A model-based method for the evaluation of project proposal compliance within EA planning

Melanie Langermeier
Software methodologies for distributed systems
University of Augsburg
Augsburg, Germany
langermeier@informatik.uni-augsburg.de

Bernhard Bauer
Software methodologies for distributed systems
University of Augsburg
Augsburg, Germany
bauer@informatik.uni-augsburg.de

Abstract—The business model and IT infrastructure of organizations are changing continually. Trends like microservices and digital transformation demand an adaption of the business models and IT infrastructure in order to stay competitive. It is important to ensure the compliance of these new projects with the current goals and principles. The discipline of Enterprise Architecture (EA) Planning provides methods for the structured development of the business and IT of an organization. In this paper we propose a tool-supported method for EA planning to evaluate to the project compliance based on established models. Different analyses are used to support the architect during project planning. Gap and impact analysis are used to ensure the change consistency. The compliance with the current strategy is finally evaluated with view generation and metric calculation. Foundation of the method is a generic analysis architecture execution environment (A2E), that provides us with the required flexibility to adapt to different needs and meta models. The method and the proposed analyses are evaluated within a case study from a medium-sized software product company.

Index Terms—Architecture Analysis, Architecture Evaluation, Enterprise Architecture, Enterprise Architecture Planning

I. INTRODUCTION

Microservices and digital transformation are innovations that force organizations to adapt their business models and restructure their IT infrastructure in order to stay competitive. Enterprise Architecture Management (EAM) has been proposed as a way to manage organizational changes and their inherent complexity. It provides a documentation of the relationships between the business and IT as well as structured procedures for transformation planning. Therewith, EAM addresses the challenges of business-IT alignment and the development of new businesses models. An optimal alignment of business and IT is crucial for the success of any organization. And while analyzing the dependencies from business down to the IT infrastructure, EAM reduces the risks during enterprise transformations [1]. The key concept of EAM, the Enterprise Architecture (EA), describes the elements and structures of an organization and their relationships. Typical layers of an EA are the business architecture describing the organization from a strategy viewpoint and the process architecture containing artifacts like business processes and responsibilities. The integration architecture and the software architecture deal with enterprise services, application clusters, data flows and

data structures and software services. Finally in the technology architecture the hardware and networks are described [2].

After documenting the current EA its future development is planned in terms of strategy and goal definitions as well as the construction of a target architecture. The discipline of EA Planning (EAP) deals in specific with the development and implementation of these future plans [3]. Transformation planning from the current to the target architecture has to be done in alignment with the strategy of the organization [1]. While implementing the desired target architecture, organizations are faced with new demands [4]. Demands can be driven by new technologies and trends, the need for cost reduction or the integration of standards. A current example are microservice architectures. Sources for business-driven demands are business developments, new innovations and strategy changes. Additionally, ensuring the compliance to legal regulations like Basel III for the financial sector or Merger & Acquisitions lead to changes in the architecture. Such changes are not necessarily conform to the architectural strategy, i.e. the specified principles and goals. The resulting phenomena of a moving target is known challenge within EA planning [3], [5]. It is important to integrate the EA strategy within the projects, i.e. plan and execute them in an EA compliant way [5]–[7]. Only if the projects follow the defined strategy, the benefits from the EA initiative and the desired target architecture will be reached.

Adequate evaluation possibilities are required to decide about the fitness and compliance of a new project proposal [8]. The respective managers require aggregated and integrated information [9], [10]. Additionally, they use different viewpoints, created upon partial EA models, for decision making [10]. In current practice the proposed methods for EA planning from literature are not widely adapted. Often because of insufficient data quality but also because practical approaches, especially for comparisons, are missing [1].

In the following we propose a practicable method to quantify and evaluate project proposals to support decision making about their initiation. The method and tool support should be defined in a generic way, that enables an adaption to different EA frameworks as well as planning processes. We employ tool support from previous work to highlight the changes to architectural elements, ensure their consistency using an im-

Spewak et al. [17,18]	Niemann [6]	Pulkkinen et al. [10,19]	Aier et al. [4]	Aier, Saat [20]
A1. Planning initiation	B1. Define goals	C1. Initiation	E1. Define Vision	D1. Strategic EA planning
A2. Define values and principles	B2. Documentation	a. Define goals	E2. Model As-is Architecture	a. Derive goals
A3. Identify business knowledge and current systems and technology	B3. Analysis	b. Resource and constraints	E3. Model (multi-step) alternative to be architectures	b. Define long term vision
A4. Blueprint data, applications and technology architecture	B4. Planning alternative scenarios	C2. Planning and development: define needed changes in architectural dimensions	E4. Analyze and Evaluate Alternative (subsequent) to-be architecture	c. Prioritize courses of action
A5. Develop implementation and migration plan	B5. Evaluation of alternative scenarios	C3. Ending phase	E5. Plan Transformation from As-is to to-be	D2. Operational EA planning
A6. Define programmatic transition	B6. Implementation	a. Plan, design and evaluate alternative architectures and solutions	E6. Implement transformation	a. Define requirements
		b. Define long term and short term targets		b. Model variants
				c. Evaluate and select domain variants
				d. Consolidate to be models
				D3. Implementation
				a. Define change projects
				b. Launch projects
				c. Monitor projects

Fig. 1. Planning processes in literature (extended table from [3])

pact analysis, generate viewpoints for an architectural review and the calculation of metrics for evaluation purposes. The concepts of the Architecture Analysis Execution Environment (A2E) are presented in [11], [12], foundations for the single analyses in [13]–[15].

For the development of the method we first analyze EAM methods for planning support and EA planning processes in current literature and extract the common steps and method blocks (section II). Based on these results we identify automation potential within the method blocks and develop a semi-automated method for project proposal evaluation with respect to EA compliance (section III). The proposed analysis support is provided with the A2E presented in section IV. It provides the possibilities to perform different kinds of adaptable analyses as well as is independent from the EA meta model used in an organization.

