Characteristics of Enterprise Architecture Analyses

Julia Rauscher, Melanie Langermeier and Bernhard Bauer
Software Methodologies for Distributed Systems, University Augsburg, Augsburg, Germany
{rauscher, langermeier, bauer}@ds-lab.org

Keywords: Enterprise Architecture, Analysis, EA Analysis, Categorization, DSL.

Abstract: Enterprise Architecture Management (EAM) deals with the assessment and development of business processes and IT landscape of an organization. Analyses are an important mean in the EAM process. They support the understanding of the architecture and evaluate the current situation as well as possible future ones. In current literature exists various different approaches to EA analyses. Each pursues different goals and utilizes different techniques. We evaluated current literature about EA analyses and identified possible categories. Therefore we define requirements for an EA analysis and utilize them for a classification of the various approaches. We propose a two-dimensional classification approach. Technical categories cluster analyses according their procedure and utilized techniques. Functional categories cluster analyses according to their goals and outcome. To validate our categorization and the analysis requirements we develop a domain specific language, which is used to formalize the existing analysis approaches from literature.

1 INTRODUCTION

Analyses are one of the most important artifacts integrated in Enterprise Architecture Management (EAM) and are indispensable in the EAM cycle. The EA process contains five phases (Niemann, 2006): Document, Analyze, Plan, Act and Check. Thus, analysis is an essential part in order to create and implement future plans. It supports decision making through an evaluation of the current architecture as well as potential future scenarios (Sasa and Krisper, 2011). The result of analysis and planning actions is finally the creation of a target architecture. Those actions enable also planning, acting, controlling and documenting through all layers.

The creation of an Enterprise Architecture (EA) model is time and cost consuming. Therefore, support for decision making and planning generates value and increases the acceptance of the EA initiative in an organization (Lankhorst, 2013). Thus, analysis support is essential in order to generate value from an EA model. The execution of an analysis decomposes the analyzed object in its components. Those single elements are examined and evaluated as well as the relationships and interactions between them. Applying existing analysis on established models is expensive, since the corresponding meta models typically require some adaptions (Langermeier et al., 2014). This makes reuse of existing solutions and research findings hard.

In current literature a great variety of different analysis possibilities are described. They rely on different technologies, like probability networks (Närman et al., 2008), ontologies (Sunkle et al., 2013) or expert interviews (Kazienko et al., 2011), have different preconditions and provide different kind of results. E.g. preconditions can be required properties for model elements or specific data structures. Typical results are quantitative ones like an overall metric for the architecture, measures for specific architecture elements, but also a determined set of elements. Every analysis supports a different goal and thus, for a sound evaluation of the architecture different kinds of analyses are required.

In this paper we want to provide a categorization of existing EA analyses from literature. The main goal of the categorization is to create a possibility to conduct analyses organized and controlled, to create new analyses and to choose the best suited analysis depending on the goal and requested technique. We study existing analyses regarding their requirements for execution and their provided result. This provides us a sound overview of analysis approaches in the context of EA and of the issues they address. Based on the identified similarities we establish a categorization of the approaches. Once according to their functional dimension, and once according to their technical dimension. The resulting categories with their characteristics are evaluated through the establishment of a Domain Specific Language (DSL).

Rauscher J., Langermeier M. and Bauer B.
Characteristics of Enterprise Architecture Analyses.
DOI: 10.5220/0006222701040113
In *Proceedings of the Sixth International Symposium on Business Modeling and Software Design (BMSD 2016)*, pages 104-113
ISBN: 978-989-758-190-8

This allows us to formalize them and validate their correctness by modeling existing analyses. Additionally such a language allows us to make the requirements of an analysis visible in a structural way. The calculated outcome as well as the preconditions in order to execute the analysis are easily accessible.

The remaining paper is structured as followed. First we introduce foundations of EA analysis (section 2). Following we present in section 3 our approach for determining the analysis categories. The categories itself are also presented shortly with their main characteristics. The DSL to describe the analyses is introduced in section 4.1. Its application is shown exemplary for one category. Finally we discuss the categorization and the DSL (section 4.2).

2 ENTERPRISE ARCHITECTURE ANALYSIS

Architectures are used to describe the elements of a system and the relationships between them. The term also comprises the process of creation and maintenance, the architecture work (Lankhorst, 2013). Often used layers are the strategic layer to represent the organization's strategy with its goals, the business layer describing the business processes and products, the information layer with the information objects, and the application layer as well as the infrastructure layer describing the Soft- and Hardware components (Lankhorst, 2013; Winter and Fischer, 2006). Despite the examination of different layers the focus of an EA are the dependencies between layers, i.e. how business and IT relate to each other. Layers are dependent according to the Align-Enable-Principle. The lower layers are the foundation for the upper ones, and the upper ones adjust the lower ones (Winter and Fischer, 2006; Krcmar, 2015).

