Interactive Roadmap Generation in Enterprise Architecture Planning

Philipp Diefenthaler[1,2]

[1] University of Augsburg, Augsburg, Germany
[2] Softplant GmbH, Munich, Germany
philipp.diefenthaler@softplant.de

Abstract. Enterprises need to adapt to changes with an increasing speed in order to stay competitive. Many tools support modelling states of enterprise architectures at different points in time but lack a support in the generation of different transformation paths from one state to another. Enterprises know their goals and are able to model them, but are not supported in sequencing and scheduling the alternative ways towards achieving the goal. We want to improve the manual creation of roadmaps in enterprise architecture planning by providing enterprise engineers with suggestions (1) how to detail a target state of an enterprise architecture, (2) what sequences of change activities are possible to achieve a desired target state and (3) how these change activities can be scheduled into a roadmap. The focal point of the suggestions is the changes that take place in the application architecture which are affected by the environmental influences of the business and technology architectures.

Key words: transformation planning, enterprise architecture management, knowledge-based planner

1 Introduction

The dynamics in the environment of an enterprise and within an enterprise represent a challenge for enterprise architecture (EA) planning. Roadmaps are used to describe the path of change from the current towards the target state and are especially influenced by the dynamics. The thesis at hand tries to improve the roadmap generation with tool support in an interactive way. The focus of the thesis is on roadmaps for applications, application components and application services, taking into account the context of the business and technology that influence the changes of the latter. Objectives of the thesis are creation of a transformation model during the detailing of a target state, the possible sequencing of changes that can be derived, and the scheduling of a roadmap taking resources and time into account. Whereas the transition states need to be productive and further transition states can be suggested. Furthermore, the creation of the roadmap in interaction with the EA Planner in an existing architecture process by is an objective of the thesis at hand.

The research methodology applied in this thesis is design-science research [1] (DSR) and follows the DSR process as proposed by Peffers et al. [2]. According to this process at first the problems should be identified and a motivation for the research have to be posed. Section 2 introduces the identified problems and presents the resulting research questions. Afterwards the objectives of a solution should be posed to describe what a better artifact accomplishes. Therefore the proposed solution and the objectives of it are described in Section 3. Furthermore the artifacts the thesis is developing are introduced. Section 4 presents related work from the topics of enterprise architecture planning and semantic web based planning systems. Section 5 clarifies the state of the thesis, by presenting first achievements and future plans for the thesis. The future plans also describe how the developed artifacts are going to be evaluated.

2 Problem Formulation

Enterprise architecture planning is a multi-stakeholder effort to change the enterprise and its underlying technical artifacts in a controlled way. There is a lack of formalization in models to allow a division of planning activities ([3], p. 56). Furthermore, the support for modelling different paths of development and the derivation of 'transformation steps' is currently not supported in a feasible way ([4], p.21 f.). A transformation step is an intermediate step in the transformation path from a current towards a target state. An important challenge in planning is also the support of the transformation of models from a current to a target state ([5], p. 299).

Many best-practices and insights are available but no computer aided enterprise engineering tool for detailing a target state and supporting the sequencing and scheduling of change activities exists. The importance of successor relationships between elements in EA models has been shown by Aier and Gleichauf [6]. However, only successor relationships on the same level of abstraction are considered and cyclic dependencies, which may arise on different levels of abstraction in transformation planning, are not discussed. Moreover, the elaboration of a transformation model which captures relevant aspects for planning and taking dynamic changes into account is important and has not been addressed by research yet ([6], p. 69).

Sousa et al. [7] take the importance of projects into account and provide several viewpoints for applications, but do not show how the derivation of projects from necessary changes is possible in the first place.

In order to be able to demonstrate the suitability and applicability of better artifacts a prototypical instantiation within a tool, which can be evaluated, will be designed. Such a tool allows to detail planning where it is necessary and allows the development of alternative transformation paths. The level of detail that can be accomplished is tightly coupled to the information available in the models of the current and target state. The generation and addition of details in models of the target state is a manual task and will also be supported by a tool to allow a replanning as unplanned changes occur during the implementation of the transformation.