II. METHOD BLOCKS FOR EA PLANNING

In this section current EAM methods proposing steps for implementing an target architecture as well as EA Planning processes are presented. From these methods and processes method blocks that are utilized are identified and summarized in table I.

According to [7] the EAM process integrates three different but interrelated cycles. The *Strategic planning cycle* starts with an analysis of the current situation and its documentation and concludes with a roadmap and project portfolio to develop the target architecture. Initiating a project triggers the *Project life cycle*, that deals with the set-up, design, implementation and roll-out of a concrete project. Finally in the *Operation and monitoring cycle* the EA is monitored and changes are collected, assessed and implemented. During monitoring, essential information for the architecture evaluation is collected and provides feedback for the strategic planning. Following [7] a project can be initiated during strategic planning in order to realize the desired target architecture or be a demand driven project. It is important that demand-driven projects are realized with respect to the EA strategy. To ensure the

EA compliance the author proposes reviews of the created specifications, design documents or prototypes.

Another common EAM method is the TOGAF Architecture Development Method (ADM). Based on the architecture vision (phase A) the business, information systems and technology architectures are defined (phases B, C, D). Each of the phases comprises the definition of the current state, a target architecture as well as a respective roadmap. The subsequent phases deal with the realization of the specified target architecture (phases E, F, G) and the architecture change management (phase H). They include performance and gap analysis to ensure the conformity of change requests with the EA governance and framework.

According to Niemann [5] the EAM cycle contains the following 5 steps: *Document*, *Analyze*, *Plan*, *Act* and *Check*. Within the planning step, the roadmap from the current architecture to the desired target is defined. Development planning is required in order to integrate new projects with the optimization of the existing application landscape. The proposed process for this step according to [5] is shown in figure 1. Within this process the author proposes "what-if" analyses, gap analyses and well as an architecture evaluation according to the goals, costs and risks.

Despite method blocks within EAM methods, there exist also specific approaches for EAP. One of the first approaches for EAP is the 'Wedding Cake Model' from Spewak and Hill [16], [17]. Based on an analysis of the current application landscape, e.g. through the use of metrics, the data, application and technology architecture are developed. The results of a gap analysis between the as-is and the to-be architecture are used to develop an implementation and migration plan. The EAP process from Aier et al. [3] starts with definition of the vision and the creation of the as is-architectural model. In the following, (multi-step) to-be alternatives are modeled, analyzed and evaluated. Finally, the transformation from the as-is to the to-be is planned and implemented.

In [9], [18] the authors propose a process model for the management of architectural decisions in EA planning. Thereby they differentiate between architectural layers (i.e.

	Method block	Authors
M1	Development of scenarios or alternatives	[1], [3], [5], [9], [18], [19]
M2	Use of domain architectures	[1], [6], [9], [18], [19]
M3	Evaluation of cost, risk, metrics, performance	[1], [3]–[5], [9], [16], [18], [19]
M4	Evaluation of impact	[4]
M5	Evaluation of compliance to principles and goals	[5], [7], [16], [20]
M6	Evaluation of gaps	[1], [4], [5], [16]
M7	Evaluation through different views (visualizations or abstractions)	[1], [4], [5]

business, information, application and infrastructure) and three decision making levels (enterprise, domain and system level). The authors propose a spiral model for decision making, where decisions are refined top-down, considering the EA layers and the decision-making levels. A concrete EA project is performed in three steps: Initiation, working phase and ending phase. The phases encompass goal and architectural change definitions as well as the design and evaluation of alternative architectures and the creation of a development road map.

In [19] the authors performed a comparison of EA planning approaches in practice and literature and concluded with a set of common activities during EA planning. The proposed process differentiates the three main steps *Strategic EA Planning*, *Operational EA Planning* and *Implementation*. Similar the the results of [9], [18] the authors identified three different levels for the activities: The Enterprise Level (mostly strategic planning), the Domain Level (mostly operational planning) and the Project Level (mostly implementation). The definition of requirements for the identified actions as well as the modeling of different to-be variants in the operational step is performed on domain level. Since the changes do not affect the entire EA the concrete planning can be performed using domain architectures. The established to-be models are finally integrated in the enterprise model to get a consistent blueprint [19]. Evaluation takes place on domain level as well as on the consolidated enterprise level.

Additionally to EAM methods and EAP processes [1] and [21] identified several requirements for EAP processes, that are also considered while identifying the method blocks. Nowakowski et al. [1] provide a review of current literature about EAP as well as results of practitioner interviews [1]. Based on them, they extract several requirements for EA planning. They include the ability to analyze and compare the current as well as the to-be architecture (including qualitative and quantitative metrics), the support for different planning scenarios and transformation paths as well as the availability of up-to-date data. In order to set-up a transformation path it is necessary to create a transformation model and to derive transformation projects from the to-be architecture. The authors propose also a gap analysis. They also point out the relevance of individual life-cycles, the ability to react to unplanned changes as well as the importance of tool support.

In comparison with the practitioner interviews [1] found out that EA planning is primarily carried out by visual comparison of specific models, while the methods proposed by literature are not considered in practice. They propose the development of simpler and more practical approaches especially for the comparison of scenarios.

Aier et al. [21] also work out several requirements for EAP. Among those are the development and evaluation alternatives and the consideration of successor relationships and life cycles in the to-be models. It must be possible to derive project activities from the developed to-be model. Overall mean of EAP is the provisioning of information to support change projects. Thereby the different requirements from the stakeholders have to be considered.

Finally Foorthuis et al. [6] differentiate in their work about EA project compliance between three different layers: Enterprise Architecture (EA), Domain Architecture (DA) and Project Architecture (PA). EAs describe the as-is and to-be at the highest level, whereas DAs focus on one specific group of products, services or functions. The focus of PA are the relevant artifacts for one project and describe specific solutions. Additionally, the authors propose an EA compliance model including a compliance testing process in [20]. The results of this work are also considered in table I.

