

Towards Graph-based Analysis of Enterprise Architecture Models

Muhammed Smajevic and Dominik Bork

In:

*ER 2021 – 40th International Conference on Conceptual Modeling,
18-21 October 2021 St. John's, NL, Canada , pp. 199 - 209.*


©2021 by Springer.

Final version available via DOI:

https://doi.org/10.1007/978-3-030-89022-3_17

www.model-engineering.info

Towards Graph-based Analysis of Enterprise Architecture Models

Muhammed Smajevic and Dominik Bork ^[0000–0001–8259–2297]

TU Wien, Business Informatics Group, Vienna, Austria
`e11742556@student.tuwien.ac.at`, `dominik.bork@tuwien.ac.at`

Abstract. A core strength of enterprise architecture (EA) models is their holistic and integrative nature. With ArchiMate, a de-facto industry standard for modeling EAs is available and widely adopted. However, with the growing complexity of enterprise operations and IT infrastructures, EA models grow in complexity. Research showed that ArchiMate as a language and the supporting EA tools lack advanced visualization and analysis functionality. This paper proposes a generic and extensible framework for transforming EA models into graph structures to enable the automated analysis of even huge EA models. We show how enterprise architects can benefit from the vast number of graph metrics during decision-making. We also describe the implementation of the extensible Graph-based Enterprise Architecture Analysis (eGEAA) Cloud platform that supports the framework. The evaluation of our approach and platform confirms feasibility and interoperability with third-party tools.

Keywords: Enterprise Architecture · Model transformation · ArchiMate · Graph Theory · Analysis.

1 Introduction

Enterprises are complex systems composed of different domains that affect each other. Describing enterprises holistically is helpful in many aspects like business and IT alignment. Enterprise Architecture (EA)s represent the high-level view of different enterprise domains and the connections between them. However, holistic EA models grow in size, thereby hampering manual analysis by enterprise architects [11]. To mitigate this problem ArchiMate, the de-facto industry standard for EA modeling defines a viewing mechanism where only selected aspects are considered in one model (i.e., view) and most EA tools provide a repository of EA entities to ease reuse. Still, more advanced support in addressing the inherent complexity of EA models is required. Although EA modeling is widely adopted, the analysis of EA models is surprisingly underrepresented in research so far [3, 14]. Only recently, the first proposals emerged aiming to equip EA modeling by advanced visualization and analysis techniques [4, 12, 20, 16, 25].

Automated EA model analysis can mitigate some of the discussed problems by scaling well and by providing interactive analysis means that extend static ones [17]. In this exploratory and applied research, we present a *generic* and

extensible framework for the transformation of EA models into graph structures that addresses the challenges mentioned at the outset. Thus, we aim to “*extract valuable information from the EA component’s relationships to support the analytic process.*” [27, p. 1] Once the EA model is transformed into a graph, enterprise architects can apply the plethora of existing algorithms/metrics (e.g., centrality and community detection) and tools to assist in decision making [15]. Our approach is generic in the sense of being realized on the meta-metamodel level (i.e., independent of a particular modeling tool platform) and extensible to ease its adoption for other conceptual modeling languages. We report on the prototypical implementation of our approach and its evaluation in a case study.

With the paper at hand, we aim to address the following research objectives:

RO-1: Development of a generic and extensible framework for the transformation of conceptual models into graph structures.

RO-2: Investigate the benefits of supporting enterprise architects by graph-based analysis.

RO-3: Implementation of an EA graph analysis platform.

This paper unfolds as follows. Foundations of Enterprise Architecture Modeling (EAM) and graph analysis are defined in Section 2. Section 3 then introduces our generic framework for transforming conceptual models into graphs. The prototypical implementation of our framework is presented in Section 4 and evaluated in Section 5. Eventually, concluding remarks are given in Section 6.