2.1 EA Transformation Model for Planning and Scheduling

An EA model specifies the concepts of the architecture. It allows to describe the state of an EA with concrete objects [8]. The current state is the modelled EA, which serves as the planning basis. Whereas the target state is a desired future model of the EA. In contrast to an EA model a transformation model relates elements of an EA model at a point in time to elements at another point in time via successor relationships ([6], p. 63). A successor relationship relates exactly one element of one state with an element in a successor state. The successor relationship does not mean that it replaces an element. It is also used to relate an element, which remains unchanged between two states (see Aier and Gleichauf [6], p. 64, Figure 3). A roadmap is an "abstracted plan for business or technology change, typically operating across multiple disciplines over multiple years" ([9], chapter 3). Moreover, it describes the transformation path, in a qualitative and quantitative way, from the current to the target state ([10], p. 291).

Change activities may depend on each other and may restrict the execution sequence of changes. Therefore, it is important to take the dependency of changes into account. Dependent changes are changes that influence each other and cannot be planned separately. Changes have an influence on each other if they have elements in the current or target state in common. Additionally, Aier and Gleichauf ([6], p.64) define dependent elements in one state if they have a relationship to each other. To detect dependent changes between two states of an EA it is necessary to know the successor relationships between elements of the current and target state. Figure 1 shows the successor relationships between elements and the dependencies between elements in one state. For example, element A and B both have the successor C in the target state. Furthermore, A and B do not have any further successors and C does not have any further predecessors, then A, B and C are part of a dependent change. A and B depend on other elements but these dependencies are adopted by the successor C.

As element E and F from Figure 1 remain unchanged A, B and C form a dependent change. Moreover, D forms a dependent change on its own, as it is retired without replacement and no other elements depend on D (see Figure 1). Therefore, information if an element has no successor, no predecessor or is unchanged is important. Dependent changes in one state are also given by the relationships between elements that are to be changed.

Three options how to get the successor relationships between elements of the current and target state exist. One is to allow the user to model them manually. This is a huge effort for example for application components and their coupling via services and/or interfaces in large application landscapes. A second option is to insert the successor relationships as the user models the target state in detail and by adding the information of the successor relationships automatically. A user can decide to use C instead of A and B in the target state, which implies that C is a successor of A and B. The third option is a completely automatic derivation of the relationships based on a modelled current and target state.

The information of changed relationships between the current and target state is important for planning but does not necessarily have to be part of a transformation model if the elements of the current state can be distinguished from the elements of the target state. The research question that stems from these points is: *Which preliminaries and extensions to transformation models are necessary to allow an interactive roadmap generation?*

In order to enable a sound scheduling it is furthermore necessary to be able to take resources and time into account. Resources are the different things an enterprise needs to successfully enact the change. Time is important to consider as it allows a sound scheduling, regarding the use of resources for certain time spans and their release after a change has been successfully implemented. Therefore the second research question is: *How can resources and time be considered to allow the scheduling of a roadmap?*

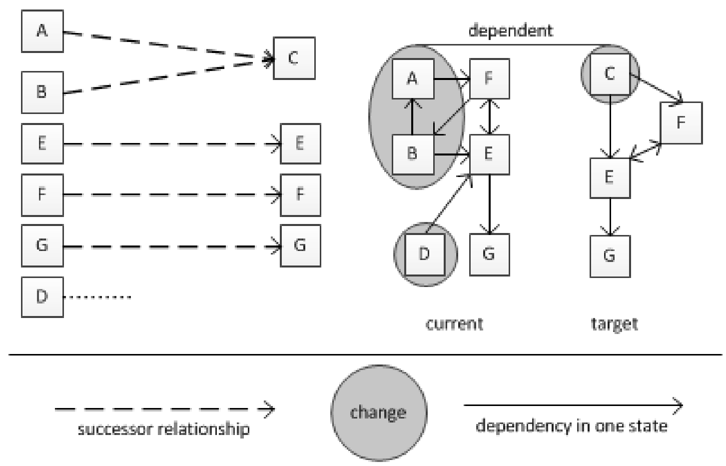

Fig. 1. Dependent Changes

2.2 Changes between States of an Enterprise Architecture

Elements in models of different states may change over time which means that they are at first conceived, then developed, then in use and then retire. Some elements may stay and undergo a change over time. The general pattern of change is the creation of an element and the deletion (see [11], p.95; [12], Appendix C, p.15; [13], p. 59; [7]; [10], p.289). All changes that occur between states of an EA can be derived from them. For example, taking the element A, B and C from above. This change corresponds to create C, delete A and delete B. If an element is changed, e. g. S-old becomes S-new, this can be expressed by deleting S-old and creating S-new.

