# View and Viewpoint Reconstruction for Assisting the Preparation of Participatory Modeling Sessions

David Naranjo and Mario Sánchez

Department of Systems and Computing Engineering
Universidad de los Andes
<da-naran><mar-san1>@uniandes.edu.co

**Abstract.** Due to the inherent complexity of Enterprise Models, it is common to employ multiple views for model exploration. Currently, this exploration remains a daunting task in collaborative modeling sessions, due to ineffective mechanisms for organizing views and presenting the relevant ones to each stakeholder. To address these issues, we propose an approach that reconstructs view-related information from the structure of the Enterprise Model and tool metadata. Then, we perform a viewpoint analysis, in order to identify the most appropriate view for each participant of the modeling session.

**Keywords:** enterprise models, architectural views and viewpoints, participative modeling

## 1 Introduction

To gain valuable knowledge and insights, we can learn from models in two manners: By constructing new models, or by manipulating existing ones [1]. In both learning tasks, we usually focus on particular fragments (views) of the model, from the perspective of specific concerns [2]. To address these concerns, we use multiple viewpoints [3], which establish the conventions for constructing, interpreting and analyzing a view [2]. For instance, when using the ArchiMate language [4], we can choose the relevant viewpoints (e.g. Capability Map Viewpoint, or Value Stream Viewpoint) for modeling certain fragment of an architecture.

The construction of a model from scratch demands a considerable effort, but the reward is that the modeler ends up with extensive knowledge of the model. However, given the advances in **Participative Modeling** [5], as well as a concerted effort to expand the reach of enterprise modeling to non-expert modelers [6], it is increasingly common that enterprise models have several authors, and address different stakeholders. Model exploration is critical for ensuring model quality, learning from unfamiliar parts of the model, and transmitting this knowledge to larger audiences. However, tool support for these tasks is insufficient for collaborative modeling sessions, as current tools are tailored for traditional (individual) modeling and have several limitations when dealing with large models and multiple views.

In this paper, we argue that we can take advantage of structural and semantic information to offer better support for view-based model navigation and the selection of appropriate views to a group of different stakeholders in a collaborative modeling project. For this purpose, we propose an approach for reconstructing a cross-reference model that describes the relationships between ArchiMate views, offering a *map* for navigating the model, as well as an analysis method for discovering the viewpoints that can be associated to every view.

This paper is structured as follows: First, we give a short theoretical background and introduce the problem to be addressed. Then, in Section 3 we describe our approach. Section 4 illustrates the approach with a widely used Case Study. Afterwards, Section 5 offers the results of our analysis, and Section 6 provides related and future work.

## 2 View-based Navigation in Participatory Modeling

The complexity of large models gets in the way of obtaining global and detailed understanding of the systems that they represent. This complexity is tamed by separating models into manageable fragments (**views**), and knowing how these fragments are related (**cross-reference view**). The ISO 42010 standard [2] offers a method for modeling and analyzing an architecture through views that address particular concerns of stakeholders. In particular, a viewpoint describes the element and relationship types allowed in a view, as well as the *"languages, notations, model kinds, design rules, and/or modelling methods, analysis techniques and other operations on views"* [2].

While the ISO 42010 standard establishes a common conceptual framework, it does not offer guidance on which viewpoints should be used. To obtain this guidance, we can make use of several architecture frameworks, e.g. TOGAF, which suggest a collection of viewpoints as first-level mechanisms –and a starting point– for modeling the architecture of a company. Many modeling languages (e.g. ArchiMate) already suggest useful viewpoints in their documentation, and modeling methods such as 4EM [7] offer a method for organizing views by the refinement of goal models to produce process, resource, and technical sub-models, with clear relationships between them. An important aspect of 4EM is that it is a participatory approach where stakeholders are included in the modeling process.

Approaches to **Participatory Modeling (PM)** [5,7] consist of Group Modeling Sessions, where expert modelers act as facilitators, and stakeholders are the real creators and owners of the model [8]. To avoid losing interest and commitment of stakeholders, PM sessions require additional setup and planning [8], with tasks such as selecting an appropriate modeling language, identifying stakeholder background and needs, and creating views of the model that are appropriate for the problem at hand. In large PM projects, the use of a preliminary model can speed up the collaborative modeling process and raise critical discussion topics [9]. Navigation of existing models is done with the aid of modeling tools, which can be employed for browsing views, selecting reusable fragments, as well as creating user-defined views [5].