2 Foundations

2.1 Enterprise Architecture Management

Enterprise Architecture Management (EAM) is broadly defined as “*management practice that establishes, maintains and uses a coherent set of guidelines, architecture principles and governance regimes that provide direction for and practical help with the design and the development of an enterprise’s architecture in order to achieve its vision and strategy*” [1]. ArchiMate [18, 21] is nowadays one of the most used EA languages in practice. ArchiMate depicts an enterprise in the ArchiMate Framework where the core entities of an enterprise are categorized along two dimensions (layers and aspects). A strength of ArchiMate is its possibility to cover relevant aspects of an enterprise in a holistic, multi-layered, and integrated manner. A shortcoming of ArchiMate is though its limited semantic specificity [23] and the limited processing of the information specified in the models [7]. Consequently, proprietary EAM tools often come with additional functionality realized on top of ArchiMate models to enable model value.

2.2 Graph Analysis

A graph connects two or more entities where entities can be anything like human beings, machines, animals, and variables in literature [22]. In Graph Theory, these entities are considered as *Nodes* while the relationships are considered as

Edges and their connections form a graph. Graphs can be classified into *directed*, *undirected*, and *mixed* graphs [22]. Analysis of such graphs is a wide research area with many applications in different domains. In the following, we will briefly introduce the two analysis techniques relevant for this paper: *quantitative graph analysis* and *visual analysis*.

Quantitative Graph Theory is defined as a measurement approach to quantify structural information of graphs [10]. In Graph Theory, quantitative measures describe the structural characteristics of graphs instead of characterizing graphs only descriptively [9]. Two well known examples of such measures are *PageRank* and *Betweenness* (cf. [9]). The former algorithm applies the well-known PageRank algorithm for websites to graphs, whereas the latter treats graphs as social networks, aiming to, e.g., identify communities and clans. Many tools exist for graph visualization, all of them providing a rich set of powerful graph layout algorithms (see [13] for an overview). The power of the tools comes with the customizability of the algorithms, e.g., different sizes and colors of nodes and edges based on graph properties or quantitative metrics.

Different formats for storing a graph structure exist. One such format is GraphML. It is XML-based and supports attributes, nodes, edges, and hierarchical ordering of graphs by means of sub-graphs [19].

3 Transforming Enterprise Architectures into Graphs

In order to analyze the EA in a graph-based manner we propose the framework visualized in Fig. 1. In contrast to the related works, this framework is generic and extensible in two ways: First, it builds upon the conceptual models produced by state-of-the-art metamodeling platforms (Ecore and ADOxx), which enables the transformation of any conceptual model created with these platforms into a graph. Second, we transform the conceptual model into GraphML that enables the use of any graph analysis tool provided that it supports the standardized GraphML format. Consequently, our framework bridges powerful modeling (and metamodeling platforms) on the one side with graph analysis tools on the other instead of implementing a solution for an individual modeling language or tool.

As highlighted in Fig. 1, our framework starts on the meta²-level. The reason behind this is the idea to define a generic transformation on the meta-meta level which can then be further customized at the modeling language level to the specifics of a modeling language. Eventually, the produced GraphML output forms the input for analysis by powerful tools like Neo4j and Gephi. The thick border around some aspects of Fig. 1 highlights the subset of the framework we will discuss in the following. Consequently, we will describe the transformation of Ecore models into GraphML graphs. This focus is a consequence of the limited space available, however, we realized, implemented, and tested the transformation also for ADOxx-based models which further increases the applicability of our generic framework. We evaluated our platform with both, Ecore- and ADOxx-based EA models (see Section 5).