The relationships between the elements can also change over time and impose an constraint on the creation and deletion of elements. For example, it is not possible to retire an element A as long as any other element depends on element A. Therefore, the change of the relationships between elements is also important information in planning changes. How the differences between two states can be derived has been discussed in literature under the terms gap-analysis and delta-analysis (c.f. [14]). However, the focus in literature is always on the differences of elements whereas the differences of the relationships are not taken into account. Therefore the third research question is: *How can the differences of relationships between elements in the current and target state be derived to support the roadmap generation?*

Moreover, it is necessary to describe the changes in a way to allow a tool support for the stakeholders involved in planning. The fourth research question is: *How can the changes (actions) in enterprise architecture be formalised?* Engels et al. [10] define reference scenarios as a sequence of practice proven changes in an application landscape, which have proven their applicability for certain gaps between current and target application landscapes. Reference scenarios represent a fixed sequence of actions from a current state followed by several transition states to a target state. TOGAF [9] and ArchiMate [15] use the term *transition architecture* for transition states. The fifth research question that is derived from the above issues: *Can reference scenarios be formalised to allow the suggestion of intermediate steps and states?* Finally, the sixth research question addresses the applicability of the artifacts that are to be developed within the thesis: *How can the stakeholders interactively develop a roadmap with a planning component?* It is possible that the method suggested by Aier and Gleichauf [6] can be reused or may need adaptations.

3 Proposed Solution

The focus of the proposed solution is on applications, their compositions of application components, as well as their implemented and used application services. Furthermore, the business and technology context of these elements is taken into consideration. Figure 2 shows the *planning component* in context with the EA Planner, as well as preliminaries and results of the proposed solution. The transformation model is not depicted in the figure. The artifacts that are designed are a model artifact as an extension to a transformation model to connect method and model independent models of the current and target state. Moreover, a method and model artifact will be designed to allow the creation of the action repository. The method artifact is used to describe how an action repository can be developed and how the different actions can be specified. Whereas, the model artifact for the action repository is used to describe the changes between two states of a model. A further method artifact will be necessary to integrate the proposed solution in existing planning processes. The planning component itself is also an artifact, i.e. an instantiation artifact, and will be used for the planned evaluation (see Section 5).

The different types of states that have to be distinguished in planning are *current state*, *target state* and *transition states*. The current state represents

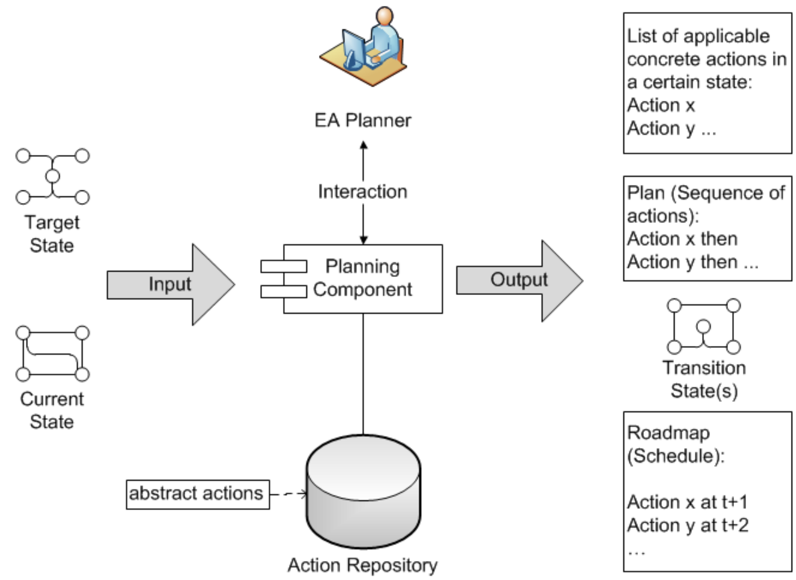

Fig. 2. Planning Component in context

the elements and relationships at present, whereas the target state represents a desired future state. Transition states are those states that can be produced during the roadmap generation as a different sequence of actions may lead to different transition states.