### 2.1 Problem Statement

Nowadays, views are the main way for model navigation and manipulation. To preserve model quality and completeness, current approaches to multi-view modeling focus on consistency checks [10, 11], and current modeling methods and tools offer several aids for this purpose. However, when envisioning multi-view modeling in 2020, von Hanxleden et al. [12] acknowledge that the modeling community is too much focused on consistency, while neglecting pragmatic aspects that are also important. More recently, Sandkuhl et al. [6] call into attention that Enterprise Modeling needs more lightweight approaches that do not focus on traditional qualities like completeness and coherence, but on usefulness and impact of models. The authors underscore that the scope of models must be managed to ensure that the right content is represented in the right way for each actor of the modeling process, and research is needed on how to automatically derive and maintain model views tailored to particular purposes.

Right now, the problem is that *session facilitators* and *tool operators*, two critical roles in PM sessions [5], have **inadequate support for browsing existing views and deriving new views that are appropriate to session participants**: Navigation mechanisms offered by current modeling tools are made for traditional (individual) modeling, and are not as effective for PM sessions [13]. Furthermore, these tools have several limitations when visualizing, navigating, and interacting with large models [5, 14], and each tool has its own mechanisms for organizing and aggregating views, as externalized metadata that is not part of the model [15]. Sandkuhl et al. argue that tool-related research should investigate which types of concerns of which stakeholder groups can typically be supported by which types of models [6]. In this regard, viewpoints are useful for producing views that are tailored to particular stakeholders. However, a consequence of the heterogeneity of stakeholders in PM sessions implies that traditional viewpoints offered by languages such as ArchiMate might not not sufficient, or even too overarching (e.g. the Layered Viewpoint).

## 3 Proposed Approach

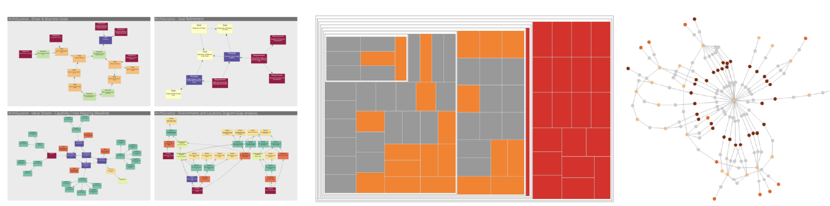


**Fig. 1.** Extracted Views (left), View Tree (center), and View Model (right).

In order to create a model of all views, we propose a structural analysis in three steps: First, we parse the entire model and extract all the views therein.

Then, we match the dependencies between views, and generate a hierarchical structure. Finally, we enrich the model with cross-references of elements and relationships embedded in each view (see Fig. 1).

**View Extraction** Several modeling tools provide mechanisms for exporting a model as interchange files that employ well-known formats, e.g. XMI, KM3, or XML. Tool-dependent **metadata** is embedded in these files, containing additional information, such as the structure, names, and contents of individual views. We can process the model file to find isolated clusters of model elements and relationships that can be interpreted as views. This is done by generating a graph structure of the model, with model elements as nodes, and relationships between elements as edges. Nodes are enriched with additional information, such as View name, creation date, and other *metadata*.

**View Tree Generation** Most modeling tools have a mechanism for organizing views, either by describing them as independent sub-models, or aggregated by tool constructs such as Folders. We can extract, also from tool metadata, these organizing criteria. In some cases, tool vendors provide the Tool Metamodel that describes these constructs. In the worst case, when no metamodel is available, it is possible to do a reverse engineering of the model file to identify aggregating constructs. This processing generates a hierarchical structure (a tree) of views.

**Cross-Referencing** Finally, the View Model is enriched with additional relationships between views that are inferred from tool metadata. This is done when the model contains *navigation views*, which are a special kind of views that do not contain model elements or relationships; instead, they reference other views.