III. SEMI-AUTOMATED PROJECT PROPOSAL EVALUATION METHOD

In the following we propose a method for the evaluation of project compliance during EA planning based on the identified common method blocks. The method was created within an iterative process compromising the steps (1) development of process activities, (2) enhance tool support and (3) evaluation of its applicability. Tool-supported analyses are employed for the single method steps to manage the complexity and extend of current EA models. The data foundation for the analyses is provided through utilizing established architecture models. To ensure the applicability, the method and the analyses should be adaptable to individual needs as well as combinable with existing EAP processes and EA models. The utilized Architecture Analysis Execution Environment (A2E) provides support for generic analysis definitions as well as customization of the analyses to the specific needs of the stakeholders. The A2E as well as details about the utilized analyses are results of previous work [11]–[15].

Figure 2 gives an overview of our proposed method and the necessary upstream and downstream activities. Preliminary the goals and principles for the enterprise architecture have to be defined as well as metrics in order to quantify them. Additionally, the current architecture is documented and up to-date and partial model(s) representing the proposal(s) are available. Project proposals may introduce changes to the business model, like new provisioning model for products, but also changes within the application and infrastructure architecture. [1], [5], [9], [19] recommend also the development of alternatives in order to find the best solution. For each proposal the evaluation method will be executed and finally the
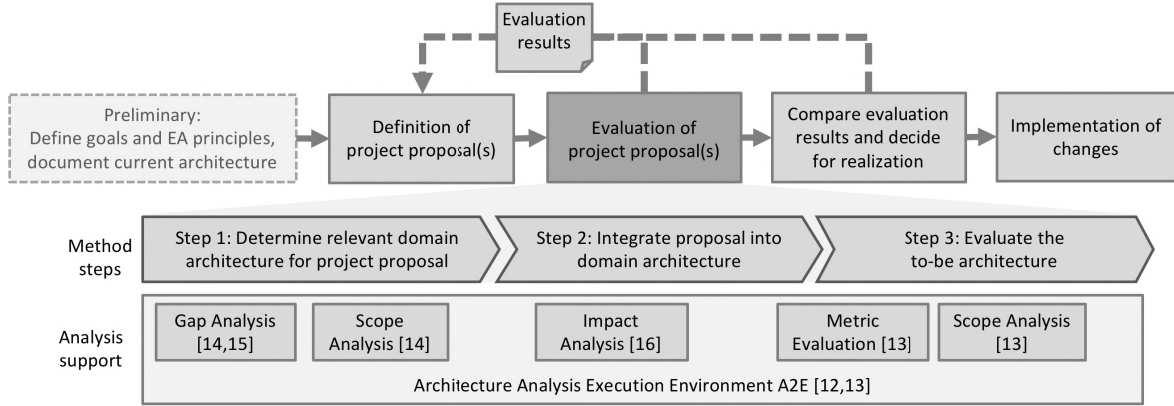
Fig. 2. Overview of project proposal evaluation method

results can be compared to each other. As a result, the architect decides for one alternative and starts the development of an implementation and migration plan. If the evaluation results are not promising, the proposal(s) can be rejected or improved based on the evaluation results. This triggers a new iteration of the method.

Our proposed method focuses on the activity *Evaluation of project proposals*. This method consists of three steps: The determination of the relevant domain architecture in order to narrow the scope and support user based verification. The integration of the proposal into the domain architecture and generation of the to-be architecture. And finally, the evaluation of the to-be architecture. The key analysis concepts utilized in the single steps are gap analysis, generation of domain architectures, impact analysis as well as metrics calculation and view generation. Impact analysis is used to ensure the change consistency of the project proposal. Metric calculation is used to quantify the fitness of the proposed to-be architecture and views are used to enable a manual verification by a specialist. Domain architectures are an often proposed concept to provide the architect manageable parts of the architecture, where dependencies can be obtained visually [6], [9], [19]. Due to the large nature of EA models and their high complexity, domain architectures and views are required to enable user feedback throughout the planning process. Full automation without user feedback is not applicable, since the architect has relevant but implicit knowledge about the architecture and the strategy. [7]. Thus, user verification of created artifacts and feedback loops are another main concept of our method.

The analysis support for these key concepts is described in detail in the following, utilizing the fictitious architecture of a car rental company (CarRental) as running example. The project proposal describes the introduction a 24h car return possibility. The execution details for the automated parts are described in section IV.

A. Determine relevant domain architecture

The first step identifies the relevant domain architecture for the proposal. Therefore, the dependencies between the current

EA and the project proposal are defined through a comparison of them. This task is supported by the *Gap Analysis* described in section IV-B. Finally, an dependency attribute is added to each element:

Unaffected	Element is only available in the current architecture or element is available in the current architecture and the proposal and the properties and relations keep unchanged
Affected	Element is available in the current architecture and the proposal and at least one property or relation is changed.
New	Element is only available the proposal.
Deleted	Element only available in the current architecture and removed in the proposal.

Affected elements can also point to a predecessor element in the current architecture, which they will replace. This successor information must be added by the user. Since the proposal is an uncompleted part of the future architecture, deleted elements cannot be specified automatically within the analysis. Also the user has to specify them explicitly. To support this task, the gap analysis calculates a set of deletion candidates. These elements are part of the current architecture and not in the proposal but have an relationship to an element of the proposal. Figure 3 shows the current architecture of the car rental company on the left and the project proposal for the 24h car return on the right. The results of the gap analysis are indicated through the different colors of the elements.