The *EA Planner* is the user of the planning component and can interact with the planning component to create a roadmap. Not depicted in Figure 2 is the knowledge engineer who creates the formal description of the *abstract actions* in the *action repository*. These formal descriptions are designed according to the EA Planners requirements on the changes that have to be part of a *roadmap* and that have to be supported by the planning component. An abstract action describes the preconditions and the effects of an action in general, for example, for the development of a new application service. A concrete action relates to concrete objects represented in the model and changes the state.

The action repository contains three different types of abstract actions. These types are *detailing actions*, *sequencing actions* and *debugging actions*. Latter are relevant for the knowledge engineer to check whether the planning component creates only transition states, which are allowed according to the constraints posed by the EA Planner. For debugging also test cases are necessary, which contain all possible combinations of successor relationships, to be able to check if the plan generation is possible and no debugging action can be taken along the transitions from the current to the target state. An example for a debugging action is the action that can be enacted if an element of the type application component uses an element of the type application service, which is not live. Not live means in this context that the service is not provided by an application and thus cannot be used.

This allows the planning component to detect states that are inconsistent. For an EA Planner debugging actions are not important for roadmap generation, but he can express illegal conditions which the knowledge engineer can then form in abstract debugging actions. Furthermore, it is necessary to ensure that only actions can be selected by the EA Planner which lead towards the target state.

Detailing actions are those actions that help the planning component to create suggestions how to detail the model of the target state. Moreover, the detailing actions will add the successor relationships and further sequencing constraints, if specified by the EA Planner, to the transformation model. Sequencing actions are those actions that allow the planning component to create a sequence of concrete actions in interaction with the EA Planner.

A *plan* is, in contrast to a *schedule*, a sequence of concrete actions that is not taking resources and time into consideration. The roadmap contains the sequence of actions which then can be related to points in time, start and end time of actions, or to time horizons if no accurate estimations are possible to the EA Planner. Furthermore, it should be possible to take resources into account, as for example budget or IT staff limitations may pose additionally constraints on the scheduling of actions.

3.1 Enterprise Models based on OWL

The Web Ontology Language (OWL), latest version number 2 [16], is a recommendation of the World Wide Web Consortium (W3C)[1] for an ontology language. In OWL classes, properties, individuals and data values can be specified to describe the building blocks of an EA and their states. Using OWL ontologies has the benefit that reasoning techniques can be applied on the formal models and thus allows a system to draw conclusions. For example, if the same business process is supported by two different applications in the current and target state, it is possible to conclude that the application in the target state is a successor of the application in the current state. These mechanisms will be used in the detailing of the target. Furthermore, the information from several federated sources can be combined to allow the reuse of existing information [17]. The integration of relational databases is possible to allow a reuse of existing data, but the integration aspect is not part of the thesis.

The application of OWL for EA modelling is described by Allemang [18] for the Federal EA reference models and by Gerber [19] for the TOGAF Content Metamodel. Tools that already use OWL for EA management are the TopBraid Suite[2] and the Essential Project Manager[3], which is based on the Protege ontology editor of the Stanford University[4]. OWL ontologies can be represented using

[1] http://www.w3.org
[2] http://www.topquadrant.com/docs/whitepapers/WP-BuildingSemanticEASolutions-withTopBraid.pdf
[3] http://enterprise-architecture.org/
[4] http://protege.stanford.edu/

the Resource Description Framework (RDF) [20] which allows to represent the ontology in a special XML Schema. Furthermore, ontologies represented in RDF are graphs consisting of triples. It is possible to query models represented in RDF by using SPARQL Query Language for RDF (SPARQL) [21]. OWL ontologies are taken as a starting point to model enterprise models and the transformation model as they provide a formally defined meaning and support reasoning mechanisms.

3.2 Interaction of an EA Planner and the Planning Component

Regarding the level of detail the models of the current and target state differ at the beginning. A detailed model consists of application components and their implemented and used application services. Furthermore, the models provide information about the environment of the applications, i. e. their business and technological context.

The following scenario is presumed: a detailed model of the current state of an enterprise's application architecture and a coarse model of the target state are available. A coarse model of the target state is, for example, a business support map with the future business support of applications leaving out the details of which application components it will consist of.