### 3.1 Viewpoint Analysis

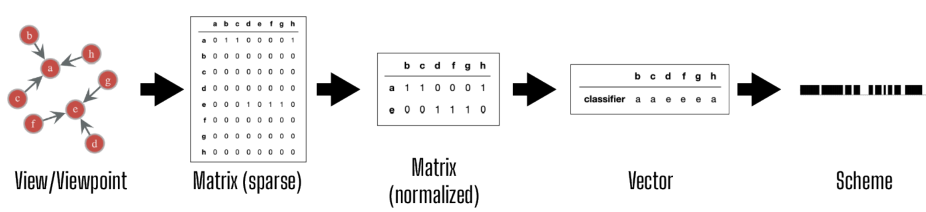


**Fig. 2.** Summary of the transformation from View/Viewpoint to Scheme

In order to identify which views are relevant to particular stakeholders, we create diagrams for the joint comparison of views and viewpoints. We start by creating **Schemes**, which are vectorized representations of views and viewpoints.

The translation from View to Scheme is done in an automated manner (see Fig. 2). Then, we apply a series of production rules for stacking and sorting Schemes. This can be seen as something akin to playing *Tetris*.

The idea behind Schemes is that any disjoint classification or partition of model elements and relationships has a hierarchical structure that can be represented as an adjacency matrix. A feature of these matrices is that they are fairly sparse, and can be represented in a more succinct manner using a simple visual notation that we call **Schemes**, which are vectorized representations of (meta)model fragments. Each dimension of the vector (i.e. the index of the array) corresponds to a particular concept of the metamodel, and a solid color inside each position corresponds to the presence of such concept.

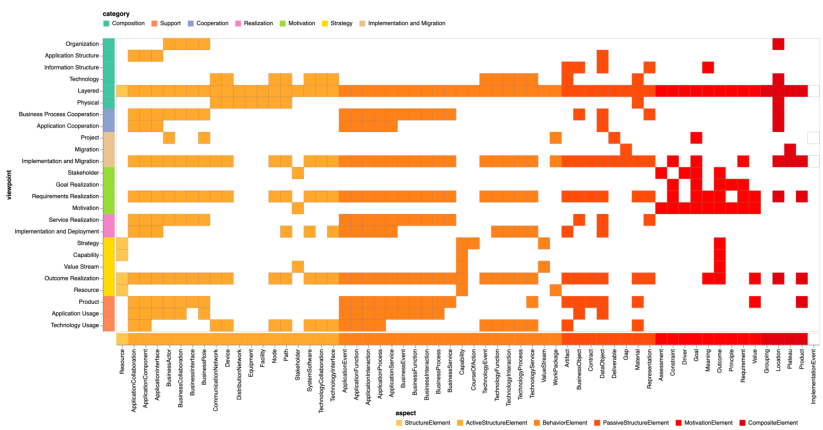


**Fig. 3. Left:** Standard Viewpoints Matrix.

Viewpoints can be represented as Schemes by indexing all the concepts of the metamodel, with a solid color for concepts that are present in the viewpoint. For example, Fig. 3 shows the matrix for all 25 example viewpoints that are in the ArchiMate standard. We applied two classifiers that apply orderings to the matrix: For sorting rows, we consulted the viewpoint classification of the standard that offers seven categories. For sorting columns, we classified ArchiMate concepts by their Aspect, also from the standard. Views can be represented in the same manner, by inserting solid squares if there are elements in the view that are instances of the concepts.

After having a Viewpoint Matrix and a View Matrix, we can calculate the similarity between views and viewpoints by applying a cross product between both matrices. The result of this product is a **Similarity Matrix** (see Fig. 5) that contains views as rows and viewpoints as columns. Each cell of the matrix has a shade of grey: The darker the color, the more compatible is a view to a viewpoint. The similarity value is calculated by applying vector multiplication.

## 4 Illustration

To illustrate our approach, we employ ArchiMate, a widely used Enterprise Modeling language. In particular, we make use of the ArchiSurance Case Study [16], but this approach can be applied to any ArchiMate model. The model for the case study is provided by The Open Group as an XML exchange file[1] that can be imported to different modeling tools. The entire model, which consists of 315 elements and 467 relationships, is embedded inside a larger one that includes view metadata. This data describes, among other things, tool-specific information and visual attributes of the diagrams. We use this complete model as a starting point for extracting view information.

### 4.1 View Reconstruction

First, we obtained the tool metamodel from the source code repository of Archi, a popular tool for ArchiMate models. This metamodel has visualization information for displaying the different views of the model with the visual ArchiMate notation. Inside this metamodel we found the following concepts that are relevant to extract view information: An **ArchimateModel** contains **Folder** elements, and each folder has **ArchimateDiagramModel** elements that represent each diagram. Each one contains **DiagramModelArchimateObject** instances, which can be either **ArchimateElement**s, or **DiagramModelReference**s.