Once the dependencies between the current architecture and the proposal are approved by the user, it is possible to generate the relevant domain architecture. It is defined with respect to the content of the project proposal and utilizes the same abstraction layer as the enterprise architecture. The domain architecture encompasses business as well as IT aspects. It is generated automatically using the Scope Analysis described in IV-D. We identified the following requirements, that have to be met by the domain architecture in order to provide a solid foundation for the subsequent evaluation:

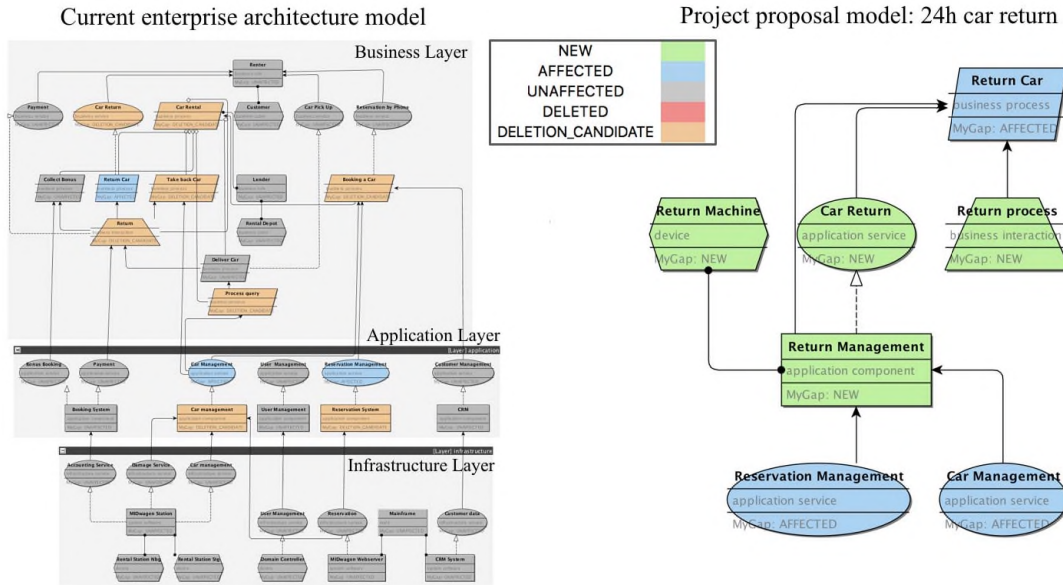- All affected and deleted elements must be included.

Fig. 3. Gap analysis result between the proposal and the current architecture

- Elements that are required to provide to the affected elements.

The last point ensures that potential dependencies or changes, that are not considered in the initial proposal, may be detected during the following evaluation. Thus, inconsistencies in the planning proposal can be identified. Result of this step is the relevant part of the current enterprise architecture for the project proposal evaluation. The scope of the domain architecture has to be big enough to cover all effects of the project proposal but also small enough to keep manageable by the architect. The architect has to verify the generated as-is domain architecture and if necessary make adaptions, like including or excluding further elements. This is important, since the resulting domain architecture is used for all subsequent analysis and evaluation steps.

B. Integrate proposal into domain architectures

After the definition of the relevant domain architecture, the architectural changes made by the project are integrated and validated. Therefore, the to-be domain architecture is established and an approximation of the change impact is used for validation. The to-be domain architecture contains all affected elements and unaffected element from the calculated as-is domain architecture. Also newly introduced elements from the proposal as well as deleted elements are added. Thereby an attribute indicates the planning status for this element, i.e. if it is new, deleted or affected. Additionally, new elements can have further information about predecessor elements in the current architecture. Figure 4 shows the to-be domain architecture for the 24h car return proposal. The color of the elements indicate their planning status.

The automatically defined domain architecture has to be verified by the user and if necessary further adaptions have to be made. In order to validate its scope and quality the change consistency, i.e. if all direct and indirect effects of the proposed changes are considered, can be verified. For this task, each element has to be assigned one of the following change statuses:

no change	no changes are actively made to the respective element
extension	the element's functionality keeps remained, but new one is added
modified	the element's functionality is changed
deleted	the element is no longer available

The information is utilized to approximate the direct and indirect effects of the changes using the *Change Impact Analysis* described in section IV-C. The result is visualized, for example using different colors, within the to-be domain architecture and used by the architecture for a verification. Indicators for inconsistencies are unchanged elements, for whom a change is calculated or change effects outside the domain architecture. In the last case, potential changes may be unconsidered. The architect should evaluate the respective elements and optionally extend the domain architecture. Also, a large amount of unchanged elements is an indicator for a insufficient scope of the domain architecture. In order to ease the ongoing evaluation, the scope of the domain architecture may be decreased. Result of this step is a verified to-be domain architecture regarding its scope and consistency as well as the specification of a more detailed change status.
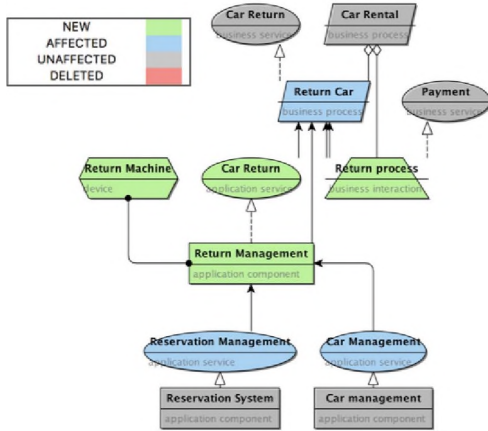
Fig. 4. To-be domain architecture for project proposal

C. Evaluate the to be architecture

At least, the to-be architecture is evaluated to ensure the EA compliance of the project and to evaluate its fitness with the architecture strategy. We propose view generation and metrics calculation to support the architect in this task. Different views on the to-be architecture support the architect in assessing the quality of the designed solution in detail. Each view serves a specific need and has a different focus where irrelevant elements are hidden. For example, the business perspective hides elements from the application and infrastructure layer and therewith enables a detailed review of the dependencies between the business elements. To ease this task and ensure the consistency of the views the view creation process can be automated with the *Scope Analysis* (section IV-D).