The main reasons of analyzing EA is to receive an overview of the whole architecture, its components and connections, and to analyze the as-is state (Langermeier et al., 2014). Furthermore weak points can be revealed, new advantages be discovered and various design alternatives be evaluated (Zia et al., 2011). The result of analysis activities is the development of a to-be architecture as well as improved decision making. The focus of every analysis depends on the analysis type. Additional the questions of what is feasible and what is desirable are crucial (Johnson et al., 2007). The process of analysis can be segmented in different phases and activities (Wan and Carlsson, 2012). We used the parts "system thinking", "modeling", "measuring", "satisfying", "comparing with requirements" and "comparing alternatives" in this work to identify characteristics of analysis categories.

To receive a basis for our work we conducted a detailed literature research (Rauscher, 2013; Rauscher, 2015). We exclusively chose analyses, which purely analyze EA and are not transferred from other topics. Hereof a pure EA analysis has the focus on collecting data and discovering the current state of an enterprise architecture in a quantitative or functional way to create a summary, alter the state or control different aspects. The goal of this selection was to create an overview of current EAM analyses and to receive approaches utilizable for a categorization (e.g. Della Bordella et al., 2011; Johnson et al., 2007; Razavi et al., 2011). We identified 105 EAM analyses which are roughly grouped into 40 EA analysis types in previous work (Rauscher, 2013). An analysis type describes analyses which have the same rough scope and are built independent on the realization method. The goals of contained analyses can differ significant. Examples of these types are 'Quality Analysis' (Närman et al., 2008), Requirements Analysis' (Aier et al., 2009) and 'Analysis of Costs' (Niemann, 2006). The different types of analyses which have been discovered in the literature research can be treated as a first categorization. However this categorization only makes raw statements about the rough purpose of the contained analyses. Although analysis of the same analysis type have the same field of interest their individual goals and implementations can differ. It's not detailed enough to derive characteristics and the covered analyses don't have to share them. Quality analyses, for example, can be conducted in various ways and can target different goals from quality of a whole system to maturity quality of a single artifact.

3 CATEGORIZATION

The huge amount of different approaches clarifies importance of EA analyses and coherence of a successful architecture. To ease the usage of analysis and get an understanding about current work we categorized them according their characteristics. We define characteristics of an EA analysis as all necessary steps and components of an analysis to accomplish its goal. The characteristics are a main part of the categorization because they are guaranteeing the accuracy of the conducted analysis and the achievement of the target. Figure 1 gives an overview over the categorization approach, which is described in detail in sub-section 3.1. The resulting categories are presented in sub-sections 3.3 and 3.4.

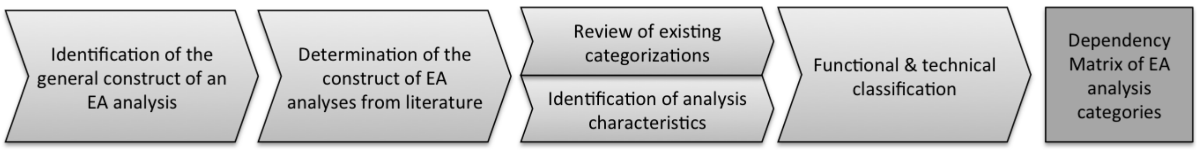
Identification of the general construct of an EA analysis
Determination of the construct of EA analyses from literature
Review of existing categorizations
Identification of analysis characteristics
Functional & technical classification
Dependency Matrix of EA analysis categories

Figure 1: Categorization approach

3.1 Categorization Approach

Preliminary work for the categorization includes the definition of a general understanding of an analysis to determine a general purpose construct. We could identify three main constructs of an EA analysis, which are used as foundation for the categorization and determination of characteristics. These are data *intake*, *processing* and *outcome*. Other parts vary per analysis. Based on these parts we determine the meaning and boundaries of a category: Analyses can be merged to a category if at least one of the three parts given above is similar. In an optimal condition all parts are equal, but this condition is usually not given. Therefore we classify different analyses to the same category if they have at least the same target or same processing technique.

After defining our basis we conduct literature research of current EA analysis (procedure see section 2). We determined the construct for each of the identified EA analysis approaches. Thereby we ensured that only paper with a high elaboration level are used. Because of missing details it's not possible to analyze rough approaches and to identify their construct and characteristics. For elaborated analyses with less detailed parts, we made necessary assumptions. In the case that an EA analysis approach is realized using another non-EA related analysis approach, this non-EA analysis is included too. This proceeding ensures the construction of a data basis with categorize-able analyses according to the general construct of intake, processing and outcome.

Based on the experiences made while identifying the construct of the EA analysis we can refine our categorization approach. Considering different existing kinds of categorization (Lankhorst, 2013; Buckl et al., 2009b) we conducted our approach with two main fields of categories: functional and technical. This decision brought the most advantages in comparison with other approaches because of the division in "How" (technical aspects) and "Why" (functional purpose). The additional distinction in architecture levels is not included in our approach because a plain allocation wouldn't be possible. Most analyses can be conducted in many levels or can only be performed by involving other levels. Through the new and detailed knowledge from the first evaluation of EA analysis constructs we introduce characteristics to ensure accuracy. As only properties and steps can show the components responsible for classification, we used these characteristic kinds to analyze the approaches again to receive detailed information of their single categories.