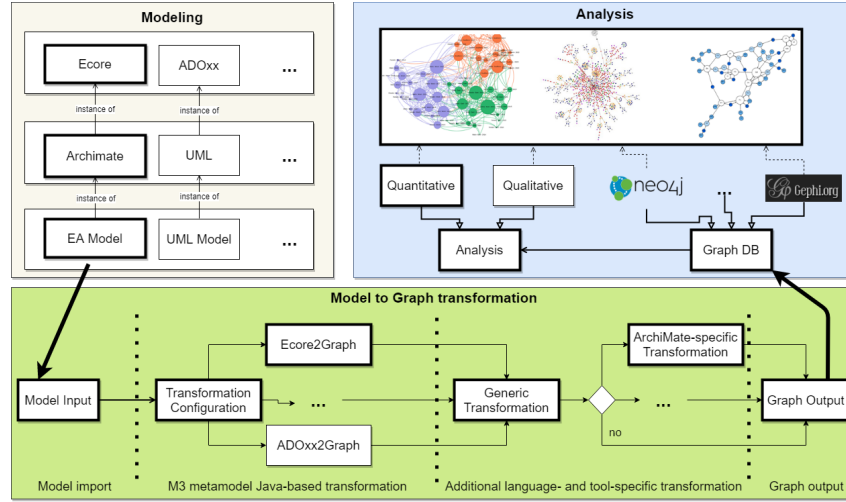


Fig. 1: Generic framework for enterprise architecture models into graphs.

3.1 Generic Model to Graph Transformation

The transformation of EA models created with the Archi modeling tool is decomposed into two parts. First, the generic transformation from Ecore to GraphML is discussed (as Archi is based on Ecore). Thereafter, the specific rules for transforming Archi-based ArchiMate models into GraphML are presented.

The generic rules for transforming Ecore models into GraphML are visualized in Fig. 2 by means of mapping the concepts of the Ecore metamodel to the concepts of the GraphML metamodel. The initial *EPackage* is mapped to *GraphML* while the others are then mapped to *Graph*. Each *EClass* is mapped to *Node* while each *EReference* is mapped to an *Edge* with the *source* and *target* values. All additional information defined by an *EClass* and *EReference* by means of *EAttributes* are transformed to *Data*.

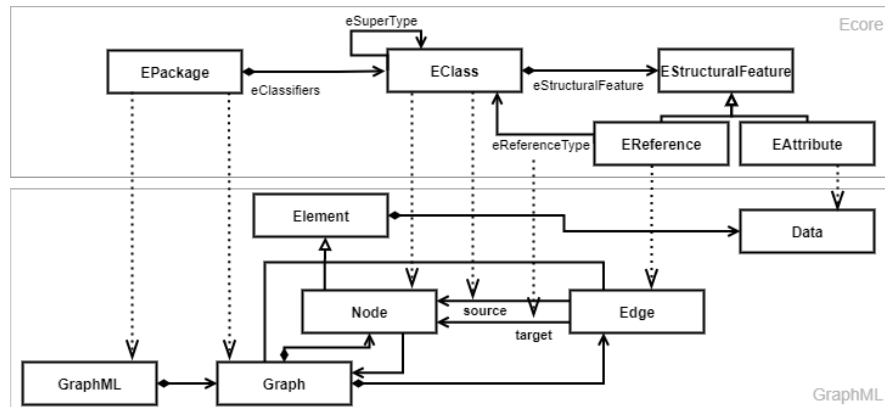


Fig. 2: Generic transformation from Ecore to GraphML.

In the following, some generic Ecore2GraphML transformation rules are overridden to address the specifics of the modeling tool (i.e., Archi) and the modeling language (i.e., ArchiMate (ArchiMate)). The first rule transforms a *Grouping*, *Folder*, or *View* element into a nested *Graph* (instead of creating a *Node*). All *nested elements* of the *Grouping* in the ArchiMate model will be added as *Nodes* in the nested graph. Secondly, since Archi stores the ArchiMate relationships as entities (i.e., *IArchiMateRelationshipEntitys*), instances of that entity need to be transformed into an *Edge* with additional edge data to store the relationship endpoints.

3.2 Graph-based Analysis of ArchiMate models

Once the transformation from ArchiMate models into GraphML is achieved, enterprise architects can analyze the resulting graph structure. Table 1 lists several graph metrics and maps them to exemplary Competency Questions (CQ) an enterprise architect might have, and that can be responded to by that metric. Noteworthy to say is that the GraphML specification opens the door to manifold graph analysis algorithms already implemented in openly available tools.

Table 1: Interpretation of sample graph metrics for ArchiMate models

Graph metric	EA interpretation and exemplary competency question (CQ)
Centralities	
Degree	The higher the value the more edges a node has. CQ: How many business services are used by one business role?
Closeness	How close is a Node to the other graph components. CQ: What is the closest switch that can be used to connect two servers?
Betweenness	How important is a Node in connecting different parts of a graph? CQ: What is the impact of removing a web server?
Community Detection	
Connected Components	For one community there exists a path from each node to another one without considering the direction of the relationship. CQ: Which connections exist between two network components?
Strongly Connected Components	For one community every node is reachable from every other node when considering the direction of the relationship. CQ: Can each device in a group exchange information with another one?

