serral, E., Stirna, J., Ralyté J., et al (eds.) The Practice of Enterprise Modeling—14th IFIP WG 8.1 Working Conference, PoEM 2021, Riga, Latvia, November 24–26, 2021, Proceedings, Lecture Notes in Business Information Processing, vol. 432, pp. 48–63. Springer. https://doi.org/10.1007/978-3-030-91279-6_4 (2021)
66. Sola, D., Warmuth, C., Schäfer, B., et al.: SAP signavio academic models: A large process model dataset. In: Montali, M., Senderovich, A., Weidlich, M. (eds.) Process Mining Workshops—ICPM 2022 International Workshops, 2022, Revised Selected Papers, Lecture Notes in Business Information Processing, vol. 468, pp. 453–465. Springer. https://doi.org/10.1007/978-3-031-27815-0_33 (2022)
67. Sollis, E., Mosaku, A., Abid, A., et al.: The nhgri-ebi gwas catalog: knowledgebase and deposition resource. Nucleic Acids Res. **51**(D1), D977–D985 (2023)
68. Sousa, P., Vasconcelos, A.: Enterprise Architecture and Cartography—From Practice to Theory. Springer, From Representation to Design (2022). https://doi.org/10.1007/978-3-030-96264-7
69. Tanhua, T., Pouliquen, S., Hausman, J., et al.: Ocean fair data services. Front. Mar. Sci. **6**, 440 (2019)
70. Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., et al.: The FAIR Guiding principles for scientific data management and stewardship. Sci. Data **3**(1), 160018 (2016). https://doi.org/10.1038/sdata.2016.18
71. Winter, R.: Business Engineering Navigator: Gestaltung und Analyse von Geschäftslösungen" Business-to-IT". Springer, Berlin (2010)
72. Zhi, Q., Zhou, Z.: Empirically modeling enterprise architecture using archimate. Comput. Syst. Sci. Eng. **40**(1), 357–374 (2022). https://doi.org/10.32604/csse.2022.018759



**Philipp-Lorenz Glaser** is a Master's student in Software Engineering and Internet Computing at the Faculty of Informatics, TU Wien. He is also a Student Assistant in Research and Administration at the Business Informatics Group, TU Wien. His research interests combine Conceptual Modeling with Artificial Intelligence, particularly Knowledge Graphs, -Machine Learning, and Natural Language Processing. In addition, he is interested in Modeling Tool Development and Software Engineering. For more information, you can contact the author at philipp-lorenz.glaser@tuwien.ac.at.

**Emanuel Sallinger** is Professor of Scalable Artificial Intelligence at the Faculty of Informatics, Institute of Logic and Computation, Databases and AI Group at TU Wien. His research interests comprise Knowledge Graphs, including data management and symbolic and subsymbolic AI methodologies associated with them. This incorporates logic-based reasoners, Knowledge Graph Embeddings, Graph Neural Networks and Large Language Models, as well as their combinations in the form of neuro-symbolic AI. For more information, you can contact the author at emanuel.sallinger@tuwien.ac.at or visit https://kg.dbai.tuwien.ac.at/.

**Dominik Bork** is an Associate Professor of Business Systems Engineering at the Faculty of Informatics, Institute of Information Systems Engineering, Business Informatics Group at TU Wien. His research interests comprise conceptual modeling, model-driven engineering, and modeling tool development. A primary focus of ongoing research is on the mutual benefits of conceptual modeling and artificial intelligence. For more information, you can contact the author at dominik.bork@tuwien.ac.at or visit https://www.model-engineering.info/.