After this step the final categorization of the analyses was received based on our main idea of distinction between functional and technical. The business functions of every analysis are determined based on the concepts *purpose dependent division* ('Fundamental", "Main" and "Decision-oriented") and *activity dependent division* ("System thinking", "Modeling", "Measuring", "Satisfying", "Comparing with requirements" and "Comparing alternatives") (Wan and Carlsson, 2012). We used these concepts to analyze the identified analysis approaches according to their goals and activities. Thereby the functional categories have been determined by using a prepared template of aspects. This template consists of the analysis activities, the intermediate objectives and the main goal. After analyzing all approaches we identified 10 categories from classifying the various analyses goals. Attention should be paid to the fact of multiple classification. For example, a security analysis is able to analyze dependencies and requirements and therefore can be assigned to both functional categories.

After we completed the functional classification, we conducted the technical categories. This procedure was more detailed and complex because of the large variety of existing methods in EA analysis. Only analyses with the same method and same steps of goal attainment can form a technical category. This constraint is necessary to enable discovery of shared characteristics. The already mentioned template was altered for creating technical categories. The new focus lies on the constructs, methods, techniques (including single steps) and artifacts. First, rough technical categories have been determined based on the dimensions "quantitative", "analytic", "simulation" and "functional" (Lankhorst, 2013). After this prestage detailed categories were created. Each identified analysis approach passed through this procedure. In contrast to functional categories every analysis was assigned to one specific technical category. As final result we concluded with 17 technical categories.

Altogether 105 analysis approaches fulfilled our criteria and have been incorporated. Only 9 of them couldn't be classified. These ones were to specific and individual to create a category. For each analysis we identified exactly one technical category and at least one functional category. To create an overview of all possible combinations of categories three matrices were created. Two matrices represent the combination of the analysis approaches and the categories. Here we have to mention the features of both tables. The functional matrix has more possible combinations because analyses can achieve more targets simultaneously. However the technical matrix has only one combination per analysis. For example, Närman et al. (2008) is assigned to the functional categories **System Analyses**, **Attribute Analyses** and **Quality Analyses** and to the technical one **Bayesian Networks**. To provide an overview of the functional and technical combinations both matrices have been joint which resulted in a shared matrix (see figure 2). Thus, we get an overview of the realization techniques of a functional category and also the other way round for analyses and their used techniques. In the table an "x" represents that there is at least one analysis in current literature that was matched to both categories, the functional and the technical one.

3.2 Identification of Characteristics

As only properties and steps can show the components responsible for classification, we introduce characteristics to ensure accuracy. We define a characteristic as requirement, since an analysis can only be conducted target-aimed with all indispensable artifacts. Requirements support the achievement of goals and are used to identify hidden characteristics (Van Lamsweerde, 2001). Whereas properties can differ significantly, on some spots we had to choose the most elaborated one or to create a higher abstraction level. There are two types of characteristics: category specific ones and general characteristics. The second type includes a meta model and scenarios, determined at the beginning of an analysis. Another universal characteristic is the main goal. These characteristics have to be conducted for all analyses. Together they provide a high level of abstraction.

For the specific characteristics we distinguished five different kinds. The conducted kinds of characteristics are important for the identification of properties from technical analyses. Whereas functional categories have rough properties, technical categories have similar structure. We identified the following kind of characteristics: *Input*, *Conditions*, *Construct*, *Measurement*, and *Output*. The basis of an analysis is always represented in terms of *Input* data. In every case an architecture or scenarios, in form of an model, are needed to conduct the following steps and final measurements. Before the main part of an analysis can be performed, sometimes *Conditions* are needed. For example the possibility of succeeding must be given. Most of the *Conditions* are analysis independent and therefore can be seen as generally valid. The main part and procedure of an analysis is the *Construct*, containing all details of the procedure. It's required to conduct all details successfully to be able to finish the analysis. Examples are detailed steps, mathematical algorithms, relationship types and weighting of artifacts. To prove and measure the results and development, every analysis needs some kind of *Measurement*. This characteristic is responsible to witness the achievement of goals. Most of time a *Measurement* is proceed with scales, KPIs and metrics to control functional and non-functional goals (Davis, 1989). This characteristic can vary dependent on the analysis and its goals. As last characteristic kind the *Output* was identified. It includes the way of presentation and type of outcome such as percentage, graphics or matrices. This category is crucial because analyses within the same category should have the same *Output*. We used these characteristic kinds to analyze the approaches again to receive detailed information. Through the new and detailed knowledge we had to rearrange the analysis categories on necessary points. New identified characteristics have been verified on correctness and necessity. After this step the final categorization of the analyses was received.