Detailing the Target State and Setting Successor Relationships The planning component guides the EA Planner through the detailing process of the target state. At first the planning component creates from the abstract actions *create new application component*, *retain one application component* or *retain several application components*, applied on the current state, concrete actions. The EA Planner may need additional informations to decide which of the suggestions is the most beneficial. Additionally, the planning component sets the successor relationships, within the transformation model, in the background as the EA Planner makes the decisions which application components have to be built or retained. If a new application component is selected to be built the planning component will suggest to implement and use the services implemented and used by the predecessor. The EA Planner may follow these suggestions or adjust them to his needs.

At this point the EA Planner does not specify the temporal sequence of changes. He is detailing the information to allow a sound sequencing and scheduling later on. All parts that are not specified in interaction between the EA Planner and the planning component remain the same as in the detailed current state, as no changes are specified. As a *result* a detailed target state and the successor relationships between elements of the current and target state are available. The transformation model contains the successor relationships at the end of this phase.

Creating detailed Plans Based on the detailed current state, target state and information on the successor relationships it is possible for the planning component to generate an *initial state* and *goal state* for planning. These states

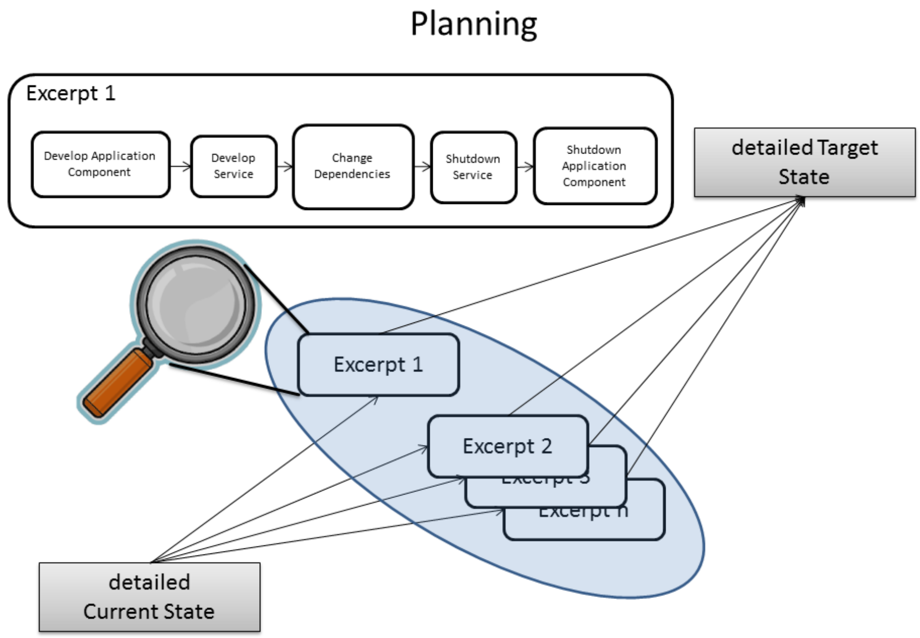

Fig. 3. Excerpts of changes to create detailed Plans

are different from the current and target state. The initial state contains all application components and application services with their implementation and usage dependencies in the current state. Furthermore, the successor relationships and their planned application components and application services are part of the initial state. In contrast the goal state contains information about all application components and application services that are retired and developed in the target state, including implementation and usage dependencies in the target state.

The EA Planner selects an excerpt for detailed planning. An excerpt can be a bundle of dependent changes, formed by the successor relationships of elements of the current and target state, or a set of bundles defined by the EA Planner. The initial and goal state are important for the planning component to compute the applicable concrete actions, the resulting state and a sequence of concrete actions towards the goal state. Figure 3 shows how an excerpt may look in detail. Excerpts are not alternative transformation paths from the detailed current towards the detailed target state. If all excerpts are planned, then all changes necessary to transform the current state to the detailed target state are considered within a plan. The EA planner may decide to plan only certain excerpts in detail.

Moreover, the EA Planner may decide to add further transition steps to sequence a change in accordance to a reference scenario. If reference scenarios can be formalised in an appropriate way this modelling can also be supported by the planning component (research question 5). If not, the EA Planner will have the choice to model these details, but with a notification that the models of the transition states need to be modelled carefully, e.g. no application service is shut-down as long as there exist dependencies on that service.

