Constructing an Enterprise-specific Radar System for Assisted Project Surveillance

Sabine Buckl, Alexander M. Ernst, Florian Matthes,
Christopher Schulz, Christian M. Schweda

{buckls,ernst,matthes,schulzc,schweda}@in.tum.de

Abstract: Since information technology (IT) plays an increasingly important role for modern enterprises, the number of running IT projects raises steadily. These days, an enterprise architect is confronted with a multitude of projects he or she has to supervise in order to avert potential project threats. As a matter of fact, the number of projects that are to be supervised exceeds the available architects many times over, which makes it nearly impossible for the architect to monitor every single project in detail. To overcome this deficiency, we propose the application of an architecture radar system, which continuously scans ongoing projects for the occurrence of threats arising from specific architectural environments. Whenever an architectural threat is detected by a scan, the responsible enterprise architect is informed to carry out a monitoring of the endangered project. In this paper, we present a method to construct such an architecture radar system. We then expatiate on different categories of threats, which we derive from a holistic model of the enterprise architecture.

1 Motivation

Nowadays, application landscapes represent strongly interwoven systems with hundreds or even thousands of business applications, which are tightly interconnected. The growing number of business applications and their relations to supported business processes, responsible organizational units, hosting infrastructure elements, etc. results in a continuously increasing complexity, which needs to be detected, managed, and adequately mastered. Enterprise architecture (EA) management is a commonly accepted means to support the managed evolution of the constituents of an enterprise and to obtain transparency regarding their interdependencies [Fra02, Sch08, The09, WBFK07]. Thereby, the main goal of EA management is to support the transformation of the enterprise and increase the alignment between business and information technology (IT) [HV99, Luf03].

The transformation of the enterprise is typically performed via implementing projects originating from business as well as IT demands. These *transformation projects*, which have to cope with the above described complexity of the EA, represent large investments for companies. The current chaos report of the Standish Group [Int09] revealed, that only 32% of all IT projects are successfully completed. Thereby, a project is considered to be failed, if the realization has not been achieved on time, budget, or with required features, and functions [Int09]. This perception of *project failure* is more stringent than understanding

a project to be failed, only if it has been stopped prior to its completion. Using the more rigorous definition, various projects threats, which might lead to budget overrun, time delay, underestimation of overall costs, or in the last resort to the withdrawal of the project could be identified in advance. A survey on IT projects in public authorities showed that among other things, the undervaluation of complexity of IT integration, missing standards, lack of fall back strategies, or a disadvantageous relation between time for preparation and time for realization represent major threats to IT projects [Mer09]. In this paper, we refer to the former threats, like missing standards or high architectural complexity, as *architectural threats* to emphasize that they arise from specific architectural environments. The later ones are called *project preparation and execution threats* as they originate from the project prearrangement and implementation itself.

In the context of EA management especially architectural threats for IT projects are considered. Thereby, threats do not pertain to threats regarding the architecture of a single business application but rather refer to threats, which might arise from the dependencies between different constituents of the EA. For instance, a project, which replaces a central business application, might be at risk due to the number of interfaces to depending applications, supported business processes, using organizational units, etc. From this perspective, it would be helpful to have a responsible enterprise architect for each project team, who provides additional guidance in challenging situations and who also helps to foresee difficulties caused by architectural modifications. Unfortunately, equipping every IT project with an experienced enterprise architect is not feasible, as the number of approved projects typically outweighs the number of available enterprise architects by far. In this respect, an enterprise architect needs to identify those projects requiring supplementary supervision.