4 Implementation

In this section, we report on the prototypical implementation of our framework (see Fig. 1) called extensible Graph-based Enterprise Architecture Analysis (eGEAA). eGEAA is cloud-based, thereby offering the transformation and analysis functionality without the need to install any software. The implementation and the example models can be found in the accompanying repository [5].

The architecture of the eGEAA platform consists of three components (cf. Fig. 3): *Modeling*, *eGEAA Cloud*, and *Third Party Tools*. In the modeling component, users export the created enterprise architecture models created with

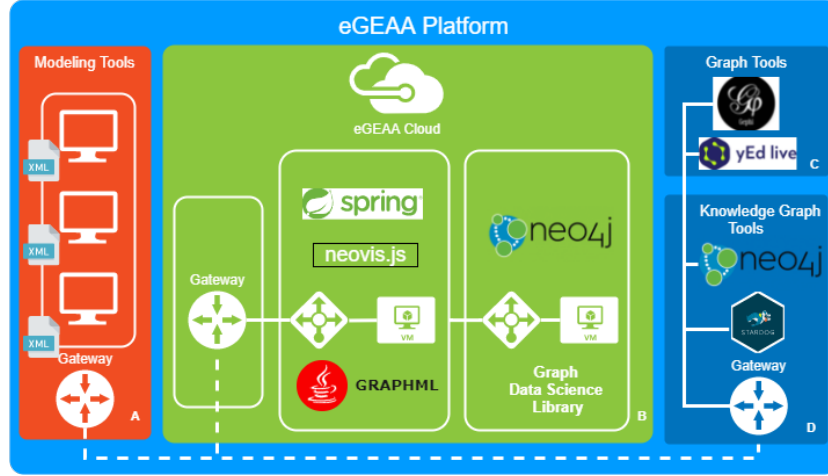


Fig. 3: eGEAA Cloud platform architecture.

any EMF- or ADOxx-based modeling tool in XML format, which serves as the input for the *eGEAA Cloud* component. *eGEAA Cloud* consists of two virtual machines running the web server and the database, respectively. The web server serves the web application and the model transformation service, while the Neo4j database stores the graph and offers additional functionality. The front-end is realized with the *neovis.js* [8] library, which provides a rich set of graph visualizations. If further analysis is required, *eGEAA Cloud* enables the export of the GraphML files, which can then be used as an input for any dedicated graph analysis or Knowledge Graph tools like Gephi, yEd, Neo4j and Stardog [6]. Due to limited space, in the following, we focus on the core functionality provided by the *eGEAA Cloud* platform.

Model Transformation & Inspection The *eGEAA Cloud* platform offers the functionality to upload an EA model in XML format, transform it into a GraphML-conforming XML format, and inspect both source and transformed model in the browser. Moreover, the created GraphML file can be directly downloaded or accessed via the database reference.

Graph Visualisation & Analysis The *eGEAA Cloud* platform enables the customization of the graph visualization in the browser (e.g., size, color, and labels of the graph elements) by providing visualization parameters or executing graph algorithms. *eGEAA Cloud* provides pre-defined graph centrality and community detection algorithms. When an algorithm is selected, the corresponding Neo4j command is shown, enabling the user to either directly execute or customize it. Eventually, users can define powerful Neo4j queries, directly execute them on the Neo4j database, and investigate the query results in the browser.

Interoperability *eGEAA Cloud* provides the reference to the graph in the Neo4j database. This enables a direct connection via the Neo4j Desktop Browser or any third-party graph analysis software for further analysis.

5 Evaluation

In the following, we will report the extent to which the individual research objectives specified in the introduction have been achieved.

