Towards a Unified and Modular Approach for Visual Analysis of Enterprise Models

David Naranjo, Mario Sánchez, Jorge Villalobos
Department of Systems and Computing Engineering
Universidad de los Andes, Bogotá, Colombia
{*da-naran,mar-san1,jvillalo*}@*uniandes.edu.co*

***Abstract*—From the point of view of an enterprise architect, the visual analysis of the structure of an Enterprise Model can provide powerful insights and facilitate various analysis tasks. However, analysis methods and visualizations that offer the most valuable results usually depend on the concepts found on the Enterprise Metamodel, and are subjected to a concrete EA modeling tool. In this paper we argue that many structural and domain-specific analysis methods can be automated, composed, and their results visualized in a modular manner. To achieve this, we describe the foundations of a flexible and configurable approach for EA Analysis that is based on components that encapsulate metamodel-specific analysis and visualization techniques, supported by a metamodel-independent platform. The approach is explained with ArchiMate by using a module, currently in development, which we call PRIMate. Benefits from this approach include the integration with existing modeling environments, the reuse of analysis and visualization capabilities in any metamodel, as well as an explicit, composite, and traceable visual analysis process.**

***Keywords*-enterprise architecture; analysis; visualization; tool support**

I. Introduction

For its architects, who hold the 'holistic' and 'maestro' [1] view of an organization, Enterprise Architecture can be a demanding discipline that brings a rather large set of requirements. Ranging amongst "very specific programming skills to broad leadership qualities as well as the ability to develop a business strategy" [2], architects must also acquire a deep knowledge of the enterprise under several perspectives, and be able of abstracting accurately the most relevant aspects of this knowledge into an Enterprise Model.

Several studies on the role of the Enterprise Architect [1][2][3][4][5] suggest that *'Analytical Skills'* are a core competency of an architect. These competencies, often overshadowed by other skills (such as modeling and communicating with stakeholders), can be seen [1] as the ability to "identify problems and break them down into manageable pieces that can be studied and assessed to work out strategies", and is based in evidence collection, in order to arrive to better, informed decisions.

Instead of always starting from scratch, the support for the analysis tasks of these architects is located in the tools that serve as an architecture modeling, management and repository solution. These tools are based on an EA metamodel, and all their modeling, analysis, and visualization capabilities depend on this metamodel.

For instance, most EA management tools provide reports of change impact analysis. However, it is more difficult to find tools that offer more complex analysis techniques for the calculation of performance metrics, e.g. for calculating the throughput of a system with Data Flow Networks [6]. The reason is that domain-specific analysis techniques often have a dependency on certain entities and relations, and they also require the introduction of additional facts (attributes, elements or relations) to the enterprise model.

Thus, we can say that the architect is left with partial tool support to his analysis tasks, opening the door to incomplete or inaccurate assessments. In order to arrive to more powerful insights, we require a more comprehensive toolbox of automated analysis techniques that allows to combine and create new ones easily. Furthermore, given that views and reports offered by EA Management tools rarely offer the whole panorama of the architecture [7], it is critical to visualize the new facts introduced by these techniques, as well as navigate, interact and diagnose the model in order to discover anomalies in its structure [8].

For these reasons, we propose a visual and holistic approach for EA analysis. It is grounded on the **PRIMROSe** framework, a flexible platform for Visual Analysis, currently in active development, that focuses on *Overview Visualizations* and allows the creation of a) generic analysis functions for topological analysis, and b) metamodel-specific analysis functions for specialized analysis methods. These last functions are encapsulated in an isolated module that also contains the description of how they should be visualized, separating visualization and analysis capabilities and offering the possibility of managing several metamodels at the same time. This approach will be illustrated with **PRIMate**, a module for visual analysis of ArchiMate models.

The structure of this paper is as follows: Section II will discuss some aspects of analysis and visualization. In Section III we will describe the PRIMROSe framework. Section IV is an insight into PRIM modules, the packaging components of a metamodel-specific analysis environment, and Section V is the illustration of these PRIM modules with ArchiMate, and includes a case study. Finally, Section VI will briefly describe similar approaches, and Section VII will be a space for discussion and future work.

II. Rethinking EA Analysis

In order to design, develop, and communicate an Enterprise Model[1], an architect "needs to think reflectively, as well as to able to analyze and visualize results." [2]. In this section we will describe these two critical activities, *to Analyze*, and *to Visualize*, that while being mutually dependent, usually are not given equal attention.

A. Analysis Techniques

Model-driven EA Analysis focuses on bringing additional facts to the Enterprise Model, with the purpose of answering concrete questions (e.g. *What is the average processing time of an order?*), as well as diagnosing the model by identifying critical elements, relations and groups that follow some pattern.

*1) **Answering Concrete Questions**:* An architect can begin the analysis process with a set of questions and concerns in his mind. In other words, he has some suspicions that need to be confirmed (or denied) by evidence, which means that he must perform a deeper examination of the model. Most of these concerns can be translated into quantitative measures (See Table I), e.g. performance indicators, or into a selection of model elements or relations (e.g. change impact, queries).

Given the value of their outcomes, the research community is constantly proposing new EA and domain-specific analysis techniques (see Table I) that introduce new facts to the model in the form of:

- New attributes on concepts of the metamodel as well as their respective values on elements in the model.
- New concepts that represent relationships with attributes (e.g. cost, weight).
- New relations that are derived from existing ones (e.g. paths).
- New concepts that group elements and/or relations with common properties.

These techniques can be classified using several dimensions, such as type of analysis (quantitative or functional) and type of technique (analytical or simulation) [9], as well as its body of analysis, time reference [10], among others. Ramos et al. [11] argue that we can also classify and describe an analysis technique by its set of structural attributes that enrich the model, the affected entities and relations, and a formal description of the algorithm used.