We therefore propose an *architecture radar system*, which supports the detection of threatened IT projects based on architectural properties of the EA. The system determines which set of projects are most likely to run into difficulties due to an accumulation of potential architectural threats and should hence be investigated further by an enterprise architect. In this vein, architecture surveillance for an IT project is established as consisting of two basic mechanisms – *scanning* and *monitoring*. With the help of the architecture radar system, the current and scheduled projects are scanned, i.e., the system uses indicators and rules to analyze the *ex ante* and the *planned ex post* architecture associated to a project. In case a scan indicates an increased threat level, the second architecture surveillance mechanism, the project monitoring by an enterprise architect, is started. In this manner, the architecture radar system would represent the first architecture surveillance mechanism for projects. By constructing such a system, we provide the possibility to automate architecture scans. We consider this helpful as we expect that scanning a project once at its kick-off date is not fully sufficient to cover all architectural threats. Such threats may also evolve during project execution, e.g. when architecture elements surrounding the business application, which is affected by the project, are changed. Therefore, project scans should be either performed on a regular basis over the project's lifetime or should be executed demand driven, i.e., triggered by architecture changes. Summing up, the research questions of this paper can be stated as follows:

> How should an architecture radar system be constructed, which scans ongoing IT projects based on indicators and rules in order to identify those projects that

need to be monitored by an enterprise architect due to architectural threats? How can architectural threats for projects be identified, operationalized, and measured?

Preparing the answer for the above stated research questions, Section 2 gives an overview on existing literature in the domain of project threats. Furthermore, different methods for measuring architectural properties, e.g. by indicators or rule-based, are discussed. Subsequently, Section 3 presents an approach to construct an enterprise-specific architecture radar systems illustrated by a running example from the manufacturing industry. The exemplary threats are complemented in Section 4, which sketches a classification schema for architectural threats and points out additional threats. Finally, Section 5 provides a critical reflection of the findings and hints to further areas of research.

2 Related work

Project preparation and execution threats as well as derived indicators are largely covered by a variety of IT project management literature, where budgeting, scheduling, resource allocation, etc. is discussed elaborately (see e.g. [CG06, WM07, MP07]). In particular, approaches for objectively measuring project progress (cf. [FK06]) have been proposed. Similarly, the work presented in [Küt06] motivates and comprehensively defines indicators for project management, although solely project preparation and execution threats are described. Neither of the approaches further specifies details, when it comes to architectural threats, i.e., indicators shedding light on the threats emanating from the current and planned EA. Nevertheless, in [Küt06] the idea of a *radar system* for project surveillance is spawned and advocated for.

In many cases, the implementation of one IT project affects the architecture of several business applications as well as the overall application landscape. For assessing the quality of such an architecture, expert systems as described in [Jac98] may play a helpful role. By leaning against *rule-based inference* and *case-based reasoning*, one is able to evaluate the modifications an architecture is undergoing. Similar evaluation techniques might be applicable to the architecture of an enterprise, although an attempt to apply such techniques has not yet been undertaken.

Tying in with the usage of expert systems, the idea of *anti-patterns* like explained in [BMM98] should be evoked. By differentiating between managerial, architectural, and developmental anti-patterns, [BMM98] elaborates on common mistakes and problems in software development today. With these anti-patterns at hand, software development experts should be able to identify the occurrence of well-known problems in a development project. In contrast, a non expert-based technique (e.g. via using rules) for discovering the presence of an anti-pattern is not described. Nonetheless, the general idea of such patterns is very likely to be helpful for assessing architectural threats.

A prominent example for measures and indicators is the one of [KN91] motivating the use of a *balanced scorecard*. The article inspires managers to look on their businesses from

four essential perspectives – *financial*, *customer*, *internal process* as well as *innovation and learning*. Hence, the business aspect of the EA is strongly alluded to, while other EA aspects are not considered in detail. However, the authors suggest the concept of multiple perspectives including a strategic and visionary one, thereby measuring not only one single business factor but instead several measures.

One conventional type of an IT project is developing a new or maintaining an existing business application (cf. [WM07]), which is integrated in an existing application landscape. The article [AS08] details on a two-phase ex-ante approach on evaluating applications in the context of their vicinity. It determines the influence a certain business application has on the landscape using an aggregated indicator, the so called *indication*. With this focus on the application landscape as an important part of the EA, neither further aspects of the latter are examined nor projects and related indicators are discussed by the authors. Nonetheless, the article serves as substantive starting point, when approaching the question of quantitative assessment of EAs, since it does not solely consider internal quality of a single system's software architecture but takes a broader perspective.