Excerpt 1 in Figure 3 shows the sequence of abstract actions in detail for the case of one application component and service replacing one application component and service of the current state. The abstract actions necessary to complete excerpt 1 are: *develop application component*, *develop service*, *change dependencies*, *shutdown service* and *shutdown application component*. *Change dependencies* is used to change the *uses* relationships of application components to the successor service. Concrete actions change application components and services that have to be developed or shutdown in the transformation from the current to the target state. The abstract actions are applicable to every excerpt and only differ in the concrete objects they are applied to.

The EA Planner can add additional temporal constraints from one excerpt to another or from one concrete action of an excerpt to the concrete action of another excerpt. For example, this allows specifying that a certain application component should or must be built before another application component is built. Moreover, the EA Planner can add information about the resources and time necessary to implement the changes, if available. As a result of this phase the sequences of actions for excerpts were modelled and information on time and resources is available, if specified.

It is also possible to detect if constraints that have been posed cannot be sequenced. This may be the case if the EA Planner specified for example that an application A has to be built before application B and furthermore the EA Planner, another person in the role of the EA Planner, specified in a different excerpt that B has to be built before A.

Creating the Roadmap If all constraints imposed by the EA Planner can be satisfied the planning component can schedule a roadmap with the information gathered from the creation of the detailed plans. The planning component has to take the durations of the activities and the resources into account. Furthermore, if any additional temporal constraints have been posed by the EA Planner during the *Creating Detailed Plans* phase the planning component has to take them also into account. An optimization of the roadmap taking further needs of the EA Planner into account is favourable. For example, the planning component can schedule the shortest roadmap, regarding the timespan and considering resource constraints, or taking urgency and importance of changes into account.

4 Related Work

4.1 EA Planning

Spewak [22] introduces an EA planning model (wedding cake model) which describes how the blueprints for the target state are developed from the analysis of the current state and how the change is structured in an implementation and migration plan.

Postina [13] presents a method to manage service and process oriented enterprise architectures. He uses a Case Based Reasoning approach and a case repository to provide information for the evolution of the enterprise architecture. The target state can be reached from the current state through several evolutionary steps. A case is defined by the type of an evolutionary step (create, update, delete), the element and type involved, e.g. organizational unit billing, and the viewpoints attached to it. The cases help to provide views for future evolutionary steps with the same combination of element type and evolutionary step type. Sequencing and scheduling activities is not considered by the method.

Postina and Gringel [23] present a prototypical tool for closing the gaps between an ideal and current application landscapes. As a result a target landscape can be interactively designed by an enterprise architect and the tool. The term ideal landscape is tightly coupled to the Quasar Enterprise approach which considers domains, ideal interfaces, ideal components and ideal operations for interfaces. Based on the structural differences between the current and ideal landscape the tool can identify the gaps and provides an action list to close the gaps. The modelling of an ideal landscape is a preliminary and a sequencing and scheduling is not considered within the tool.

Sousa et al. [7] describe an approach to reconstruct enterprise architecture models from existing project information, instead of deriving projects from the models. Furthermore, the Blueprint Management System (BMS) is introduced which allows to detect the temporal dependencies of projects on certain artifacts and can generate different viewpoints. Time is explicitly taken into account by providing a timeline bar in the BMS which enables the user to browse through the different points in time. The tool is capable of providing different states as each project has a list of artifacts it creates (Project.aliveList) and deletes (Project.deadList).

4.2 OWL and Enterprise Architecture in combination

Allemang et al. [18] describe how the reference models of the Federal Enterprise Architecture (FEA) can be represented in OWL. The FEA reference models consists of the five reference models for performance, business, service component, technology and data. Gerber et al. [19] describe how the TOGAF Content Metamodel can be formalized within an OWL ontology. Furthermore, they show how ontology technologies can be used to detect inconsistencies and ambiguities in the model provided by The Open Group.

4.3 Semantic Web Planners

Bouillet et al. [24] describe a knowledge engineering and planning framework based on OWL ontologies. They demonstrate the applicability of the approach for the creation of workflows in the domain of stream processing. Goals are represented as graph patterns which have to be satisfied. It is possible to specify them as graph patterns in OWL.