*2) **Diagnosing the Model**:* However, we have to accept that not all analysis endeavors begin with a suspicion in mind. In some cases, we need to explore the model to discover patterns and make a diagnosis. This is not a new idea; experts in any domain, from medicine to carpentry, can visually detect something is good or bad by quickly examining a specialized artifact, from an x-ray to a woodwork blueprint. In more familiar fields, we can find examples of this practice: For instance, the search of 'code smells' is a valid analysis technique in Software Engineering, as well as automated pattern matching techniques such as symptom matching [18][19][20]. On EA, we can make a relation between certain code smells [21] and EA anti-patterns [22]. While research on this type of analysis is still on its infancy, we can find interesting approaches, such as the discovery of hidden structures and critical clusters on Enterprise Models [17].

Technique	Concern	Reference
Quantitative Performance Analysis	Workload, Processing time, Utilisation	[12]
Change Impact Analysis	Impact of a Change	[13]
Probabilistic Relational Models	SLA Fulfillment, Response Time, Reliability, Security, Accuracy, Modifiability	[6][14][15]
Enterprise Resiliency	Flexibility, Adaptability, Vulnerability, Agility	[16]
Rule Validation, Relation Derivation, Quantitative Analysis	Alignment, Workload	[11]
Visibility Matrix	Hidden Structures	[17]

Table I
Analysis techniques

B. Visualization

Visualization deals with the depiction of information using a visual metaphor. Lankhorst [9] emphasizes the distinction between a view and a visualization, the former being a query of the enterprise model, and a model by its own rights, and the latter a visual representation of a model (or view). These views, or better, diagrams of these views, are artifacts whose primary function is bridging the language gap between stakeholders of different knowledge domains, e.g. people with business or technological backgrounds, as well as enabling detailed analysis to a selection of model elements.

While visualizations may have an arbitrary level of detail, we are interested in Overview Visualizations [8], which are a visual representation of the totality of elements of an Enterprise Model, as well as the relations between these elements. These visualizations, oriented to enterprise architects, are useful to discover overall patterns in the architecture, and facilitate topological (i.e. structural) analysis of an Enterprise Model. Their background is similar to *decision support viewpoints* such as Landscape Maps from van der Torre et al., that "create high-level, coherent overviews of enterprise architectures, providing the 'big picture' required by decision makers" and are constructed by analysis routines [23]. The value of these Overview Visualizations lies in their ease of detecting recurring structures (patterns) and symptoms or anomalies (anti-patterns), thus performing a *diagnosis* of the architecture, as well as the possibility

[1]Throughout this paper we will consider an Enterprise Model as a representation of the information in all aggregate artifacts that describe an enterprise [8].

of answering concrete questions and visualize quantitative measures with the use of visual attributes, e.g. size, position and color.

In previous work [8], we asked ourselves which techniques are useful to generate these Overview Visualizations. Results from this evaluation suggested that some of them favor convergent analysis (i.e. answering concrete questions), e.g. graph visualizations, while the strength of other techniques lies in the interactive exploration of the hierarchy of the model (e.g. treemaps). Also, these techniques can be combined for more effective results.

III. The PRIMROSe Framework

As we have discussed, Enterprise Architecture Analysis adds new knowledge to the Enterprise Model. For instance, analysis techniques may insert attributes to relationships, or even select some of them, e.g. for finding shortest paths. This makes sense if we consider that working with relationships is as important as working with elements of the model [8].

However, current analysis methods offered by EA tools are part of their application code, which explains the difficulty of providing new methods in most of these tools, as well as applying them to different metamodels. Thus, our first goal is to separate the activities of *modeling*, i.e. describing a current or future state of a subject, and *analyzing*, i.e. adding new facts to the model, in an automated fashion.

This means that we use a **Conceptual Metamodel** M_C^2 for capturing the initial facts of the enterprise, and an **Analysis Metamodel** M_A^2, which is an extended version of M_C^2, for deriving new facts.

Concerning model-based analysis, we can identify two courses of action, if we make explicit or implicit the use of this Analysis Metamodel:

The main idea behind the PRIMROSe framework[2], described in [24], is to use graph structures to perform different kinds of analysis, supported by flexible Overview Visualizations. A graph-based framework can take advantage of several graph properties, and enrich the model with graph transformations.

In this section, we will offer a quick overview of the process behind this framework.

A. A Pipeline of Pipelines

Seeing EA Analysis as a transformation of architectural data into useful information that serves as a basis for bringing and assessment or concept [8], PRIMROSe approaches analysis by a series of transformation steps under a pipeline metaphor (see Fig. 1), inspired by the Visualization Pipeline described by Chi and Riedl [25]. There are five managed stages, each with an input and output, and an internal pipeline that transforms the input. Interaction inside each stage defines how the transformation is made.

[2]PRIMROSe stands for enterPRIse Model gRaphical Overview analySis

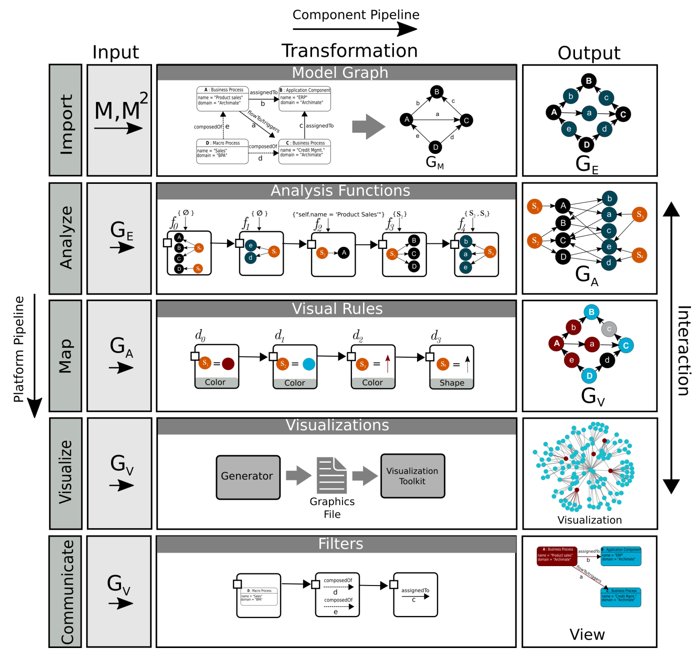

Figure 1. Overview of PRIMROSe.