### 4.2 Viewpoint Analysis

According to the tool metadata, the ArchiSurance model only employs 7 (out of 25) standard viewpoints, and just 9 views (out of 54) have an assigned viewpoint. The lack of viewpoint information for most views emphasizes the need of a Viewpoint Analysis, as views with unassigned viewpoints don't impose restrictions on which concepts are allowed. In order to analyze the similarity of a view to all viewpoints, we create the Similarity Matrix using the construction process of Section 3 (see Fig. 5). As stated, the darker the color of a cell, the more appropriate is the respective View to its respective Viewpoint.

## 5 Results

The extraction process yielded 75 distinct views. Fig. 1 shows a small sample of extracted views, the View Tree and the View Model for the ArchiSurance model. Fig. 4 shows two close-ups of the View Model. After examining each view, we tried to make sense of the organization scheme used by the original modelers. However, folder names were not descriptive, and their hierarchy was too shallow. As a next step, we studied the architecture description document [16], and identified all the views that appear in the document. Contrasting these diagrams with the 75 views extracted from the model led us to divide views into three categories:

[1] See: https://publications.opengroup.org/more-categories/archimate-models/y194m

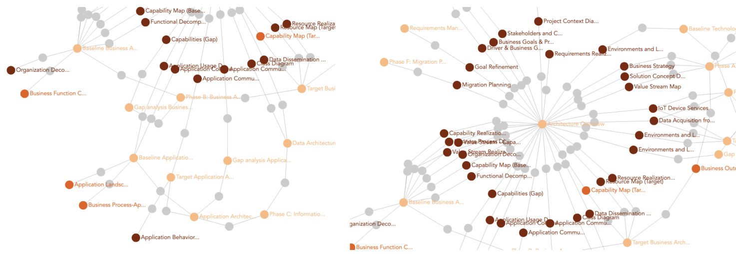

**Fig. 4.** Details of the View Model.

- **Navigation Views (21):** These views do not contain any ArchiMate element, they just have links to other views.
- **Document Diagrams (33):** These views appear in the Architecture Description document.
- **Miscellaneous Views (21):** The rest of the views (including empty ones).

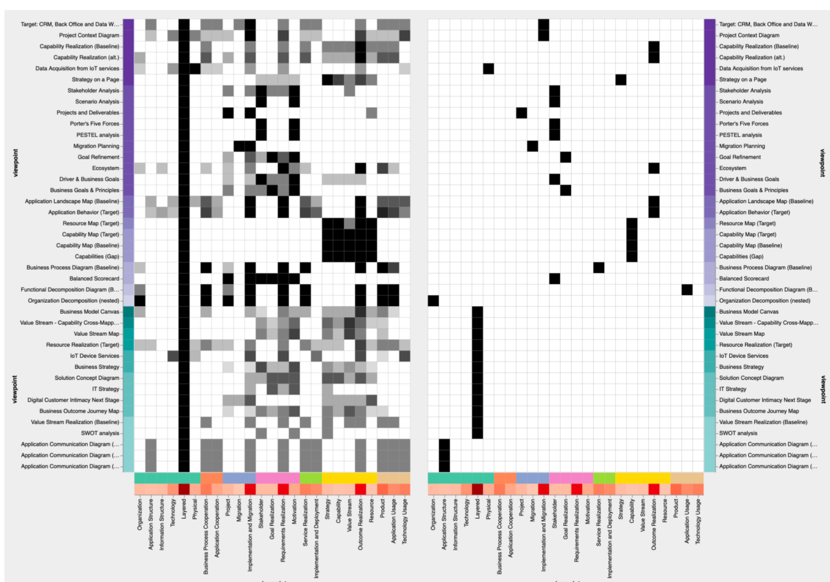

**Fig. 5.** Coverage of Views by Standard Viewpoints. **Left:** Similarity Analysis. **Right:** Chosen Viewpoints

With our method, we discovered that most views have **total coverage** by standard viewpoints. There are 5 views with exactly one match (ignoring the Layered viewpoint), so it is safe to assign the matched viewpoint to the view. However, there are 21 views that have two or more matched viewpoints with

100% coverage, so we sorted standard viewpoints by their size, and picked the smallest matched viewpoint.