[Lan08] proposes metrics as a quantitative technique to address the challenge of growing importance, size, and complexity of application landscapes. The author expatiates on the question of availability and failure propagation within application landscapes. In this context, also project proposals are assessed in respect to the impact of their changes on availability and failure propagation in the application landscape. Nevertheless, metrics for assessing how certain architectural properties can influence the execution of a project are not investigated.

The work of [FHKS08] outlines a general method for designing and utilizing indicator systems based on a modeling language named *Score-ML*. Thereby, the indicator systems can be embedded into an existing enterprise modeling approach – namely the one of *multi-perspective enterprise modeling*[Fra02]. The method complementing the Score-ML reflects a generic approach to establish an indicator system and its later utilization in a business setting. Since the paper originates from a business point of view, related but equally crucial elements of an EA are partially skipped. In particular, projects are not directly alluded to, such that the Score-ML method does not provide explicit pointers to set up an indicator system for the project surveillance purpose.

Another method for analyzing EAs is presented in [JJSU07]. Therein, relationships between properties of architecture constituents are introduced. These relationships are further employed to reason over the effects, a change in a property of an EA element might have. Making such relationships explicit can be helpful in modeling and analyzing EAs; the approach pursued thereby is quite similar to the one, presented by the Score-ML above, although the primary focus of [JJSU07] is on more technical properties, as availability. From this focus, it is not surprising that the project concept is not alluded to in the approach, although the mechanism of the *property relationship* can be applied on projects either.

None of above approaches copes with the question of elaborating and applying architectural indicators for project scanning and monitoring, although a number of highly interesting ideas are spawned. These ideas are taken into consideration in the following section,

where we describe how to determine relevant project threats, derive meaningful indicators as well as rules, and subsequently set up an architecture radar system scanning for projects, which are likely to fail due to architectural threats.

3 Exemplifying the construction of an architecture radar system

Below, we do not present one comprehensive architecture radar system for scanning IT projects, as we do not expect such *one-for-all* system to exist. In contrast, every enterprise might have its own indicators and rules contingent on the enterprise's culture and skill sets. Making the latter concepts explicit can be very helpful for architectural analyses, so we expect them to be documented in a project knowledge base related to potential project threats (including both architectural as well as project preparation and execution threats). Based on this knowledge about threats, an enterprise-specific architecture radar system can be constructed in an iterative process, of which Figure 1 gives an overview.

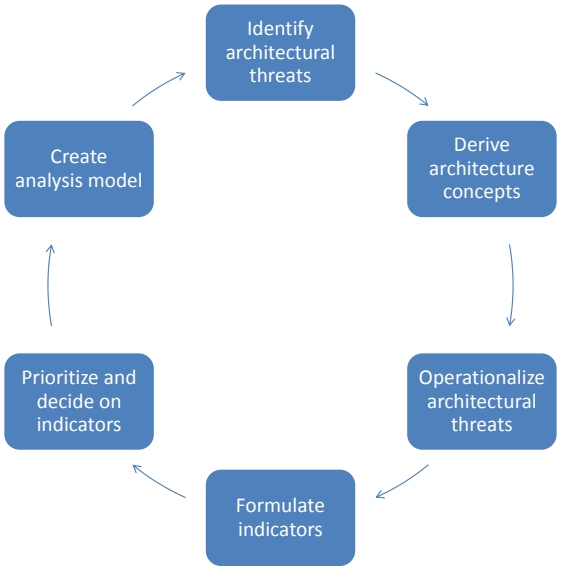

Figure 1: Iterative process for constructing an architecture radar system

The below described real world problem of a manufacturing company is used to briefly exemplify the single steps of our approach to constructing an enterprise-specific radar system.

> The manufacturing company experienced manifold challenges and problems during the execution of IT projects in the past. A majority of these problems emerged from the complex web of dependencies from affected business applications to surrounding elements, e.g. supported business processes, hosting

organizational units, etc. Furthermore, the manufacturing company experienced a decreasing number of failed IT projects, if these were monitored by an enterprise architect. As the enterprise architecture group contains only a few persons, not all IT projects can be monitored by them. Therefore, the manufacturing company wants to establish an architecture radar system to support the decision, which projects should be monitored by an enterprise architect.