*1) **Import**:* The process begins with an Enterprise Model M and its corresponding metamodel M^2 received as input. The first transformation is from M to a *model graph* G_M:

> **Model Graph:** It is a directed graph G_M with a set of vertices and a set of edges. Each vertex references one element of the original EM, and each edge is a relationship between the corresponding pair of elements.

Then, G_M is transformed into an *expanded graph* G_E, where each edge in the model graph is replaced by an incoming edge, a vertex, and an outgoing edge. This transformation is an *homeomorphism* of G_M, that means, it contains exactly the same information as G_M, and its topology is preserved.

> **Expanded Graph:** It is a directed and ***bipartite*** graph G_E, where its vertex set is the union of the vertices of G_M and its edges, transformed into vertices.

With G_E we can insert new properties to the relations of the model, as well as identify each one in an unique manner.

*2) **Analyze**:* As we have only transformed its structure, G_E contains the same information as M. In order to obtain additional knowledge from the model, we introduce the *Analysis Graph* G_A:

> **Analysis Graph:** A directed graph G_A, where its vertex set is the union of the vertices in G_E and a set S, that represents new vertices that are not present in G_E, and they are called *Selectors*.

As described in Section II-A, analysis techniques either a)identify groups of elements and relations that meet certain criteria, or b)add new facts to the model. To cover both aspects, G_A has a set S of additional vertices, *selectors*, that point to existing vertices and relations.

Selector: A vertex in G_A that is not present in G_E, but has edges that point to vertices of G_E.

Selectors are added by means of the application of functions that operate over Analysis Graphs. These functions, which should be specifically defined depending on the kind of analysis been performed, can be of two types: Analysis Functions, and Decorator Functions.

Analysis Function: Is a graph rewriting function $f_A : G_A \times P \rightarrow G_A$, with $P = \{p_0, p_1, ..., p_n\}$ the set of parameters. These functions have an unique identifier, and can be either 1)Generic, i.e. can be applied to any Enterprise Model, or 2)Metamodel specific.

Decorator Function: An Analysis Function f_D that produces a graph with the same vertices and edges as the original one, but complements the vertices with additional attributes.

*3) **Map**:* After all the Analysis Pipeline (see Fig. 1) is executed, we have G_A with all the additional facts added, and affected nodes pointed by selectors. Now we have to translate this data into visual information. To do this, we use *Visual Rules*, that are pairs of a Visual Attribute (e.g. color, shape, position) and a function that returns the value of such attribute for a given vertex.

Visual Rule: A rule $R_i = \{A, f_R\}$ is a pair of A a Visual Attribute, and $f_R : V \rightarrow X$ a function, where $V \in V(G_A)$ and X is a value of A.

For instance, a Visual Rule returns the value 'red' of the visual attribute **color** if an attribute (e.g. throughput) of a vertex is greater than 0.5, or 'green' otherwise. In order to apply multiple rules to a set of vertices pointed by a selector, we use a *Visual Decorator*:

Visual Decorator: A function $f_V : G_A \times S_i \times R \rightarrow G_A$, with S_i a given selector, that removes S_i and its associated edges, and applies a set of visual rules $R = \{R_0, ..., R_n\}$ to all target nodes of such edges.

This function effectively removes the selector, and enriches the selected nodes with visual information. In the end of the *Mapping* Pipeline, when there are no selectors left, we should have a *Visual Graph* G_V:

Visual Graph: It is a directed graph G_V, is an isomorphism of G_E, whose nodes have been enriched with visual properties.

*4) **Visualize**:* Having G_V with all the information needed to visualize the analyzed model, we can generate all the necessary graphic files of a given format (e.g. GraphML, JSON, DOT) required by a particular graphical toolkit. Its result is a Visualization that the architect uses to propose

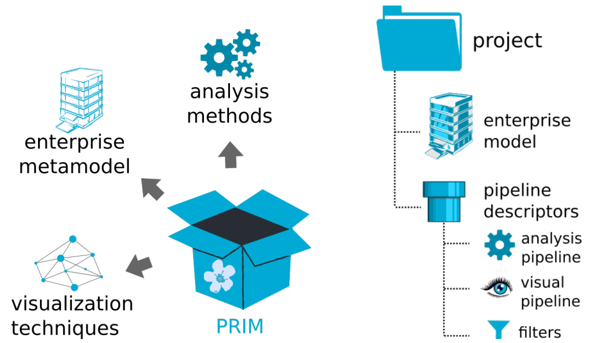

Figure 2. Structure of a PRIM module and project structure.

and validate (or deny) hypotheses in a sense-making loop with two types of interaction: 1) View Operators, such as zoom, pan, select and navigate, that do not alter the visualization, and 2) Data Operators, that control the flow of the global pipeline and modify the parameters in the internal pipelines of any previous stage, effectively modifying the Visualization.

This step will be explained in more detail on Section IV.

*5) **Communicate**:* Finally, as the architect finds some evidence that supports his suspicions, he would like to focus on the relevant elements and relations, and communicate his assessments to decision makers. For this purpose we use *Filters*:

Filter: A function $f_F : G_V \times Pr \rightarrow G_V$, that removes vertices (and their respective edges) of G_V that satisfy a predicate Pr.

These filters are applied sequentially in the Filters Pipeline, and the resulting graph is exported to a model M', which is a View of M.

IV. Modular EA Visual Analysis

One of the issues around analysis is the variety and breadth of methods, as described in Section II. On the other hand, we have also several techniques for visualizing analysis results. Thus, a repository for both aspects is critical, in order to offer a coherent tool set for visual analysis. Taking into account the dependence of analysis techniques to a given metamodel, we can package both of these types of techniques, as well as the metamodel, in an isolated deployment unit, which we will call a **PRIM**.