Furthermore, there are 15 unassigned views that have **partial coverage** by viewpoints. We proceeded to examine the differences between the concepts used by the view and the viewpoints with best match. In most cases, the difference is just one concept: *Grouping*. This concept does not change the semantics of any of the suggested viewpoints, so it is safe to ignore this concept and pick one of the best matches. There are three courses of action for the remaining views:

- Assign them to the Layered viewpoint
- Enrich one of the matched viewpoints with the additional concept(s)
- Create custom viewpoints

For simplicity, we opted for the first case, but we suggest taking any of the other two courses of action, depending on the context and usage of the views. Fig. 5 shows to the right the chosen viewpoints.

### 5.1 Discussion

In View Reconstruction, we identified 42 views that were not part of the Architecture Description Document of the case study. Furthermore, we identified the main point of entry of the model, a Navigation View that links to other views. Finding this view in the Archi modeling tool is more difficult, as the tool operator needs to expand several folders and possibly open each view to make sense of the organization scheme of the modelers. Conversely, by reconstructing a View Model, we can visually inspect the organization of views and choose the ones that are more relevant.

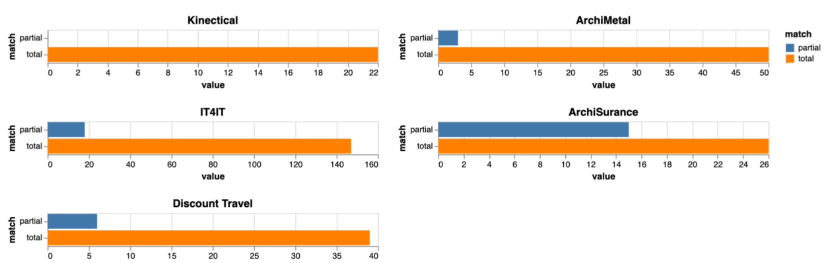


**Fig. 6.** Partial and Total matches for five Enterprise Models.

In the Viewpoint Analysis, we found that in most cases the ArchiSurance modelers tried to employ the example viewpoints from the standard. We did not expect the appearance of so many partial matches (12) so, for comparison, we applied our method to other four ArchiMate models[2], and discovered that on

[2] We excluded navigation views and empty views from this analysis

average 92% of unassigned views can be assigned to a standard viewpoint (see Fig. 6). The outlier was the ArchiSurance model, which has more partial matches, but the reason behind this larger number is that modelers introduced one or two additional concepts to these views, and provided three valid courses of action. Some views, e.g. *Business Model Canvas*, probably need custom viewpoints.

# 6 Conclusion

Even medium-sized models (such as the ArchiSurance one) can have a large number of views. Navigating among these views using the mechanisms offered by modeling tools is a daunting task that needs better assistance in PM sessions. Our method allows the reconstruction and visualization of all views and their cross-references, providing a map for model exploration. Furthermore, our Viewpoint Analysis allows the automated classification of views with 100% coverage, and informs the best matches for views with partial coverage, as well as the difference in concepts. With this information, Session Facilitators can select the views that are most appropriate to a particular stakeholder, by selecting the relevant viewpoints that best represent their concerns.

This approach makes part of a larger endeavor for providing assistance to *Tool Operators* and *Facilitators* of PM Sessions. This assistance is provided in tasks such as model exploration and (re)design of alternative solutions, as well as the preservation and explanation of the rationale behind design decisions. Currently, we are designing an experiment to evaluate the effectiveness of the whole approach for addressing *Wicked Problems*.

## 6.1 Related Work

Approaches for recovering view information and facilitating model navigation [15, 17], while valuable, introduce additional complications, e.g. they require specialized knowledge and careful planning, or access to the source code of the modeling tool for annotating metamodel concepts. A limitation of our approach is that View Extraction needs depends on the modeling tool that produced the model. In most cases it can be very similar to a reverse-engineering effort of the model file produced by the modeling tool. An advantage of our approach is that it does not need a modification of the tool internals, and that it provides a visual and interactive visualization of the view model. To date, we could not find similar approaches for discovering compatible viewpoints of a view.

*Acknowledgment.* This work was supported by the COLCIENCIAS grant 727 for doctoral studies.