3.3 Functional Categorization

In the following we present the categories of the functional classification. Therefore we will list them combined with an example of a contained analysis approach. The assignment of all identified analyses to the categories can be found in (Rauscher, 2015). **System Analyses** (e.g. (Närman et al., 2008)) check partial or holistic systems. Mostly time quality aspects and their optimizations are in the main focus. Analyses which are contained in this category are often also part of other functional categories because of possible sub-goals. Specific attributes and their values are analyzed by **Attribute Analyses** (e.g. (Razavi et al., 2011)). The observation and management of attributes is the focus such approaches. For instance the different states of attributes with changing input can be analyzed. Analyses which prove dependent connections are classified as **Dependencies Analyses** (e.g. (Saat, 2010)). The main goal of these approaches is the identification of dependencies in EAs and connections of single components to receive

Technical / Functional	Bayesian Networks	Business ENtities	PRM	Social Network	AHP	Time Evaluation	Tree	KPI and Matrices	Comparison	Views	Lifecycle	Ontology	EID	Weak Points	Matrices	Design	Structural	Other
System	x		x										x					
Attribute			x		x		x		x				x	x				
Dependencies	x		x				x		x		x	x			x			
Quality	x	x	x		x	x	x	x					x		x			x
Design				x					x			x			x	x	x	
Effects									x	x		x	x		x			
Requirements										x	x							
Financial								x						x	x			
Data													x					
Business Objects		x		x		x	x		x	x		x		x				
Other										x					x			x

Figure 2: Dependency matrix of the functional and technical categories.

an understanding of the whole architecture. **Quality Analyses** have the main focus on various quality questions regarding attributes, systems, architectures and other components and target subjective and measurable goals (e.g. (Närman et al., 2008)). This category is based on ISO 9126 and analyzes, for instance, maintainability, maturity and interoperability. Another category represents the analysis of architecture design (**Design Analyses**), examples are (Aier et al., 2011; Kazienko et al., 2011). Through receiving an overview on the architecture construct all possibilities of designs can be identified. Beside the analysis of holistic or partial design, business entities, procedures and components can be analyzed and used to optimize the architecture. All approaches which control impacts in architectures and actions are joint in **Effect Analyses** (e.g. (de Boer et al., 2005)). In contrast to **Dependencies Analyses** these approaches observe the direct impact and effects of changes in architecture elements. **Requirements Analyses** target all possible requirements to achieve states or goals (e.g. (Aier et al., 2009)). To accomplish special goals, different requirements are needed. Results are either specified values or features and entities. To identify costs and benefits **Financial Analyses** are used (e.g. (Niemann, 2006)). On top financial weak points and possible impacts can be discovered. These analyses present a measurement with mathematical calculations. Therefore key figures and metrics are able to evaluate outcomes. However receiving affected entities is a side effect of the results. Consequently **Financial Analyses** observe too high costs or uncertainty and hence are an indicator of necessary architecture and procedure changes. However **Data Analyses** cover all kinds of data (e.g. (Närman et al., 2009)). The focus lies on quality and accuracy, because data are crucial for successful EA operations. Finally the category **Business Object Analyses** was identified (e.g. (Della Bordella et al., 2011)). Business objects of every kind, e.g. operations, artifacts and entities, which are part of architecture procedures and construct are addressed here.