These packaging components can be deployed and managed, supported by the generic graph-based capabilities of the PRIMROSe framework. At glance, PRIM modules (see Fig. 2) are a collection of 1)Analysis methods for a given metamodel, and 2)Abstract Visualization techniques that can display the results of analysis under some visual method (e.g. graph, hierarchical, or even 3D visualizations).

The advantage of this approach is that we can have a general platform that manages the visual analysis process,

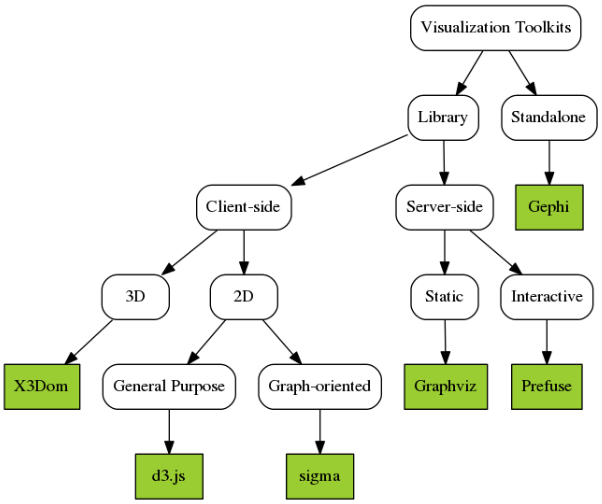

Figure 3. Types of visualization toolkits, with an example for each category.

while domain-specific techniques are encapsulated in a separate module. This offers the possibility of reusing visual and analytic capabilities among modules, as well as a lower implementation effort for these capabilities.

A. Metamodel

The first element is the Enterprise Metamodel M^2 that describes valid Enterprise Models for this PRIM, along with a transformation function that translates a concrete Enterprise Model M into a Model Graph G_M.

B. Analysis Methods

In Section III we introduced *Analysis Functions* as the way for adding new facts to the Enterprise Model. These functions need to be developed in terms of transformations of the Analysis Graph G_A, and each one represents a method of analysis.

This development is a one-time effort, and after being programmed and tested, they are ready to be used and composed. These functions can be applied to a *family* of models, that is, any model that conforms to the Enterprise Metamodel M^2, so we are assured that these functions have all the necessary information when executed.

Inside a PRIM, each Analysis Method provides an analysis function and a set of external libraries used by the function.

C. Visualization Techniques

A *visualization toolkit* is a set of tools or libraries that depict information under a visual metaphor. In order to visualize the results of the analysis, the user has two options:

1) Standalone toolkits: For immediate results, we can use a *standalone* (i.e. external) graph analysis tool, such as Gephi [26] or Cytoscape [27]. In this case, the graph is exported on a specific format.

2) Graphical libraries: We can also develop custom visualizations with the use of graphical libraries, offering more flexibility and control over what is displayed. In a distributed architecture, we have server-side (e.g. Graphviz [28] and Prefuse [29]) or client-side (e.g. d3.js [30], Sigma js [31], or X3DOM [32]) rendering of the visualization. Some of these toolkits provide interactive operations, while others generate static images.

What both types of toolkits have in common is that we need to translate the Visual Graph G_V into a format that is understood by the toolkit. Thus, the platform requires a transformation function from G_V to a concrete graph format (e.g. GEXF, JSON or GraphML) that is understood by the visualization toolkit.

With all of this in mind, **Visualization Technique** inside a PRIM contains:

- A formatting algorithm, that translates G_V into a graph format expected by the toolkit.
- A mapping algorithm for the representation of each element of the model with certain visual properties that differentiate them *(optional)*.
- A layout algorithm that is in charge of the placement of elements in the visualization *(optional)*.
- A set of interactive controls of the visualization *(optional)*.
- A library that renders the visualization *(optional)*.

Optional elements are necessary when we are using custom visualization techniques.

D. Projects

There is another level of detail in order to analyze a concrete Enterprise Model M. We introduce the notion of *projects* (see Fig. 2) to describe *what* type of analyses we would like to perform to M, *how* to visualize results from this analysis, and *which* results we would like to communicate. For this reason, a visual analysis project, which is associated to a given PRIM, has the model M as input, as well as a descriptor of the *analysis pipeline* to be executed, and a descriptor for the Visual Rules and Visual Decorators to be applied for a concrete Visualization Technique, and a set of filters to derive views. To do this, we will introduce three new concepts to our framework:

Operation: An instance of an Analysis Function or Decorator Function, each with an unique identifier and the values for the required parameters of a function.

Analysis Pipeline: An ordered set of Operations that transform G_E into G_A.

Visual Pipeline: An ordered set of Visual Rules and Visual Decorators that transform G_A into G_V.

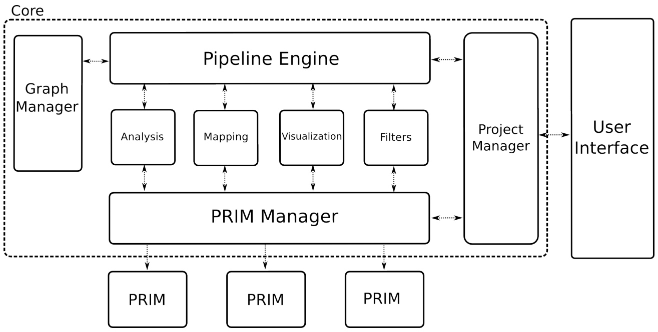

Figure 4. Architecture to support PRIM modules.

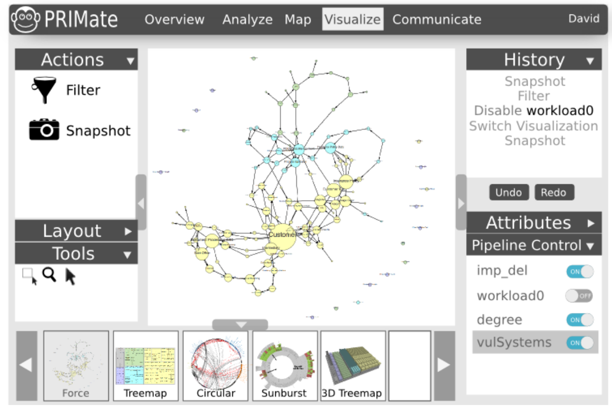

Figure 5. Interface design for the *Visualizate* stage.