# References

1. Morgan, M.S., Morrison, M. (eds.): Models as Mediators: Perspectives on Natural and Social Sciences. Ideas in Context, Cambridge University Press, Cambridge ; New York (1999)

2. ISO: ISO/IEC/IEEE Systems and software engineering: Architecture description. ISO/IEC/IEEE 42010:2011(E) (Revision of ISO/IEC 42010:2007 and IEEE Std 1471-2000) pp. 1–46 (Dec 2011)
3. Lankhorst, M.M., van der Torre, L., Proper, H.A.E., Arbab, F., Hoppenbrouwers, S.J.B.A., Steen, M.W.A.: Viewpoints and Visualisation. In: Enterprise Architecture at Work, pp. 171–214. The Enterprise Engineering Series, Springer (2017)
4. The Open Group: ArchiMate 3.1 Specification. Tech. rep. (2019)
5. Stirna, J., Persson, A.: Enterprise Modeling: Facilitating the Process and the People. Springer International Publishing, Cham (2018)
6. Sandkuhl, K., Fill, H.G., Hoppenbrouwers, S., Krogstie, J., Matthes, F., Opdahl, A., Schwabe, G., Uludag, Ö., Winter, R.: From Expert Discipline to Common Practice: A Vision and Research Agenda for Extending the Reach of Enterprise Modeling. Business & Information Systems Engineering 60(1), 69–80 (Feb 2018)
7. Sandkuhl, K., Stirna, J., Persson, A., Wißotzki, M.: Enterprise Modeling: Tackling Business Challenges with the 4EM Method. The Enterprise Engineering Series, Springer Berlin Heidelberg, Berlin, Heidelberg (2014)
8. Persson, A.: Enterprise Modelling in Practice: Situational Factors and Their Influence on Adopting a Participative Approach. Ph.D. thesis, Stockholm University, Department of Computer and Systems Sciences, Kista, Sweden (2001)
9. Renger, M., Kolfschoten, G.L., de Vreede, G.J.: Challenges in Collaborative Modeling: A Literature Review. In: Dietz, J.L.G., Albani, A., Barjis, J. (eds.) Advances in Enterprise Engineering I. pp. 61–77. Lecture Notes in Business Information Processing, Springer, Berlin, Heidelberg (2008)
10. Reineke, J., Tripakis, S.: Basic Problems in Multi-View Modeling. In: Ábrahám, E., Havelund, K. (eds.) Tools and Algorithms for the Construction and Analysis of Systems. pp. 217–232. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2014)
11. Awadid, A., Bork, D., Karagiannis, D., Nurcan, S.: Toward Generic Consistency Patterns in Multi-View Enterprise modelling. In: Bednar, P.M., Frank, U., Kautz, K. (eds.) 26th European Conference on Information Systems: Beyond Digitization - Facets of Socio-Technical Change, ECIS 2018, Portsmouth, UK, June 23-28, 2018. p. 146 (2018)
12. von Hanxleden, R., Lee, E.A., Motika, C., Fuhrmann, H.: Multi-view Modeling and Pragmatics in 2020. In: Calinescu, R., Garlan, D. (eds.) Large-Scale Complex IT Systems. Development, Operation and Management. pp. 209–223. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2012)
13. Ghidini, C., Rospocher, M., Serafini, L., Faatz, A., Ley, T., Pammer, V., Lindstaedt, S.: Collaborative enterprise integrated modelling. In: Proceedings of the 5th Workshop on Semantic Web Applications and Perspectives (SWAP2008). p. 10. CEUR-WS.org, Rome, Italy (2008)
14. Naranjo, D., Sánchez, M., Villalobos, J.: Evaluating the capabilities of Enterprise Architecture modeling tools for Visual Analysis. The Journal of Object Technology 14(1), 3:1 (2015)
15. Jakob, J., Königs, A., Schürr, A.: Non-materialized Model View Specification with Triple Graph Grammars. In: Graph Transformations. pp. 321–335. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (Sep 2006)
16. Jonkers, H., Band, I., Quartel, D., Lankhorst, M.: ArchiSurance Case Study Version 3.1. Tech. rep., The Open Group (2019)
17. Jiménez-Pastor, A., Garmendia, A., de Lara, J.: Scalable model exploration for model-driven engineering. Journal of Systems and Software 132, 204–225 (Oct 2017)