RO-1: Generic Transformation Framework. The first research objective is further decomposed into three research objectives, formulated as requirements for the generic transformation: 1. applicability for arbitrary conceptual modeling languages, 2. applicability for models created with different meta-modeling platforms, and 3. enabling the use of third-party tools. In contrast to existing approaches, we thus aim to develop a generic transformation that can be widely applied and that is open for future extensions.

To evaluate to what extent our approach meets RO-1, we used two different EA modeling tools: *Archi*¹ which is based on Ecore and *TEAM* [4] which is ADOxx-based; and three third-party graph analysis tools: Neo4j, Gephi, and yEd. Due to the limited space, we will show selected results for the ArchiMate models. The supplementary material in the Github repository [5] features all evaluation experiments (also comprising different modeling languages).

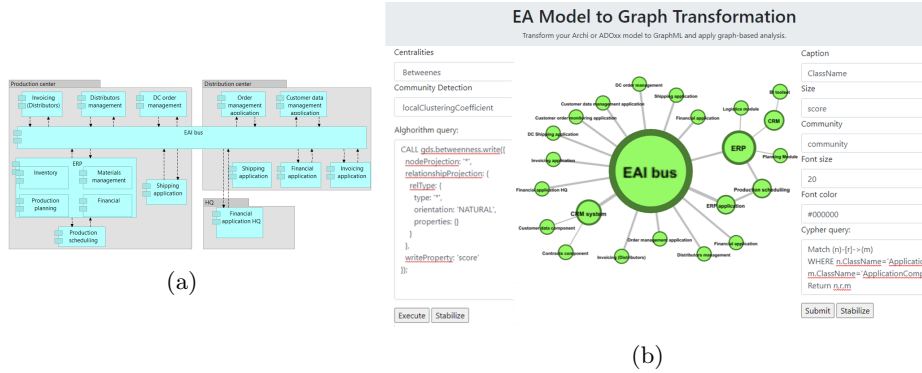


Fig. 4: ArchiMetal Application Architecture [2] (left) and User Interface of the eGEAA Cloud platform (right)

As a running example throughout the evaluation we use the publicly available ArchiMetal [2] case (see Fig. 4a). Fig. 5 shows the result of importing the transformed GraphML specification of the ArchiMetal case in the three third-party tools, thereby providing evidence on the validity of the transformation output. Consequently, we can state that we achieved a generic transformation.

RO-2: Benefits of Graph-based Analysis for EAs. We evaluated the possibility of responding to the previously introduced competency questions (cf. Tab. 1) using the ArchiMetal example. We exemplify how graph centrality and community detection metrics can be used in the analysis process. Fig. 5c exemplary shows the result of applying the Betweenness centrality metric to the

¹ Archi modeling tool [online]: <https://www.archimatetool.com/>

ArchiMetal example in the yEd tool. The higher the Betweenness value, the more intense the color and the larger the square. The importance of the *EAI bus* for the application layer can be easily detected when looking at the resulting graph. From a business perspective, this graph-based analysis indicates the severity with which individual components of the application architecture threaten the continuation of business operations.