Metrics can be used to quantify the to-be architecture and compare it with the current architecture, other alternatives or desired target values. The metrics can be used as a quantification for goals or to express architecture principles and validate their degree of compliance. The details for metric execution are provided in section IV-E. In the following we illustrate their application within the running example. The CarRental company has the goal of 'no manual data transfer' between the applications and that a process is supported by only one IT component. To measure the goal fulfillment three metrics are defined in the example: (1) The IT Coverage indicates the amount of processes with IT support; (2) IT Usage provides the average number of applications used by a process; and (3) Connectivity as the average number of services that an application uses. Table II shows the result for these metrics for the as-is and to-be architecture.

Based on the results of the evaluation the decision about project implementation can be made. In the example, the to-be architecture with the proposed project increases all metric values. Specifically, all measures of the to-be domain architecture fulfill the target value, i.e. this project is conform to the architecture strategy respectively the goal, that is measured

Measure	As-is EA	To-be EA	As-is domain	To-be domain	Target value
IT Coverage	71%	75%	66%	100%	>80%
IT Usage	1,6	1,75	0,6	1,3	>1,2
Connectivity	0,58	0,87	0,62	1,12	>1

with these metrics. If the evaluation results are not satisfying, they can be used as input for the definition of a new project proposal that fits better into the architectural strategy.

IV. ANALYSES AND THEIR EXECUTION

In the following the technical details about the analyses mentioned in section III are provided. Thereby the adaptability of the analysis execution environment, which realizes all required analyses, is ensured with two different mechanisms: The foundation on a generic meta model that represents the enterprise architecture as a stereotyped graph. This issue deals with the great variety of EA meta models in use. The second point is the customization of the analyses according to specific needs. In the following, we first present the environment itself and subsequently present the details about the single analyses.

A. Architecture Analysis Execution Environment A2E

The architecture analysis execution environment (A2E) (presented in [11], [12]) is comprised of three main components: The *Analysis Definition*, the *Analysis Execution* and the *Model Storage*. An overview is given in figure 5. For *Analysis Definition* a domain specific language (DSL) was developed that enables the specification of executable analysis constructs. Supported analyses within the language are: Metric calculation, architectural scope definition, dependency analysis, change impact analysis as well as the composition of those analyses. An advantage of A2E regarding existing work is the integration of different analysis types within one execution environment. Current EA analyses are typically single approaches that require a specific meta model [22]. A2E provides the foundation for integrating different analysis types and utilize them for different EA models. Despite specific analyses for an EA model it is also possible to define templates that can be adapted to a specific EA model. This functionality is used to provide the majority of the analyses for our method and enables the customization by the architect.
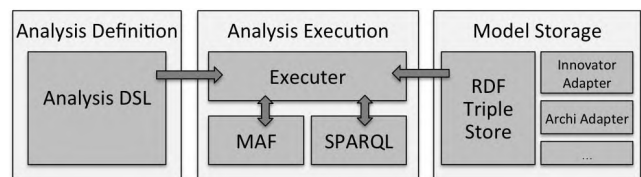

Fig. 5. A2E Architecture [12]

The *Model Storage* is built using an RDF Triple Store and adapters to several EA modeling tools like Innovator or Archi

but also for CSV files. Internally, the model is represented using the Generic Meta Model (GMM) [11], [12]. This meta model represents an EA model as a stereotyped graph with nodes, edges and properties for the nodes and edges.

Finally the *Analysis Execution* gets an analysis definition as input and converts it into executable constructs. Structural requests are converted into SPARQL queries, a graph-based query language for RDF data. Behavioral requests or recursive definitions are evaluated using Data-flow Analysis (DFA). DFA is a method to compute context-sensitive information based on declarative specification [23]. Through the combination of those techniques, we are able to provide an analysis execution environment that is able to deal with different meta models, incomplete EA models as well as covers most of the analysis types applied in the EA context [12]. In order to support our proposed method adequately, the A2E presented in [12] was extended with the Gap Analysis and the possibility to store different versions of one architecture in the model storage. Therefore, we utilized the concept of named graphs, a set of triples that is identified by a given URI. A RDF dataset can have one default graph, which represents the current architecture in our model storage. Further alternatives or target states are stored as named graph. This enables a differentiation between the same elements in different architecture versions. Combining different models, for example the current domain architecture with the project proposal model, can be done using the union-operator in the dataset. In the following, the supported analyses on these models are described in detail.

B. Gap Analysis

The purpose of the gap analysis is the determination of the differences, respectively gaps, between two different architectural models [4], [14]. According to [14] the gap analysis calculates three sets (targetArchitecture is equivalent to the proposalArchitecture in our case):

- onlyCurrentArchitecture $:= \{x \mid x \in currentArchitecture \land x \notin targetArchitecture\}$
- onlyTargetProposal $:= \{x \mid x \notin currentArchitecture \land x \in targetArchitecture\}$
- currentAndTarget $:= \{x \mid x \in currentArchitecture \land x \in targetArchitecture\}$

In the A2E these sets are calculated using the set operations *difference* and *intersection* on the RDF triple sets. Depending on the set an element belongs to, the respective planning status is defined according to the following rules:

1) $x \in onlyCurrentArchitecture \rightarrow$ unaffected
2) $x \in onlyTargetArchitecture \rightarrow$ new
3) $x \in currentAndTarget \rightarrow$ affected

Deleted elements and successor relationship between elements of the current and the proposal are unconsidered at this point. Heuristics can be used to provide further suggestions for those deleted elements or successor dependencies between elements from current and proposal. The heuristics utilize the context information of an element, i.e. the relationships, or a fuzzy name analysis. Elements with the assigned attribute

unaffected are potential deletion candidates, since it cannot be concluded automatically whether the absence in the proposal was with purpose or not. We further refine the set of deletion candidates to those, that have at least one relationship to an element affected by the project proposal. The strength of the dependency to affected elements can be expressed as metric and provides further information to the architect. But finally, it is the task of the architect to decide about the planning status of the element. In [13] similarity measures are proposed to identify successor dependencies between elements of the current and target architecture. Preliminary for this measurement is that there are already dependencies between current and proposal given, in terms of unaffected elements or already identified successors. The similarity measure is then calculated as

$$Sim(c)_p = \#sharedRelationships(c, p)/\#Relationships(c).$$

A shared relationship between c and p exists, if there is a relationship(c, r, c') in the current architecture and a relationship (p, r, p') and p' is the changed or unchanged successor of c'. C and c' are elements of the current, p and p' are elements of the proposal and r is an arbitrary relationship type. This measure is 0, if there are no shared relationships and 1 if all relationships are shared. The higher the value the higher the probability of a successor dependency. But here also, the architect has to decide about an successor relationship, an automatic conclusion cannot be made. The information about the dependencies between the current and the proposal is stored in a so-called transformation model [24]. This model contains the successor relationships between two architecture states. In our method the information is expressed using the planning status attributes in the to-be architecture as well as successor dependencies in the case of a replacement.