E. Architecture

In order to support the conceptual framework described in previous sections, we propose a high-level architecture (see Fig. 4) with a set of core components and plug-able PRIM modules. This *Core* zone contains components that are the generic part of the PRIMROSe platform:

1) Graph Manager: Contains the graph structures G_M, G_E, G_A, and G_V, that can be managed by a specialized persistence component, e.g. a Graph Database. Also, this component is responsible of the import of an Enterprise Model, and its transformation into G_M, as well as the transformation of G_V to a view.

2) Pipeline Engine: Maintains the flow of control of the global pipeline (vertical direction of Fig. 1), as well as the internal pipeline of each stage (horizontal direction of Fig. 1). Thus, this component is responsible of:

- Obtaining G_A by executing the Analysis Pipeline, with the help of the Analysis component.
- Obtaining G_V by executing the Visual Pipeline, with the aid of the Mapping component.
- Generating the respective graphic files to obtain a Visualization.
- Running the set of filters and obtaining views.

Each of the above components obtain their respective artifacts (e.g. Analysis Functions and Visualization Techniques) from the PRIM manager.

3) PRIM Manager: Deploys and manages the PRIM instances, and serves as intermediary between a PRIM (including its respective projects) and the managers of each stage.

4) User Interface: Displays the visualizations, and controls the flow of the pipelines and their parameters through interaction. It also contains the respective editors for projects, analysis pipeline, visual pipeline, and filters.

V. Case Study and Implementation

In this section we would like to illustrate the subjects of this paper with a PRIM module for ArchiMate Visual Analysis. This includes the implementation of the PRIM, as well as the creation of an analysis project based on an Enterprise Model of a case study.

A. PRIMate: A PRIMROSe module

As described previously, this PRIM is an extension of the generic platform, PRIMROSe, and contains an Enterprise Metamodel, as well as catalogs of Analysis Methods and Visualization Techniques. Our goal is to provide a workbench that assists the user on the different stages of the global pipeline shown in Fig. 1. For instance, in the *Visualize* stage, we would like to inspect the results of the analysis using different Visualization Techniques (see Fig. 5).

Thus, in this section we will describe the steps for creating a new PRIM:

1) Enterprise Metamodel: As a first step, we include the ArchiMate metamodel of *Archi* [34], an Open Source ArchiMate modeling tool, as well as a Transformation Function that receives as input a valid *Archi* model, and returns a Model Graph G_M.

2) Analysis Methods: We developed two simple Analysis Methods, written in Java, that are capable of processing the Analysis Graph:

Domain Finder: This function adds selectors that represent ArchiMate layers, each one pointing to elements and relationships belonging to a given layer, namely Business, Application, Technology and Motivation. This information is extracted from the type of each element and/or relationship in the metamodel.

Degree Calculator: A Decorator Function that inserts weights to elements of the Enterprise Model, based on their incoming/outgoing relationships.

3) Visualization Techniques: We included three custom visualization techniques in the PRIM: a) a force-directed graph technique, developed with d3.js, and two hierarchical techniques: b) Treemap, also using d3.js, and c) a 3D Treemap, using the X3DOM library. For the first technique,

G_V is transformed into a serialized JSON string that contains a graph. In the other two cases, we calculated the breadth tree of the graph, and translated it to the JSON format.

Each technique also has its respective layout and mapping algorithms.

B. ArchiSurance: An Analysis Project for PRIMate

The ArchiSurance case study is an example of a fictitious insurance company, and has been widely used to illustrate the use of the ArchiMate language, as well as an application of the TOGAF Enterprise Architecture Framework. As described by the authors in [33], the case study concerns the insurance company ArchiSurance, which has been formed as the result of a merger of three previously independent companies, and describes the baseline architecture of the company and then a number of change scenarios.

Using this case study, we created an *Analysis Project* with the following elements:

1) Enterprise Model: As a first step we imported the ArchiSurance Enterprise Model (conveniently bundled within Archi). Using the transformation function inside the PRIM, the PRIMROSe platform is in charge of transforming this model into the different graph structures.

2) Pipeline Descriptors: The Analysis Pipeline is a very simple one, with the Domain Finder and the Degree Calculator analysis methods, which are applied sequentially. Then, we created a Visual Pipeline that associated the Selectors that were created with the Domain Finder function –ArchiMate layers– to their respective colors: yellow, blue, green, and violet. Grey color was used to denote integrating elements such as Folders. Finally, we associated the size of each element of the visualization to the weight inserted by Degree Calculator. Filters are out of the scope of this case study.

3) Visualizations: Fig. 6, a graph visualization of the model, allows the proposition of several insights in the analysis:

- We can identify the connection between ArchiMate layers through the color of the different elements.
- We can also identify the most important elements of the model by their size.
- It is clear that some elements are *unaligned*, i.e. we can see some isolated elements on the periphery of the graph, that have no relationships with any other element of the model.

On the other hand, we can take advantage of hierarchical Visualization Techniques by calculating the breadth tree of the visual graph. The treemap visualization –Fig. 7–, where each element of the model corresponds to a rectangular compartment, is useful to discover clusters of elements, containment relationships, as well the distribution of Archimate layers in the model. The 3D Treemap –Fig. 8– offers a more thorough navigation of the hierarchy of the model, where height defines the level of the breadth tree.