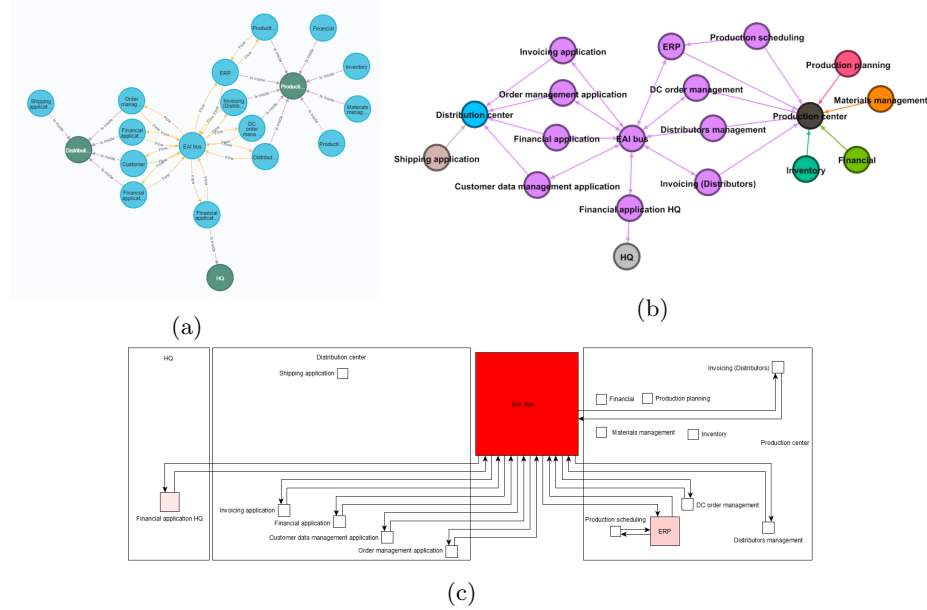


Fig. 5: Transformed ArchiMetal example in Neo4j (a), Gephi (b), and yEd (c)

Exemplary for the community detection metrics, we applied the *Strongly Connected Components* metric on the ArchiMetal example. Fig. 5b shows the resulting graph in the Gephi tool. Identified communities are color-coded. Consequently, one can visually grasp that all purple application components belong to the same community, i.e., can communicate with each other. The previous examples showcase the potential of supporting enterprise architects by the application of graph metrics. Much more can be done obviously. However, we believe that eGEAA Cloud can support the community in developing a taxonomy of graph-based analysis of EAs in the future.

RO-3: EA Graph-based Analysis Platform. Fig. 4b shows an overview of the user interface of the eGEAA Cloud platform realizing the architecture defined in Fig. 3. On the top left side, users can select the graph metric they want to apply. The corresponding query is then presented in the text area on the lower left side. Here is where users can customize the pre-defined query before validating and eventually executing it. On the right side, different customization possibilities for graph rendering are provided. On the lower right side, a text field

allows the user to easily directly define Cypher queries that will be executed on the corresponding Neo4j database. Eventually, the central area uses the `neovis.js` package to render the graph. Consequently, we can state that we achieved RO-3.

6 Concluding remarks

With this paper, we introduced a generic and extensible framework and an open source implementation of the eGEAA Cloud platform [5] for transforming conceptual models into graphs. We instantiated the approach to show, how graph analysis can support enterprise architects in decision making. We used several existing EA models and tools, realized with the two widely used meta-modeling platforms EMF and ADOxx, to evaluate the feasibility of transforming and analyzing EA models.

We hope that this generic contribution raises interest in this promising research field of conceptual model analysis. We don't see this proposal as a means to replace enterprise architects but rather to complement their domain expertise with a more scalable approach [24]. In our future research, we aim to define a taxonomy of graph-based EA analysis metrics and extend our approach to support EA mergers. Transforming two EAs into graphs and applying, e.g., euclidian distance or overlap graph metrics might be handy, especially for large EAs. Future work will also concentrate on qualitative analysis. e.g., by proposing complex queries that span multiple ArchiMate layers like impact analysis or the use of the graph structure to detect EA Smells [26].

References

1. Ahlemann, F., Stettiner, E., Messerschmidt, M., Legner, C.: Strategic Enterprise Architecture Management. Springer, Berlin, Heidelberg (2012)
2. Archi: Archimetal (2016), <https://github.com/archimatetool/ArchiModels/tree/master/ArchiMetal>
3. Barbosa, A., Santana, A., Hacks, S., Stein, N.v.: A taxonomy for enterprise architecture analysis research. In: 21st International Conference on Enterprise Information Systems. vol. 2, pp. 493–504. SciTePress (2019)
4. Bork, D., Gerber, A., Miron, E., van Deventer, P., van der Merwe, A., Karagiannis, D., Eybers, S., Sumereder, A.: Requirements engineering for model-based enterprise architecture management with archimate. In: Proceedings EOMAS 2018. pp. 16–30. Springer (2018)
5. Bork, D., Smajevic, M.: Companion source code repository of the egeaa platform. <https://github.com/borkdominik/eGEAA> (2021)
6. Brandes, U., Eiglsperger, M., Lerner, J., Pich, C.: Graph markup language (graphml). In: Tamassia, R. (ed.) Handbook of graph drawing visualization, pp. 517–541. Discrete mathematics and its applications, CRC Press (2013)
7. Buschle, M., Johnson, P., Shahzad, K.: The enterprise architecture analysis tool - support for the predictive, probabilistic architecture modeling framework pp. 3350–3364 (2013)
8. Contrib, N.: `neovis.js`. <https://github.com/neo4j-contrib/neovis.js> (2021)