C. Change Impact Analysis

In the change impact analysis the effects of the changes, made by the project, are simulated through propagating change values along the edges of the EA model. Therewith potential effects of the proposed project, that are currently not considered, can be detected. Based on the ideas of [25] we developed respective rules for change propagation using Data-flow Analysis [15]. Considered change types are extension, modification, deletion and no change. Extension means that the existing functionality is kept and only new one is added, whereas the existing functionality of a modified element will be changed. For example, a propagation rule defines that a service has the change status deleted if the hosting application has the status deleted, i.e. the change status of an element is calculated with respect to the change status of the related elements. In [12] we integrated the change impact analysis in the analysis execution environment. The analysis therein supports two different modes: A static mode calculates the effects using predefined propagation rules in a worst-case and a best-case scenario. In the dynamic mode, the propagation semantics can be defined by the user. Therefore, the model

edges are mapped to the effect types *weak*, *medium*, *strong* and *no effect*. In order to execute the impact analysis, the user has to assign change types to the elements that are actively changed. Then the direct and indirect changes are calculated and visualized as colored attributes in the EA model. This information can be used by the architect to verify the change consistency within the proposal, i.e. if all affected elements are included. Additionally, it can be used to determine the quality of the domain architecture. A domain coverage metric indicates, whether all approximated effects are within the domain architecture. An additional architecture coverage metric indicates the effects of the change type in proportion to the while architecture. Together with an additional impact degree, that quantifies the approximated changes, the severity of the project can be measured. The metrics are provided according to section IV-E as adaptable template.

D. Scope analysis

The scope analysis is utilized for the generation of the domain architecture and for generating views. Both application scenarios rely on the concept of viewpoints. The A2E enables the definition of conditions, that are used to calculate a specific viewpoint. Employing the scope analysis definition on the architecture generates the respective partial architecture diagram.

A viewpoint addresses a specific concern of stakeholder which determines the scope of the viewpoint. For example, viewpoints can be used to either concentrate on the details of an aspect or the coherences between elements. Referring to the viewpoint definition in ArchiMate, we define a viewpoint as a selection of relevant element types and their relationships and the representation of that part of the architecture [26].

Example for a viewpoint is the *Business Process Viewpoint* from ArchiMate in order focus on the structure and composition of one or more business processes and also the associated services, roles and information objects. The viewpoint is used to focus on the details of the business processes and typically addresses the stakeholders process and domain architects and operational managers. The *Application Structure Viewpoint* provides a similar perspective but with focus on the applications and not the business processes. A typical stakeholder for this detailed viewpoint is the application architect. [26]

Within A2E we provide predefined templates for viewpoints and also enable the definition of individual ones, based on conditions about node types and edge types as well as specific property values of those elements. For example the *Application Structure Viewpoint* from ArchiMate can be defined as a set containing all elements that fulfill the following condition:

```
viewpontDefinition :=
    nodeType: "Application component"      OR
    nodeType: "Application interface"      OR
    nodeType: "Data object"                OR
    nodeType: "Application collaboration"
```

A domain architecture is a specific viewpoint related to scope of the project proposal. Characteristics of domain architectures are the reduced scope and the increased level of detail with respect to the EA [27]. Thus, the corresponding viewpoint includes the business and IT constructs related to the project. The domain architecture is defined in two steps. First all elements that are affected or deleted by the proposal are determined. Additionally, to consider also the realization infrastructure of those elements, for each element the realization scope is determined using DFA and the elements are included in the domain architecture.

E. Metric Calculation

Despite the generation of different views on the architecture, the calculation of metrics is another important part for the evaluation. Metrics are used to quantify goals and determine their achievement as well as to quantify benefits, risks and costs. They are calculated for the as-is architecture as well as the to-be (domain) architecture. The resulting values can be compared to each other in order to conclude the fitness of the proposed project. Examples for EA metrics can be found in the KPI catalogue [28]. Nevertheless, the metrics should be defined with respect to the goals and strategies of the enterprise architecture. Therefore, the architect should be able to define his own metrics. Within the A2E we provide the possibility to define individual ones as well as propose predefined templates. For metric definition, the common mathematical arithmetic operations are provided, e.g.:

`SUM:` to summarize all values defined by a calculation rule

`MULT:` to multiply all values defined by a calculation rule

`COUNT:` the number of elements or edges defined by several conditions

Conditions for an element set or an edge set within the `COUNT` statement are for example nodes having a specific property, element type or connected edge type or a combination of those using the set operations `AND` and `OR`. Atomic values in a calculation rule are a property value, another numeric analysis result or a constant. A metric can be calculated for each element or for the whole architecture. An element metric has an additional clause specifying the relevant element types. [12]

For example the IT coverage used for the evaluation of the running example in section III-C can be defined as followed:

```
itCoverage :=
  (COUNT (nodeType: "Business Process"   AND
          having relation to (nodeType:
              "Application Service")))
/   (COUNT (nodeType: "Business Process"));
```

This rule defines an aggregated metric as ratio of business processes having a relation to an application service (line 2 and 3) to all business processes (line 5).