Liu et al. [25] present their Semantic Web based planning system OntoPlan, which helps a customer to configure a personal computer with components that work together and can be purchased. OntoPlan uses the DARPA Agenet Markup Language (DAML) to represent the knowledge for the planning system. The Web Ontology Language OWL is a successor language of DAML. Users can specify the goal and constraints for the configuration of a personal computer of their choice. Furthermore, users can browse the initial state, goal state and the actions. OntoPlan uses a partial-order planning approach where a plan is generated from several partial plans. Actions are for example 'SelectDisplayCard' to allow the planning system to check whether there is already a display card in the configuration and if not what different display cards are compatible to the other selected components of the personal computer.

5 State of the Thesis, Achievements and Future Plans

We finished the literature research on the state of the art in enterprise architecture planning and knowledge-based planning in artificial intelligence relevant for the thesis' topic. Furthermore, we identified the problems and defined the objectives of the proposed solution.

The first result that has been achieved is tackling the third research questions. It shows how all differences between two states modelled in OWL can be derived automatically. We modelled at first a coarse current state and target state of an application architecture based upon examples provided from Hanschke [26]. Hanschke uses business support maps to plan the changes in information systems. We adopted the procedure for a proof of concept to derive successor relationships automatically. For each cell within the business support map, consisting of business processes (x-axis) and customer groups (y-axis), the applications localized in a cell of the current target state were made successors of each other's. Based on the successor relationships and information detailing the current state with application services and information (data) objects it is possible to create suggestions for a detailed target state [27]. We modelled the states in OWL using the Protege ontology editor. The derivation of differences between the current and target state was conducted using the Protege Plugin OWLDiff [28], which can trace differences of any kind between two OWL ontologies.

The second result is still in progress and focuses on the generation of a sequence of actions using the graph transformation tool 'GROOVE' [5]. Given an initial state and goal state, as combination of current and target state it is possible to generate possible sequences of actions to transform the current into the target state. These results need to be worked out with further details and testing. An evaluation of the results is necessary and in planned. The detailing actions have not been modelled yet.

[5] http://groove.cs.utwente.nl/

As a next step the method how to develop an action repository and the actions it contains are elaborated in further detail, especially focusing on the formalization of reference scenarios. The identification of further temporal constraints that have to be considered in a transformation model, which allow a sequencing of changes will be designed afterwards. Resource and time aspects that have to be considered are addressed subsequently. Moreover, a method artifact, as an additional fragment for an existing method for roadmap generation, is crucial to show that the approach is capable to be connected with an existing architecture process. This artifact addresses research question 6: *How can the stakeholders interactively develop a roadmap with a planning component?* We favour the extension of the Architecture Development Method of TOGAF, as it has a broad acceptance in different industries and sizes of enterprises and is publicly accessible. Furthermore, an instantiation of the planning component will be built. The designed artifacts will be evaluated in case studies with industry partners with a focus on the instantiation; one automotive, one banking and one multi-industry enterprise.

References

1. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Quaterly (2004)
2. Peffers, K., Tuunanen, T., Gengler, C., Rossi, M., Hui, W., Virtanen, V., Bragge, J.: The design science research process: A model for producing and presenting information systems research. In Chatterjee, S., Hevner, A.R., eds.: Proceedings of the First International Conference on Design Science Research in Information Systems and Technology (DESRIST 2006). (2006) 83–106
3. Aier, S., Gleichauf, B., Saat, J., Winter, R.: Complexity levels of representing dynamics in ea planning. In Albani, A., Barjis, J., Dietz, J.L.G., eds.: Advances in Enterprise Engineering III. Springer Berlin Heidelberg, Berlin and Heidelberg (2009) 55–69
4. Gleichauf, B.: Planung der Unternehmensarchitektur: Beiträge zur Entwicklung einer Methode für die Planung der Transformation von Unternehmensarchitekturen auf Basis von Unternehmensarchitekturmodellen und unter Berücksichtigung dynamischer Aspekte. PhD thesis (2011)
5. Aier, S., Riege, C., Winter, R.: Unternehmensarchitektur – literaturüberblick und stand der praxis. Wirtschaftsinformatik **50**(4) (2008) 292–304
6. Aier, S., Gleichauf, B.: Application of enterprise models for enginnering enterprise transformation. Enterprise Modelling and Information Systems Architectures **5**(1) (2010) 56–72
7. Sousa, P., Lima, J., Sampaio, A., Pereira, C.: An approach for creating and managing enterprise blueprints: A case for it blueprints. In Albani, A., Barjis, J., Dietz, J.L.G., eds.: Advances in Enterprise Engineering III. Springer Berlin Heidelberg, Berlin and Heidelberg (2009) 70–84
8. Lankhorst, M.M., van der Torre, L., Proper, E., Arbab, F., Boer, F.S.d., Bonsangue, M.: Foundations. In Lankhorst, M.M., ed.: Enterprise architecture at work. Springer, Berlin (2009) 49–68
9. The Open Group: TOGAF Version 9. Van Haren Publishing, Netherlands (2009)