3.4 Technical Categories

In the following we describe the 17 technical categories. For the description we chose the most important and marked characteristic kinds (see section 3.2). Since nearly all technical categories require an architecture model, scenarios and goals as *Input* we won't mention it below. The first technical category represent analyses conducted with **Bayesian Networks** (e.g. (Närman et al., 2008)). Analyses of this category use this technique to analyze the quality. It is reused in other analyses as part of their procedures, e.g. PRM analyses. Firstly a model with Bayesian Networks is built in the main *Construct*, including all nodes and connections of the architecture. A node represents a variable with a conditional probability distribution. Therefore in the next step probabilities of attributes and the whole model can be calculated while creating matrices with discrete ranges. In conclusion this category has probability values as *Output* and can answer questions about the probability of an attribute's status. **Business Entities** are a method to receive artifacts and analyze quality (e.g. (Della Bordella et al., 2011)). As *Input* and *Conditions* the goal type, strategies and quality features must be determined. First step of the *Construct* detects advantages, operations and elements of strategies. As a second step, influencer and strategies are combined to observe the goals. For measurement the goal is quantified and evaluated. The *Output* contains valued strategies, quality values and identified operations and entities. **Probabilistic Relational Models (PRM)** contains 14 analyses and is therewith the most used technique (e.g. (Buschle et al., 2011)). For instance dependency and quality analyses can be conducted with PRM. *Conditions* require determinable goals and controllable attributes. As a prestep of the *Construct* connections are defined and uncertainties are formalized. Hereafter the PRM is used to calculate the conditional probability of all scenarios and of the dependencies and attributes. Therefore percentage of attributes, scenarios and uncertainty are in the *Output*. **Social Network** analyses differ deeply from the other categories (e.g. (Kazienko et al., 2011)). To conduct the analysis questionnaires and all available documents, like bills and connections are used. For the *Input* all available nodes (= entities) and connections are required. As *Construct* clusters are built and properties can be checked. Additional new entities and connections are found. For the *Measurement* a matrix with entities and factors is created for quantitative evaluation with factors or for identification of weak points. An overview of the whole architecture construct and its entities and connections on a social basis is found in the *Output*. **Analytic Hierarchy Process (AHP)** can be used for analyzing attributes and quality aspects and is one of the most elaborated EA analysis techniques (e.g. (Razavi et al., 2011)). *Conditions* are experts knowledge used for weighing as well as definable quality attributes and level of success. The *Construct* determines quality attributes, their criteria, subcriteria, layers and level of importance. To evaluate the attributes, criteria and layers pairwise comparisons are conducted by experts. For every part which passes the comparison process a prioritized vector is created. Afterwards the same proceeding is conducted for all scenarios. This results in an *Output* with prioritized lists of quality attributes and scenarios to determine the best scenario dependent on the attribute. The method of **Time Evaluation** (e.g. (Lankhorst, 2013)) calculates times and observes the quality of business entities and operations. As additional information the trigger and arrival times are required for the *Input*. Rules (*Condition*) are necessary to cut the architecture in views and receive five perspectives with single time measurements. In this category *Construct* and *Measurement* are combined. Within the views the specific times are calculated. Examples are 'Costumer View' and 'Process View' with 'Processing Time' and 'Response Time'. In a last step, all calculated times are summed up to a total time. **Trees** are used to analyze and identify dependencies, coherences and quality features. The *Output* of those analyses delivers the probability of an occurring failure or specific quality, depending on the purpose. In the beginning of the *Construct* the goals, entities and relations are defined. Afterwards a fault tree is built using Bayesian Networks, containing all steps or events required for the procedure of operation. For every component of this tree a conditional probability matrix is created to receive the probability of failures or quality property (Närman et al., 2011). The usage of **KPI** (Key Performance Indicator) is used in most analyses with quantitative measurement. Because of the high variety of contained analyses, a high level of abstraction was used. The goal has to be definable definite, measurable, agreed, realistic and time-bound. The *Construct* starts with identifying all artifacts which should be analyzed and determining the matching KPIs. The analysis evaluates the artifacts and compares the values with metrics for *Measurement*. The result can present achieved goals, unsatisfied quality issues and the financial situation (Niemann, 2006). **Comparison** is a simple but powerful method (e.g. (de Boer et al., 2005)). Next to whole alternatives, also single scenarios, processes, attributes and dependencies can be compared to against each other. To choose the best option, the as-is and to-be state should exist. As first step of the *Construct* a model is created containing all components which should be analyzed. Afterwards the models are compared with a previous state or with another alternative. On this way all possible states can be observed and the best alternative to achieve the goals is identified. The technique **Views** is often only a part of another analysis. However we identified analyses having highly elaborated approaches with the main focus on views (e.g. (Sasa and Krisper, 2011)). For instance views analyze effects and requirements. Therefore criteria and their desirable perspective have to be specified. After this the views can be built with all required components. A definite *Measurement* is not contained in this category. However, views can be evaluated with criteria to observe whether the view can achieve its goals, for example the processing time. A less popular methodology is the observation of **Lifecycle** (e.g. (Saat, 2010; Aier et al., 2009)). These analyses ascertain requirements and dependencies through consideration of architecture phases. In this way changes are identified and it's possible to determine whether an artifact presents a specific state at an specific time. To conduct this, lifecycle phases, attributes, times and states are necessary as *Input*. The analysis *Construct* displays the cycle on analyzed areas. Afterwards for every product the state is checked at a special point. For *Measurement* probabilities are calculated. These results, with the changed cycle are displayed in the *Output*. **Ontology** is a typical technique of EAM, however uncommon in analysis (e.g. (Sunkle et al., 2013)). A special meta model created with ontologies and ontology rules is required for the *Input* and *Conditions*. The *Construct* analyzes entities, resources, persons and products, determines the ontology of entities and defines connection types. Afterwards dependencies, viewpoints and special factors can be evaluated for the outcome. **EID** (Extended Influence Diagrams) are the third most used technique to conduct analysis in EA (e.g. (Johnson et al., 2007)). Possible results can be statements about maintainability, security and availability. Therefore systems, attributes, quality, impacts and data are analyzed. As *Condition* it has to be ensured that all contained components are able to be built with EID. Afterwards all scenarios, goals and entities are represented as EID nodes and connections. For *Measurement* Bayesian Networks are used to calculate the probabilities of attributes. Thus it's possible to analyze dependencies by inferring changes and altered values. Another identified technique is the usage of a **Matrices** (e.g. (Szyszka, 2009)). Matrices can be used in various ways, mostly they are utilized to present results. Therefore dimensions and the kind of measurement are determined in order to built and evaluate the matrix. The results can vary from quantitative outcomes to weak points, redundant artifacts and functional dependencies. Analyses joint in the category **Design** are only able to observe architecture design in a specific way (e.g. (Aier et al., 2011)). As *Condition* factors and expert knowledge is needed. In the main *Construct* items and data are determined, factors are checked with questionnaires and a cluster analysis is conducted. As *Measurement* a matrix of items and factors is built and evaluated. The results of the evaluation represent the *Output*. For the identification of **Weak Points** (e.g. (Xie et al., 2008)) and their costs the following procedure can be used. Resources are needed to create a matrix of workflows, resources and their availabilities. This matrix is filled with connections and their values and is the basis for availability calculations. Whenever the availability is higher as the respective requirement, the condition is fulfilled. In addition it's possible to weight resources and receive alternatives with higher availability. The last technical category contains analyses with **Structural** procedure (e.g. (Buckl et al., 2009a)). This analysis tries to observe design through displaying obstacles of different architecture versions. Therefore a documentation is needed as *Condition* and the main part of analysis consist of an observation of changes. The *Output* type is unique and represents potential obstacles caused by different versions.