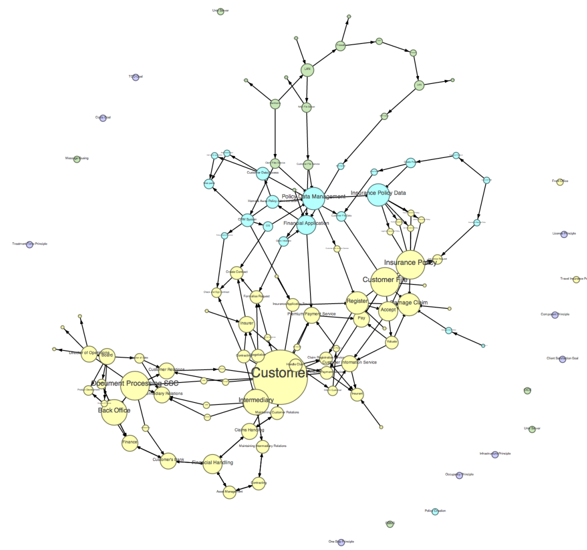

Figure 6. Graph visualization for the ArchiSurance model.

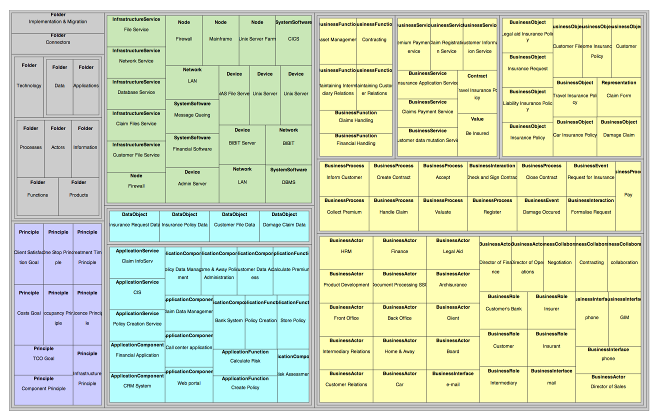

Figure 7. Treemap visualization for the ArchiSurance model.

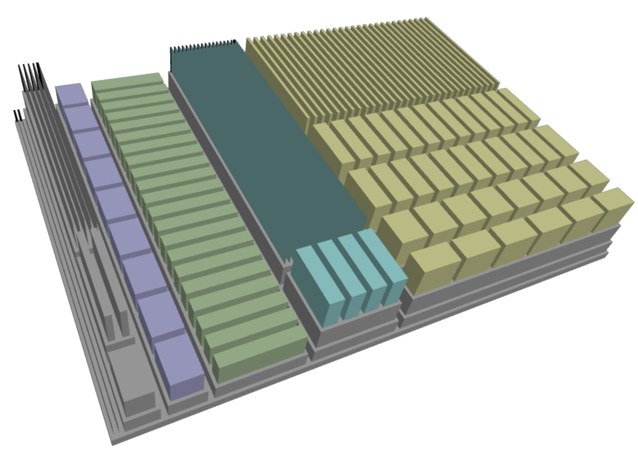

Figure 8. 3D visualization for the ArchiSurance model.

VI. Related Work

A. Modeling

Leeuwen et al. [35] recognize the integrating role of Enterprise Architecture, and propose an approach for the combination of heterogeneous models between different EA modeling tools, based on a common workbench and an integrating language -ArchiMate-, supported by the viewpoints defined in this language. Despite amalgamating only modeling capabilities, their work shows the importance of combining partial views to provide an overview of the Enterprise Model.

Also is worth mentioning the ADOxx [36] metamodeling platform[3], used for implementing editors that use an explicit modeling method that is defined by the user. Development of such editors includes not only the definition of an Enterprise Metamodel and a set of rules, but also the formulation of algorithms that support the different activities that are part of a modeling method.

ADOxx approaches the visualization of models by using a domain-specific language, *GRAPHREP*, that represents the graphical notation of the concepts of the metamodel, and includes simple rules that change its visualization based on attribute values. However, GRAPHREP describes just a part of a visualization, its representation, so aspects such as layout algorithms and the use of custom visualization techniques and toolkits are out of the scope of the ADOxx Visualization Core. Also,we have to take into account that in order to analyze an Enterprise Model, it must be developed first with the ADOxx editor.

B. Analysis

Sunkle et al. suggest that EA Analysis can be tackled by the generation of ontologies starting from EA modeling languages. While their scope is just Change Impact Analysis, they exploit the graph structure of these ontologies, and leverage analysis with the use of ontology tools and inference rule engines.

While PRIMROSe also supports the exporting of a model graph to other tools, the strength of our approach lies in the concatenation of different analysis methods. For instance, Lagerström et al. [17] use a structural approach for EA Analysis in order to uncover new facts from an Enterprise Model by extracting its 'hidden structure' by using a Visibility Matrix that allows to measure the flow and local clusters of the architecture. We are confident that analysis methods such as this one can be implemented in PRIMROSe using the methodology described in Section V.

C. Visual Analysis

Roth et al. and Hauder et al. [37], [38] approximate EA model Visual Analysis by the generation of views based on a pattern match between stakeholder needs and their access rights to information. They make use of various visualization techniques, where analytical facts can be displayed by the use of wizards and queries that allow the semantic enhancement of views.

Finally, Hipp et al. [39] propose the niPRO framework for the visual analysis of process models. While their scope is Business Process information, this framework makes a comprehensive study on how to visualize relevant and adequate process information, and goes from extracting relevant facts from several data sources, to the visualization of extracted knowledge from this model by the definition of a navigation space with several dimensions, resulting in diverse viewpoints depending on information needs.

VII. Conclusion

As the title of this paper suggests, we are describing an integration framework for two core functions of Enterprise Architecture: Analysis and Visualization. We focus on these functions for several reasons:

- The research community is mature enough; we already can describe the structure of an organization under several perspectives, and to the level of detail that we want. Having a formal description of an enterprise, we open the gates to new opportunities and challenges, focusing on how can we make sense of this structured knowledge.
- We need evidence to support our claims and suggest courses of action that drive enterprises to a desired result. With visualizations, we can point to the facts directly, providing the rationale needed for giving an assessment.
- Despite being critical for an architect, analytical skills are currently obtained by experience, given the limited and non-integrated support offered by most EA tools. In order to overcome this issue, and despite several efforts that have been made, there is yet no holistic approach for analysis, and useful techniques are scattered in the ever-growing tool market and research community.