10. Engels, G., Hess, A., Humm, B., Juwig, O., Lohmann, M., Richter, J.P.: Quasar Enterprise: Anwendungslandschaften serviceorientiert gestalten. 1 edn. dpunkt.verlag, Heidelberg (2008)
11. Keller, W.: IT-Unternehmensarchitektur: Von der Geschäftsstrategie zur optimalen IT-Unterstützung. 1 edn. dpunkt.verlag, Heidelberg (2007)
12. Hanschke, I.: C planungs-muster: Download-anhang zum buch strategisches management der it-landschaft (2012)
13. Postina, M.: Evolutionsmanagement prozess- und serviceorientierter Unternehmensarchitekturen. PhD thesis, Univ, Edewecht and Oldenburg (2012)
14. Postina, M., Sechyn, I., Steffens, U.: Gap analysis of application landscapes. In: 2009 13th Enterprise Distributed Object Computing Conference Workshops, IEEE (2009) 274–281
15. The Open Group: Archimate 2.0 Specification. Van Haren Publishing (2012)
16. Motik, B., Patel-Schneider, P. F., Horrocks, I.: Owl 2 web ontology language: Structural specification and functional-style syntax (2009)
17. Chen, W., Hess, C., Langermeier, M., Stülpnagel, J.v., Diefenthaler, P.: Semantic enterprise architecture management. In: Proceedings of the 15th International Conference on Enterprise Information Systems. to be presented at ICEIS 2013, 4-7th July 2013, Angers France.
18. Allemang, D., Polikoff, I., Hodgson, R.: Enterprise architecture reference modeling in owl/rdf. In Gil, Y., Motta, E., Benjamins, R.V., Musen, M.A., eds.: The semantic web - ISWC 2005. Volume 3729 of Lecture Notes in Computer Science. SpringerLink [host], Berlin (2005) 844–857
19. Gerber, A., Kotzè, P., van der Merwe, A.: Towards the formalisation of the togaf content metamodel using ontologies. In Filipe, J., Cordeiro, J., eds.: ICEIS 2010. Volume 2. SciTePress, [S. l.] (2010) 54–64
20. Manola, F., Miller, E., McBride, B.: RDF Primer. World Wide Web Consortium (2004)
21. Prud'hommeaux, Eric, Seaborne, Andy: SPARQL Query Language for RDF. World Wide Web Consortium (2008)
22. Spewak, S.H., Hill, S.C.: Enterprise architecture planning: Developing a blueprint for data, applications, and technology. John Wiley & Sons, New York (1992)
23. Gringel, P., Postina, M.: I-pattern for gap analysis. In Engels, G., Luckey, M., Pretschner, A., Reussner, R., eds.: Software engineering 2010. Lecture Notes in Informatics. Gesellschaft für Informatik, Bonn (2010) 281–292
24. Bouillet, E., Feblowitz, M., Liu, Z., Ranganathan, A., Riabov, A.: A knowledge engineering and planning framework based on owl ontologies. In: International Conference on Automated Planning and Scheduling 2007
25. Liu, J., Wong, C.K., Tsang, K.H.: Ontoplan: semantic web based planning. In: 2005 International Conference on Active Media Technology, 2005. (AMT 2005). (2005) 568–573
26. Hanschke, I.: Strategisches Management der IT-Landschaft: Ein praktischer Leitfaden für das Enterprise Architecture Management. 1. edn. Hanser, München (2009)
27. Diefenthaler, P., Bernhard Bauer: Gap analysis in enterprise architecture using semantic web technologies. In: Proceedings of the 15th International Conference on Enterprise Information Systems. (2013) to be presented at ICEIS 2013, 4-7th July 2013, Angers France.
28. Kremen, P., Smid, M., Kouba, Z.: Owldiff: A practical tool for comparison and merge of owl ontologies. In: 22nd International Workshop on Database and Expert Systems Applications, IEEE (2011) 229–233