4 EVALUATION

The identified requirements for the 10 functional categories and the 17 technical categories are formalized using a domain specific language in order to validate the categorization. Therewith we can elaborate the integrity and correctness of the requirements, i.e. if they are sufficient to describe the analyses in an adequate way. For each category we chose an analysis approach from literature and formalized it using the developed language. In the following we present the DSL and illustrate its usage. Finally we provide a short discussion about our results.

4.1 DSL for Analysis Description

For the language development we used Xtext[1], a framework that comprises a powerful language for the description of textual languages. The model is generated by the framework as well as an parser, linker, type checker and compiler. The DSL was developed according to the meta model development process for abstract syntax development in (Brambilla et al., 2012). This incremental and iterative process consists of three phases: The 'Modeling Domain Analysis' phase, elaborating the purpose and content, the 'Modeling Language Design' phase, defining the meta model, and the 'Modeling Language Validation' phase, verifying the correctness and integrity. For the last step we select representative EA analyses for each category and formalize them using our modeling language. Difficulties and mistakes during modeling triggers a new iteration of the development process. The concrete syntax is developed simultaneously with the abstract syntax due to the nature of Xtext.

The developed DSL is structured in a general and in a categorization specific part. Figure 3 shows the main rule for the analysis language and the realization of the dimensions. General requirements that occur in all categories are summarized at beginning in the main rule. This is the name of the analysis, the required meta model and possible scenarios to evaluate. For description of the meta model and the scenarios we developed a language construct that allows to specify them similar to UML. The goal of an analysis can be modeled using a string and its type is

[1]Xtext https://eclipse.org/Xtext/index.html

```
MetaLanguage:
  'EAM Analysis Language' '{'
  //Domain Definition: General Requirements
    'Performing Analysis' analysis=STRING
    'Metamodel' model+=UMLModel ('{'
       'Scenarios:' scenarioName+=NameIdentifier (scenarioModel+=UMLModel)*
             (";" scenarioName+=NameIdentifier (scenarioModel+=UMLModel)*)*
    '}')?
    'Goal' goal=STRING
    'Goal Type'':' goalType+=GoalType ('&' goalType+=GoalType )*
  '}'
  //Choice of Dimensions
    ('Category functional Dimension' ':' functional+=Functional)?
    ('Category technical Dimension' ':' technical+=Technical)? ;

//----------Functional Categories--------------------------------------//
Functional:
  SystemAnalysis | AttributeAnalysis |    ...     | BusinessObjectAnalysis;

//----------Technical Categories--------------------------------------//
Technical:
  BayesianNetworks | BusinessEntities |  ...  | Structural ;

//Choice of a possible technique matching to the chosen functional Category
SystemAnalysis:
  'System Analysis' (':')?
  ('Technique' analysisTechnique+=SystemAnalysisTechnique)? ;

SystemAnalysisTechnique:
  EID | PRM | BayesianNetworks ;
```

Figure 3: DSL for EA analyses - main rule.

defined using an enumeration. Possible goal types are: percentage, matrix, probability, dependency, object, effect, scenario, number or boolean. The two-dimensional categorization is realized as followed: The user can either first choose the functional dimension and then the technical, but also the other way round. The rule system of the DSL restricts the second dimension to those that are available. For example the functional dimension **System Analysis** has realizations with the technical dimensions **EID**, **PRM** and **Bayesian Networks**. The rule *SystemAnalysis-Technique* ensures the integrity of the selection according to the matrix (figure 2).

```
EAM Analysis Language{
  Performing Analysis "Information Security Analysis"
  Metamodel Model"Architecture of Information Security"{
    Class "Application"{ ...  }
    ...
  }
  {Scenarios:
    "Scenario 1" Model"UML Model Scenario 1"{    ...     };
    "Scenario 2" Model"UML Model Scenario 2"{    ...     }
  }
  Goal"Probability of quality attributes for security"
  Goal Type :Percentage
  }
  Category functional Dimension:Attribute Analysis:
  Technique Extended Influence Diagram
    INPUT{
      Metamodel "Architecture of Information Security"{
        Scenario"Scenario 1",
        Scenario"Scenario 2"
      }
    }CONSTRUCT{
      EID MODEL ELEMENTS{
        Chosen Scenario "Scenario 1"
        //Value assumptions
        Scenario Node:
          type: Decision Node "Scenario Selection" Value: 0."90"
        Goal: type: Utility Node "Profit" Value: 0."0"
        Attributes:
          type: Chance Node"User Training Process" Value: 0."75"
          ...
         Relations:
          "Scenario Selection" as Causal Relation to "User Training Process"
          ...
        }}...
```

Figure 4: Excerpt of the description of EID analysis using the DSL.

For each technical category a rule is implemented that satisfies the requirements specified in chapter 3. The rules comprise statements for defining the input, the conditions and construct, the measurement and the output. To illustrate the structure of a category definition figure 4 shows an example description of the Information Security Analysis from (Johnson et al., 2007). This analysis evaluates the architecture by calculating the probability of quality attributes for security. Corresponding to the main rule the description starts with the analysis name followed by a specification of the meta model and two scenarios. The meta model describes the classes, relationships and attributes that are necessary for the analysis. The two scenarios represent different alternatives that should be evaluated. The scenario description is followed by the goal statement and the goal type, in this case *percentage*. Then the functional and technical dimension are defined. The functional dimension is **Attribute Analysis** and the technical one is **Extended Influence Diagram**. Following the remaining structure of the analysis specification is specific for analyses of the category **EID**. The input of the analysis is here straightforward the defined meta model and both scenarios. The construct part defines the requirements in order to create extended influence diagrams. First the chosen scenario is set, then the nodes, goals and attributes with their types and values are defined. Finally the EID specific relations are specified.