Having new methods of analyzing enterprises on a bigger scale (where thousands of elements and relationships are involved), we need to approach analysis in a different way that we have been approaching modeling. With this in mind, we propose a framework and architecture that enables a greater support for several analysis tasks, based on interactive Overview Visualizations and an explicit and streamlined analysis process that does not depend on the modeling tool and is flexible enough for its use with any EA language or metamodel. We think this approach has a lot of potential in the composition and easy formulation of existing and innovative analysis methods, as well as a playground for the development of new ways of visualizing enterprises.

The strength of our graph approach is that it allows the formulation of analysis methods that can be applied sequentially in a pipeline. Each executed method leaves a

[3] http://www.adoxx.org

set of **Selectors** in the graph that allow the custom mapping between these additional facts found in the model, and visual attributes of a visualization. We can identify several other advantages of modularizing the capacities offered by this analysis tool: First, it is possible to analyze an Enterprise Model produced by any modeling tool, given a way to translate it to a graph structure. Also, we can easily replicate the plethora of automated analysis methods available, as well as reuse them with other metamodels. On the other hand, we can propose new visualization techniques, as we are not limited to a specific visualization toolkit. Finally, by offering a way to explore the model and analyze it incrementally, we can focus on bigger questions that require an architect and his analysis skills.

While the evaluation part of this paper is based on a fictional case study, our plan is to extend the application of PRIMate on a bigger scale. This requires a more detailed case study in a real environment, as well as a detailed catalog of analysis methods.

REFERENCES

[1] C. Strano and Q. Rehmani, "The role of the enterprise architect," *Information Systems and e-Business Management*, vol. 5, no. 4, pp. 379–396, 2007. [Online]. Available: http://dx.doi.org/10.1007/s10257-007-0053-1

[2] M. Op t Land, E. Proper, M. Waage, J. Cloo, and C. Steghuis, "The enterprise architect," in *Enterprise Architecture*, ser. The Enterprise Engineering Series. Springer Berlin Heidelberg, 2009, pp. 113–125. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-85232-2_6

[3] V. Clerc, P. Lago, and H. Vliet, "The architects mindset," in *Software Architectures, Components, and Applications*, ser. Lecture Notes in Computer Science, S. Overhage, C. Szyperski, R. Reussner, and J. Stafford, Eds. Springer Berlin Heidelberg, 2007, vol. 4880, pp. 231–249. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-77619-2_14

[4] B. van der Raadt, S. Schouten, and H. Vliet, "Stakeholder perception of enterprise architecture," in *Software Architecture*, ser. Lecture Notes in Computer Science, R. Morrison, D. Balasubramaniam, and K. Falkner, Eds. Springer Berlin Heidelberg, 2008, vol. 5292, pp. 19–34. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-88030-1_4

[5] J. Gotze, "The changing role of the enterprise architect," in *Enterprise Distributed Object Computing Conference Workshops (EDOCW), 2013 17th IEEE International*, Sept 2013, pp. 319–326.

[6] P. Narman, P. Johnson, M. Ekstedt, M. Chenine, and J. Konig, "Enterprise architecture analysis for data accuracy assessments," in *Enterprise Distributed Object Computing Conference, 2009. EDOC '09. IEEE International*, Sept 2009, pp. 24–33.

[7] D. Naranjo, M. Sanchez, and J. Villalobos, "Visual analysis of enterprise models," in *Enterprise Distributed Object Computing Conference Workshops (EDOCW), 2012 IEEE 16th International*, Sept 2012, pp. 19–28.

[8] D. Naranjo, M. E. Sánchez, and J. Villalobos, "Connecting the dots: Examining visualization techniques for enterprise architecture model analysis," in *PoEM (Short Papers)*, ser. CEUR Workshop Proceedings, J. Grabis, M. Kirikova, J. Zdravkovic, and J. Stirna, Eds., vol. 1023. CEUR-WS.org, 2013, pp. 29–38.

[9] M. Lankhorst, *Enterprise Architecture at Work: Modelling, Communication and Analysis*. Springer, Dec. 2005.

[10] S. Buckl, F. Matthes, and C. M. Schweda, "Classifying enterprise architecture analysis approaches," in *Enterprise Interoperability*, ser. Lecture Notes in Business Information Processing, R. Poler, M. Sinderen, R. Sanchis, W. Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw, and C. Szyperski, Eds. Springer Berlin Heidelberg, 2009, vol. 38, pp. 66–79.

[11] A. Ramos, P. Gómez, M. Sánchez, and J. Villalobos, "Automated enterprise-level analysis of archimate models," in *XIX International Conference in Exploring Modelling Methods for Systems Analysis and Design (EMMSAD-14)*, 2014.

[12] M.-E. Iacob and H. Jonkers, "Quantitative analysis of enterprise architectures," in *Interoperability of Enterprise Software and Applications*, D. Konstantas, J.-P. Bourrières, M. Léonard, and N. Boudjlida, Eds. Springer London, 2006, pp. 239–252.

[13] F. De Boer, M. Bonsangue, L. P. J. Groenewegen, A. Stam, S. Stevens, and L. van der Torre, "Change impact analysis of enterprise architectures," in *Information Reuse and Integration, Conf, 2005. IRI -2005 IEEE International Conference on.*, Aug 2005, pp. 177–181.

[14] P. Narman, M. Buschle, J. Konig, and P. Johnson, "Hybrid probabilistic relational models for system quality analysis," in *Enterprise Distributed Object Computing Conference (EDOC), 2010 14th IEEE International*, Oct 2010, pp. 57–66.

[15] R. Lagerström, P. Johnson, and D. Höök, "Architecture analysis of enterprise systems modifiability models, analysis, and validation," *Journal of Systems and Software*, vol. 83, no. 8, pp. 1387 – 1403, 2010, performance Evaluation and Optimization of Ubiquitous Computing and Networked Systems. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121210000518

[16] O. Erol, B. J. Sauser, and M. Mansouri, "A framework for investigation into extended enterprise resilience," *Enterprise Information Systems*, vol. 4, no. 2, pp. 111–136, 2010.