V. EVALUATION

For evaluation purposes we compares our method with requirements from EAP literature and discuss its adaptability and integration within existing methods. Additionally, we evaluated our approach within a case study in a medium-sized software product company. The A2E and the single

analyses are evaluated itself in previous work. [13], [14] for the gap analysis, [15] for the impact analysis and [12] for the execution environment, its analysis type coverage and adaptability. Especially in [12] we showed the meta model independence of the analysis execution platform, as well as the possibility to define generic analysis templates and execute them on a concrete model. With this work, we show the benefit of such an integrated environment, since it enables the development of an applicable and customizable method for EAP. Since current EAP methods lack the acceptance in practice [1], we focused on the applicability of our method.

Utilizing EA models as foundation for the analyses provides two advantages: First, there is no need for specific data collection in order to apply the method and second, we can show the usefulness of these models during transformation planning. Second, establishing EA models is an expensive task and gaining benefits from these models increases the acceptance of an EA initiative. The utilized A2E provides us thereby with the required functionality to keep independent from a specific meta model for enterprise architecture. All required analyses for our method are defined as generic templates within the A2E. When applying the method these templates can be re-used as well as it is possible to further refine or extend them according to specific needs.

The steps within our proposed method are developed with respect to the EAP processes in current literature. We integrated the method blocks identified in literature (see table I) in our approach as well as propose tool-supported analyses for their execution. Comparing out methods with the requirements for EAP [1], [21] we cover most of them including an analysis and comparison of the current and target architecture, support for different scenarios, considering successor relationships between current and target as well as considering specific requirements from stakeholders. Weaknesses of our approach are a missing support for life-cycles and the derivation of project activities and support for transformation paths. An additional weakness of our approach is the dependency from the data quality and the completeness of the EA model. We consider this issue by providing support for csv-files as alternative input for an EA model, since Excel is a common tool in practice to capture such an architecture.

In figure 6 we mapped our EAP method steps onto the existing ones identified in literature. Process steps from literature are represented with the abbreviations introduced in figure 1. A step is mapped to a step from an EAP process, if they have similar main activities and the goals correspond to each other. Our main contribution is comprised within the process step *Evaluation of project proposals*. For the previous and subsequent steps, we do not provide further details. Therewith, it is possible to follow other approaches for these tasks and use our proposed method for executing the respective evaluation step (steps A4, B5, C3A, D2c and D2d as well as E4). This flexibility enables the execution of our method also in different organizations, where already a planning process is implemented.

We applied our approach subsequently within a case study

Steps in our method	Equivalent steps from other work
Preliminaries (principles and to-be)	A1-A3, B1, B2, B3, C1, D1, D2a, E1, E2
Definition of project proposals	A4, B4, C2, C3(a), D2b, E3
Evaluation of project proposal	B5, C3A, (A4), D2c, D2d, E4
Compare ,decide, implement	A5, A6, B6, D3, E5, E6

Fig. 6. Comparison with related work (abbreviations according to 1)

in a medium-sized software product company. The company wanted to shift the product provisioning to a Software-as-a-Service (SaaS) model. This transformation is divided into three main activities, which result in three different project proposal: Cloud Business Management, SaaS Operational Support and SaaS Security and Policy. The proposals have a size between only 12 architectural elements and over 80 architectural elements. During the SaaS project a model of the current architecture as well as to-be models for each proposal were created manually. For the evaluation we compared the generated models with the manually created ones.

The defined analysis templates could be applied to all three proposal without adaption. We observed that the rules for the generation of the domain architecture have to be adapted due to the different project sizes. For the large project, SaaS Operational Support, we restricted the amount of context element in order to ensure the manageability of the resulting domain architecture. Although, the rule for the generation have to be adapted in order to get the optimal output, the effort to create them manually was greater. Especially in the large projects the subsequent view generation was essential, to assess the quality and compliance of the project. We were able to generate the process specific views, that were created manually beforehand. The final view generation and metric calculation provide valuable input to decide about the quality of the proposals.

Especially the flexible creation of metrics to compare different scenarios is not sufficiently supported in current EA tools [29]. Additionally we experienced in our case study that the functionality for specific view creation is not sufficient for the planning task, which lead to manual adaptions. The amount of effort for the manual creation is significantly higher than with our method, additionally the consistency of the created models is not ensured.

VI. CONCLUSION

In this paper, we presented a tool-supported method for the evaluation of project proposals. Our focus was the development of a practicable method for EA planning. Therefore, we identified common method blocks from existing EAP processes and requirements in literature and developed tool support for them. Preliminary for the method is a documentation of the current enterprise architecture, defined goals and principles as well as project proposal model. Based on these artifacts in a first step the relevant domain architecture for the planning scenario is defined. This is supported with a gap analysis between current and target architecture and a

subsequent definition of the relevant architecture part using a scope analysis. In the second step the proposed changes are integrated in the domain architecture and a change impact analysis is performed to ensure the consistency and validate the domain architecture. Finally, views can be determined onto the to-be domain architecture for quality reviews as well as metrics can be calculated to quantify the target model. Again the scope analysis is utilized for view generation.

The analysis execution environment A2E provides a single point of access for all the different analysis types. The definition of generic analysis templates eases the execution of the proposed method. The applicability of our approach is also supported by utilizing the information within an EA model, thus additional data collection is not necessary. The universality of A2E enables an easy adaption to different EA models as well as provides the functionality to extend the analyses according to specific needs from stakeholders. For example, individual metrics or view definitions can be defined and the rules for the generation of the domain architecture can be adapted.

Currently we are applying the proposed method within two other use cases, a medium-sized production company and a hospital. In future work, we want to address the current weaknesses and include support for life cycles. We also take into account the integration of subsequent steps like deriving project activities or the definition of transformation paths.

REFERENCES

[1] E. Nowakowski, M. Farwick, T. Trojer, M. Husler, J. Kessler, and R. Breu, "Enterprise Architecture Planning: Analyses of Requirements from Practice and Research," in *Proceedings of the 50th Hawaii International Conference on System Sciences*, Jan. 2017.