9. Dehmer, M., Emmert-Streib, F., Shi, Y.: Quantitative graph theory: A new branch of graph theory and network science. *Information Sciences* **418-419**, 575 – 580 (2017)
10. Dehmer, M., Kraus, V., Emmert-Streib, F., Pickl, S.: What Is Quantitative Graph Theory?, pp. 1–33 (11 2014)
11. Florez, H., Sánchez, M., Villalobos, J.: A catalog of automated analysis methods for enterprise models. *SpringerPlus* **5**(1), 1–24 (2016)
12. Gampfer, F., Jürgens, A., Müller, M., Buchkremer, R.: Past, current and future trends in enterprise architecture—a view beyond the horizon. *Computers in Industry* **100**, 70–84 (2018)
13. Herman, I., Melancon, G., Marshall, M.S.: Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics* **6**(1), 24–43 (2000). <https://doi.org/10.1109/2945.841119>
14. Iacob, M.E., Jonkers, H.: Quantitative analysis of enterprise architectures. In: *Interoperability of Enterprise Software and Applications*, pp. 239–252. Springer (2006)
15. Johnson, P., Ekstedt, M.: *Enterprise architecture: models and analyses for information systems decision making*. Studentlitteratur (2007)
16. Jugel, D.: An integrative method for decision-making in EA management. In: Zimmermann, A., Schmidt, R., Jain, L.C. (eds.) *Architecting the Digital Transformation - Digital Business, Technology, Decision Support, Management*, pp. 289–307. Springer (2021)
17. Jugel, D., Kehrer, S., Schweda, C.M., Zimmermann, A.: Providing EA decision support for stakeholders by automated analyses. In: *Digital Enterprise Computing (DEC 2015)*. pp. 151–162. GI (2015)
18. Lankhorst, M., et al.: *Enterprise architecture at work*, vol. 352. Springer (2009)
19. Messina, A.: Overview of standard graph file formats. Tech. Rep. RT-ICAR-PA-2018-06 (2018), <http://dx.doi.org/10.13140/RG.2.2.11144.88324>
20. Naranjo, D., Sánchez, M., Villalobos, J.: Primrose: A graph-based approach for enterprise architecture analysis. In: *International Conference on Enterprise Information Systems*. pp. 434–452. Springer (2014)
21. OMG: ArchiMate® 3.1 Specification. The Open Group (2019), <http://pubs.opengroup.org/architecture/archimate3-doc/>
22. Pachayappan, M., Venkatesakumar, R.: A graph theory based systematic literature network analysis. *Theoretical Economics Letters* **8**(05), 960–980 (2018)
23. Pittl, B., Bork, D.: Modeling digital enterprise ecosystems with archimate: a mobility provision case study. In: *International Conference on Serviceology*. pp. 178–189. Springer (2017)
24. Potts, M.W., Sartor, P., Johnson, A., Bullock, S.: A network perspective on assessing system architectures: Foundations and challenges. *Systems Engineering* **22**(6), 485–501 (2019)
25. Roelens, B., Steenacker, W., Poels, G.: Realizing strategic fit within the business architecture: the design of a process-goal alignment modeling and analysis technique. *Software & Systems Modeling* **18**(1), 631–662 (2019)
26. Salentin, J., Hacks, S.: Towards a catalog of enterprise architecture smells. In: Gronau, N., Heine, M., Krasnova, H., Poustcchi, K. (eds.) *15. Internationalen Tagung Wirtschaftsinformatik, WI 2020*. pp. 276–290. GITO Verlag (2020)
27. Santana, A., Fischbach, K., Moura, H.: Enterprise architecture analysis and network thinking: A literature review. In: *2016 49th Hawaii International Conference on System Sciences (HICSS)*. pp. 4566–4575. IEEE (2016)