[17] R. Lagerström, C. Baldwin, A. MacCormack, and D. Dreyfus, "Visualizing and measuring enterprise architecture: An exploratory biopharma case," in *The Practice of Enterprise Modeling*, ser. Lecture Notes in Business Information Processing, J. Grabis, M. Kirikova, J. Zdravkovic, and J. Stirna, Eds. Springer Berlin Heidelberg, 2013, vol. 165, pp. 9–23.

[18] M. Mantyla, J. Vanhanen, and C. Lassenius, "Bad smells - humans as code critics," in *Software Maintenance, 2004. Proceedings. 20th IEEE International Conference on*, Sept 2004, pp. 399–408.

[19] S. Olbrich, D. S. Cruzes, V. Basili, and N. Zazworka, "The evolution and impact of code smells: A case study of two open source systems," in *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*, ser. ESEM '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 390–400.

[20] F. Khomh, M. Di Penta, and Y. Gueheneuc, "An exploratory study of the impact of code smells on software change-proneness," in *Reverse Engineering, 2009. WCRE '09. 16th Working Conference on*, Oct 2009, pp. 75–84.

[21] M. Mantyla, J. Vanhanen, and C. Lassenius, "A taxonomy and an initial empirical study of bad smells in code," in *Software Maintenance, 2003. ICSM 2003. Proceedings. International Conference on*, Sept 2003, pp. 381–384.

[22] S. Buckl, A. M. Ernst, F. Matthes, and C. M. Schweda, "How to make your enterprise architecture management endeavor fail!" in *Proceedings of the 16th Conference on Pattern Languages of Programs*, ser. PLoP '09. New York, NY, USA: ACM, 2010, pp. 2:1–2:7.

[23] L. van der Torre, M. M. Lankhorst, H. Doest, J. T. Campschroer, and F. Arbab, "Landscape maps for enterprise architectures," in *Advanced Information Systems Engineering*, ser. Lecture Notes in Computer Science, E. Dubois and K. Pohl, Eds. Springer Berlin Heidelberg, 2006, vol. 4001, pp. 351–366.

[24] D. Naranjo, M. Sanchez, and J. Villalobos, "Primrose: A tool for enterprise architecture analysis and diagnosis," in *ICEIS (3)*, S. Hammoudi, L. A. Maciaszek, J. Cordeiro, and J. L. G. Dietz, Eds. SciTePress, 2014.

[25] E. H.-h. Chi and J. Riedl, "An operator interaction framework for visualization systems," in *Proceedings of the 1998 IEEE Symposium on Information Visualization*, ser. INFOVIS '98. Washington, DC, USA: IEEE Computer Society, 1998, pp. 63–70.

[26] M. Bastian, S. Heymann, and M. Jacomy, "Gephi: An open source software for exploring and manipulating networks," in *Proceedings of the Third International Conference on Weblogs and Social Media (ICWSM'09)*. The AAAI Press, 2009.

[27] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker, "Cytoscape: a software environment for integrated models of biomolecular interaction networks," *Genome research*, vol. 13, no. 11, pp. 2498–2504, 2003.

[28] J. Ellson, E. R. Gansner, E. Koutsofios, S. C. North, and G. Woodhull, "Graphviz and dynagraph static and dynamic graph drawing tools," in *Graph Drawing Software*, ser. Mathematics and Visualization. Springer Berlin Heidelberg, 2004, pp. 127–148.

[29] J. Heer, S. K. Card, and J. A. Landay, "prefuse: a toolkit for interactive information visualization," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, ser. CHI '05. New York, NY, USA: ACM, 2005, pp. 421–430.

[30] M. Bostock, V. Ogievetsky, and J. Heer, "D3: Data-driven documents," *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*, 2011. [Online]. Available: http://vis.stanford.edu/papers/d3

[31] A. Jacomy, "Sigmajs," 2014, online; accessed on May 15, 2014. [Online]. Available: http://sigmajs.org

[32] J. Behr, P. Eschler, Y. Jung, and M. Zöllner, "X3dom: a dom-based html5/x3d integration model," in *Proceedings of the 14th International Conference on 3D Web Technology*. ACM, 2009, pp. 127–135.

[33] H. Jonkers, I. Band, and D. Quartel, "Archisurance case study," last retrieved on April 4, 2014. [Online]. Available: https://www2.opengroup.org/ogsys/catalog/y121

[34] P. Beauvoir, "Archi - the free archimate modelling tool," last retrieved on April 4, 2014. [Online]. Available: http://www.archimatetool.com

[35] D. van Leeuwen, H. W. L. ter Doest, and M. M. Lankhorst, "A tool integration workbench for enterprise architecture - integrating heterogeneous models and tools," in *ICEIS (3)*, 2004, pp. 470–478.

[36] H. Kühn, "The adoxx® metamodelling platform," in *Workshop on Methods as Plug-Ins for Meta-Modelling, Klagenfurt, Austria*, 2010.

[37] S. Roth, M. Hauder, M. Zec, A. Utz, and F. Matthes, "Empowering business users to analyze enterprise architectures: Structural model matching to configure visualizations," in *Enterprise Distributed Object Computing Conference Workshops (EDOCW), 2013 17th IEEE International*, Sept 2013, pp. 352–360.

[38] M. Hauder, S. Roth, S. Pigat, and F. Matthes, "A configurator for visual analysis of enterprise architectures," in *ACM/IEEE 16th International Conference on Model Driven Engineering Languages and Systems (MODELS 2013), Miami, USA*, 2013.

[39] M. Hipp, B. Mutschler, B. Michelberger, and M. Reichert, "A framework for the intelligent delivery and user-adequate visualization of process information," in *Proceedings of the 28th Annual ACM Symposium on Applied Computing*. ACM, 2013, pp. 1383–1390.