[2] R. Winter and R. Fischer, "Essential layers, artifacts, and dependencies of enterprise architecture," in *10th IEEE International EDOC Conference Workshops*, 2006.

[3] S. Aier, B. Gleichauf, J. Saat, and R. Winter, "Complexity Levels of Representing Dynamics in EA Planning," in *Advances in Enterprise Engineering III*. Springer, Berlin, Heidelberg, 2009, pp. 55–69.

[4] D. Hornford, *TOGAF Version 9.1*. Zaltbommel: Van Haren Publishing, 2011. [Online]. Available: http://pubs.opengroup.org/architecture/togaf9-doc/arch/

[5] K. D. Niemann, *From enterprise architecture to IT governance*. Springer, 2006.

[6] R. Foorthuis and S. Brinkkemper, "Best Practices for Business and Systems Analysis in Projects Conforming to Enterprise Architecture," *Enterprise Modelling and Information Systems Architectures*, vol. 3, no. 1, pp. 36–47, 2008.

[7] F. Ahlemann, E. Stettiner, M. Messerschmidt, and C. Legner, Eds., *Strategic Enterprise Architecture Management*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[8] P. Andersen, A. Carugati, and M. G. Srensen, "Exploring Enterprise Architecture Evaluation Practices: The Case of a Large University," in *2015 48th Hawaii International Conference on System Sciences*, Jan. 2015, pp. 4089–4098.

[9] M. Pulkkinen, "Systemic Management of Architectural Decisions in Enterprise Architecture Planning. Four Dimensions and Three Abstraction Levels," in *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06)*, vol. 8, Jan. 2006, pp. 179a–179a.

[10] G. Riempp and S. Gieffers-Ankel, "Application portfolio management: a decision-oriented view of enterprise architecture," *Information Systems and e-Business Management*, vol. 5, no. 4, pp. 359–378, 2007.

[11] M. Langermeier, C. Saad, and B. Bauer, "A unified framework for enterprise architecture analysis," in *18th IEEE International EDOC Conference Workshops*, 2014, pp. 227–236.

[12] M. Langermeier and B. Bauer, "Generic EA Analysis Framework for the definition and automatic execution of analyses," in *International Conference on Enterprise Information Systems, ICEIS 17*, 2017.

[13] F. Lautenbacher, P. Diefenthaler, M. Langermeier, M. Mykhashchuk, and B. Bauer, "Planning Support for Enterprise Changes," in *The Practice of Enterprise Modeling*, ser. LNBIP. Springer, Berlin, Heidelberg, 2013, pp. 54–68.

[14] P. Diefenthaler and B. Bauer, "From Gaps to Transformation Paths in Enterprise Architecture Planning," in *Enterprise Information Systems. 15h International Conference, ICEIS 2013, Angers, France, July 4-7, 2013, Revised Selected Papers*, vol. 190, 2014.

[15] M. Langermeier, C. Saad, and B. Bauer, "Adaptive approach for impact analysis in enterprise architectures," in *Business Modeling and Software Design*. Springer, 2014, pp. 22–42.

[16] S. H. Spewak and S. C. Hill, *Enterprise Architecture Planning: Developing a Blueprint for Data, Applications and Technology*. Wellesley, MA, USA: QED Information Sciences, Inc., 1993.

[17] S. H. Spewak and M. Tiemann, "Updating the enterprise architecture planning model," *Journal of Enterprise Architecture*, vol. 2, no. 2, pp. 11–19, 2006.

[18] M. Pulkkinen and A. Hirvonen, "EA Planning, Development and Management Process for Agile Enterprise Development," in *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, 2005, pp. 223c–223c.

[19] S. Aier and J. Saat, "Understanding processes for model-based enterprise transformation planning," *International Journal of Internet and Enterprise Management*, vol. 7, no. 1, p. 84, 2011.

[20] R. Foorthuis, F. Hofman, S. Brinkkemper, and R. Bos, "Compliance Assessments of Projects Adhering to Enterprise Architecture," *Journal of Database Management*, vol. 23, no. 2, pp. 44–71, 2012.

[21] S. Aier and B. Gleichauf, "Applying design research artifacts for building design research artifacts: A process model for enterprise architecture planning," *Global Perspectives on Design Science Research: 5th International Conference, DESRIST 2010, St. Gallen, Switzerland, June 4-5, 2010. Proceedings.*, pp. 333–348, 2010.

[22] J. Rauscher, M. Langermeier, and B. Bauer, "Classification and definition of an enterprise architecture analyses language," in *Business Modeling and Software Design*, ser. LNBIP, no. 275. Springer, 2017.

[23] C. Saad and B. Bauer, "Data-flow based Model Analysis and its Applications," in *Proceedings of the 16th International Conference on Model Driven Engineering Languages and Systems (MoDELS'13)*, September 2013.

[24] S. Aier and B. Gleichauf, "Application of enterprise models for engineering enterprise transformation," *Enterprise Modelling and Information System Architectures*, vol. 5, pp. 56–72, 2010.

[25] F. de Boer, M. Bonsangue, L. 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.

[26] The Open Group, "ArchiMate 2.1," Tech. Rep. Open Group Standard, 2012.

[27] W. A. Bruls, M. van Steenbergen, R. M. Foorthuis, R. Bos, and S. Brinkkemper, "Domain architectures as an instrument to refine enterprise architecture," *Communications of the Association for Information Systems*, vol. 27, pp. 517–540, 2010.

[28] F. Matthes, I. Monahov, A. Schneider, and C. Schulz, "EAM KPI Catalog," Technical University Munich, Tech. Rep. v 1.0, 2011.

[29] F. Matthes, S. Buckl, J. Leitel, and C. M. Schweda, "Enterprise architecture management tool survey 2008," Technical University Munich, Tech. Rep., 2008.