4.2 Discussion

We were able to define and apply a domain specific language for the description of EA analyses. The identified categories, functional as well as technical, are integrated. It was also possible to describe the analysis approaches from literature using the DSL. Additionally the language is easy extendable and without special knowledge understand- and usable. The language can be reused for the development of new analyses, since it provides a sound foundation of requirements that have to be extended. After further development the language can also be used as an entry point for the specification and execution of analysis. Most of the requirements for the technical categories are realized. A few requirements are determined as given and not further mentioned in the language, since these requirements are obviously. Examples are the possibility to raise data or whether data can be used and be accessible. In addition requirements which are not verifiable couldn't be included. For instance, it is not verifiable whether the meta model is usable to achieve goals, whether artifacts are able to image with EID components or used nodes are controllable. Additionally the acceptance of an used technique or the availability of experts knowledge is not verifiable and thus not integrated in the language. Requirements that are defined in a graphical way, for example matrices, are difficult to realize in a textual language. Also the definition of patterns is only specified with limitations in the language.

The lower number of functional categories in contrast to the technical one can be explained with the focus on one field of interest. Since we concentrate on pure EA analyses the analysis goals were repetitive. A problem during categorization was the issue that not all aspects from the analysis are described in detail in the available publications. At this point we were only able to identify limited requirements or we had to make assumptions to proceed. A interrelated problem is the fact of the low amount of available descriptions of conducted analyses to evaluate our language. Additional some analyses use very specific techniques or modeling approaches. Here, it was not possible to consider all details in order to create a sound categorization. We abstracted from some specifics in order to define the general requirements for a category. We received generally valid requirements by orienting on the approach which is most elaborated and create a higher level of abstraction. An example is the technical category KPI with a high abstraction level. The contained analyses differ deeply in measuring values with different formulas.

Encountered categorization approaches in related work tried to work on a meta level after studying the analyses. However, in contrast to our target they designed an analysis framework meta model independently (Langermeier et al., 2014), developed a category independent meta language (Buckl et al., 2011) and had the main focus on characteristics (Buckl et al., 2010). Additional EA analyses can be distinguished between the point of execution time. Therefore the analyses are sorted in ex-ante and ex-post to determine whether an analysis will be conducted before or after the adoption of an architecture. It is also possible to separate the analyses according to their execution technique: expert-based, rule-based or indicator-based (Buckl et al., 2009b). However, both classifications are not detailed enough to identify characteristics and most of analyses can't be strictly classified within these divisions. Lankhorst (2013) conducted an initial categorization with the already mentioned four dimensions. Quantitative and functional differ at the input and output data, whereas functional can be distinguished in static or dynamic. However this division is not detailed enough to identify the explicit requirements of classified analyses and four categories is a rough classification.

5 CONCLUSION

In this paper we presented a two-dimensional categorization of EA analyses, based on the characteristics of the approaches found in literature. The first dimension addresses the functional aspect, the second one the technical aspect. Altogether we identified 105 analyses, which are classified in 10 functional categories and 17 technical categories. Using this categorization we can identify 40 different analysis types used in EA. The dependencies between the approaches of the functional and technical dimensions are visualized in a matrix. The dependencies as well as the characteristics of the analysis categories are formalized with a domain specific language. The language provides a structural way to represent the preconditions of an analysis, the technical requirements for execution and also the outcome of it. Beside evaluation purposes the language can also be used by the enterprise architect to decide whether the outcome of an analysis is from interest for his question, if the analysis is applicable on his EA model and how great the effort of adaption are, in order to execute the analysis. The idea of such an EA analysis catalog is the support of reuse of existing work in the domain of EA. Therefore future work has to investigate techniques for context independent execution of those analysis. This could be the development of tool support for the usage of the categories and the DSL. Thus computations which need further tools can be included, new analyses could be created easily and requirements are checked automatically. Additionally a higher abstraction level of the category characteristics would be conceivable to make the requirements general valid.

REFERENCES

Aier, S., Buckl, S., Franke, U., Gleichauf, B., Johnson, P., Närman, P., Schweda, C. M., and Ullberg, J. (2009). A survival analysis of application life spans based on enterprise architecture models. In *3rd Workshop on EMISA*, pages 141–154.

Aier, S., Gleichauf, B., and Winter, R. (2011). Understanding enterprise architecture management design-an empirical analysis. In *Proceedings of 10th Conference on Wirtschaftsinformatik*.

Brambilla, M., Cabot, J., and Wimmer, M. (2012). *Model-driven software engineering in practice*.

Buckl, S., Buschle, M., Johnson, P., Matthes, F., and Schweda, C. M. (2011). A meta-language for enterprise architecture analysis. In *Enterprise, Business-Process and Information Systems Modeling*, pages 511–525. Springer.

Buckl, S., Matthes, F., Neubert, C., and Schweda, C. M. (2009a). A wiki-based approach to enterprise architecture documentation and analysis. In *ECIS 2009 Proceedings*, pages 1476–1487.

Buckl, S., Matthes, F., and Schweda, C. M. (2009b). Classifying enterprise architecture analysis approaches. In *Enterprise Interoperability*, pages 66–79. Springer.

Buckl, S., Matthes, F., and Schweda, C. M. (2010). A Metalanguage for EA Information Modeling State-of-the-Art and Requirements Elicitation. In *Enterprise, Business-Process and Information Systems Modeling*, pages 169–181. Springer.

Buschle, M., Ullberg, J., Franke, U., Lagerström, R., and Sommestad, T. (2011). A tool for enterprise architecture analysis using the PRM formalism. In *Information Systems Evolution*, pages 108–121. Springer.

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS quarterly*, 13(3):319–340.

de Boer, F. S., Bonsangue, M. M., Groenewegen, L., Stam, A., Stevens, S., and Van Der Torre, L. (2005). Change impact analysis of enterprise architectures. In *IEEE International Conf. on Information Reuse and Integration*, pages 177–181.

Della Bordella, M., Liu, R., Ravarini, A., Wu, F. Y., and Nigam, A. (2011). Towards a method for realizing sustained competitive advantage through business entity analysis. In *Enterprise, Business-Process and Information Systems Modeling*, pages 216–230. Springer.

Johnson, P., Lagerström, R., Närman, P., and Simonsson, M. (2007). Enterprise architecture analysis with extended influence diagrams. *Information Systems Frontiers*, 9(2):163–180.

Kazienko, P., Michalski, R., and Palus, S. (2011). Social network analysis as a tool for improving enterprise architecture. In *Agent and Multi-Agent Systems: Technologies and Applications*, pages 651–660. Springer.

Krcmar, H. (2015). *Informationsmanagement.* Gabler.

Langermeier, M., Saad, C., and Bauer, B. (2014). A unified framework for enterprise architecture analysis. In *18th IEEE International EDOC Conference Workshops*, pages 227–236.

Lankhorst, M. (2013). *Enterprise Architecture at Work: Modelling, Communication and Analysis*. Springer.

Närman, P., Franke, U., König, J., Buschle, M., and Ekstedt, M. (2011). Enterprise architecture availability analysis using fault trees and stakeholder interviews. *Enterprise Information Systems*, 8(1):1–25.

Närman, P., Johnson, P., Ekstedt, M., Chenine, M., and König, J. (2009). Enterprise architecture analysis for data accuracy assessments. In *13th IEEE International EDOC Conference*, pages 24–33.

Närman, P., Schonherr, M., Johnson, P., Ekstedt, M., and Chenine, M. (2008). Using enterprise architecture models for system quality analysis. In *12th IEEE International EDOC Conference*, pages 14–23.

Niemann, K. D. (2006). *From enterprise architecture to IT governance*. Springer.

Rauscher, J. (2013). Analysen in Unternehmensarchitekturen - Ziele, Techniken, Anwendungsbereiche. *Bachelor Thesis, University Augsburg*.

Rauscher, J. (2015). Anforderungen an und Definition von einer Analysesprache für das Enterprise Architecture Management. *Bachelor Thesis, University Augsburg*.

Razavi, M., Aliee, F. S., and Badie, K. (2011). An AHP-based approach toward enterprise architecture analysis based on enterprise architecture quality attributes. *Knowledge and information systems*, 28(2):449–472.

Saat, J. (2010). Zeitbezogene Abhängigkeitsanalysen der Unternehmensarchitektur. In *MKWI*, pages 119–130.

Sasa, A. and Krisper, M. (2011). Enterprise architecture patterns for business process support analysis. *Journal of Systems and Software*, 84(9):1480–1506.

Sunkle, S., Kulkarni, V., and Roychoudhury, S. (2013). Analyzable enterprise models using ontology. In *CAiSE Forum*, volume 998, pages 33–40.

Szyszka, B. (2009). Analysis and classification of maturity models in enterprise architecture management. *Bachelor Thesis, Technical University Munich*.

Van Lamsweerde, A. (2001). Goal-oriented requirements engineering: A guided tour. In *5th IEEE International Symp. on Requirements Engineering*, pages 249–262.

Wan, H. and Carlsson, S. (2012). Towards an understanding of enterprise architecture analysis activities. In *Proceedings of 6th ECIME*.

Winter, R. and Fischer, R. (2006). Essential layers, artifacts, and dependencies of enterprise architecture. In *10th IEEE International EDOC Conference Workshops*.

Xie, L., Luo, J., Qiu, J., Pershing, J., Li, Y., Chen, Y., et al. (2008). Availability “weak point” analysis over an SOA deployment framework. In *IEEE Network Operations and Management Symposium*, pages 473–480.

Zia, M. J., Azam, F., and Allauddin, M. (2011). A survey of enterprise architecture analysis using multi criteria decision making models. In *Intelligent Computing and Information Science*, pages 631–637. Springer.