Identify architectural threats
In this step, the architectural factors that impede the successful preparation and execution of an IT project are identified. This can be done in various ways, the least formal one being interviews with the managers of projects, which failed according to the definition provided in Section 1. During the interviews, especially the environment of the business applications, which were changed by the project, should be discussed, e.g. the interfaces to other applications or the provided support for business processes. Complementing the interviews, documentations of the EA before and after the project can be used to derive threats that might have hampered the implementation of the IT project. All threats identified thereby need to be documented, if not already present in a project knowledge base, documenting both success factors and potential project impediments. As far as not solely architectural aspects are concerned by a threat, we would advise to separate architectural and project preparation and execution threats clearly via different descriptions.

In order to identify potential architectural threats, an enterprise architect is instructed to analyze failed projects. Therefore, he/she inspects the documentation of multiple failed projects and interviews the project managers. The analyses revealed two main architectural threats for projects. Firstly, projects affecting a business application, which is strongly connected to others, are at risk due to their high impact on surrounding elements. Secondly, projects installing new business applications, which do not use standard technology (e.g. .NET instead of J2EE) are at risk due to missing experiences and skills as well as the larger technology stack to be taken care of afterwards. These two architectural threats – *complexity* and *standard compliance* – are documented by the enterprise architect in the knowledge base.

Derive architecture concepts
The architectural threats, elicited in step one, are very likely to refer to different constituents of the EA, among others most probably the business applications, if IT projects are considered. For constructing the architecture radar system, these EA constituents and their relationships to each other act as *reference objects*, i.e., the objects, where observable and derived properties are attached to. During this step, a list of relevant EA concepts is gathered from the descriptions of the architectural threats and is subsequently consolidated by removing synonyms and resolving homonyms. As a first artifact from this step, a glossary of relevant concepts is developed and compared to existing information, which is already present e.g. in an EA management tool used in the company or a CMDB. The

second artifact is an object-oriented information model[1] describing the architecture concepts as classes, attributes, and associations. This model should further employ the *IT project* concept and corresponding relationships, that can be used to distinguish between the architecture *before* and *after* the execution of the project. To support concise modeling of these project relationships, a stereotype, which indicates that a concept is changed by a project, can be utilized (cf. [BEMS09]).

Based on the documentation of the threats *complexity* and *standard compliance* in the knowledge base, the relevant EA concepts and their relationships are derived and the subsequent glossary is developed:

Business application is a version of a software system within an organization, which contributes to the business process support.

Interface refers to a set of functionality offered by a business application for use by other business applications.

Project affectable is an abstract class, which makes it possible via inheritance to assign a project dependency to a concept in the information model[2].

IT project is a planned activity concerned with introducing or retiring one or more business applications[3].

Technology represents a technical constituent of a business application, ranging from an implementation framework or platform to a database management system or a user interface toolkit. Exemplarily spoken, technologies may be .NET, Apache 2.0.53 or Oracle 9.2i.

Figure 2 illustrates the object-oriented information model, which describes the relations between the concepts. The information model enables the modeling and storage of information before and after the execution of the project.

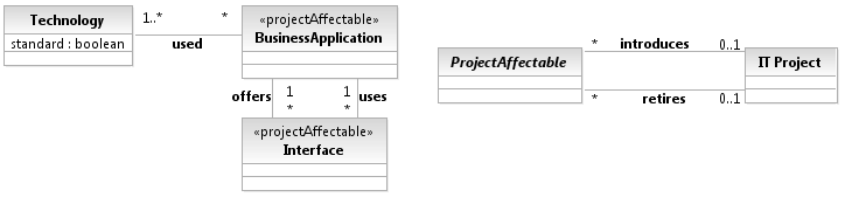

Figure 2: Object-oriented information model of the derived concepts

[1] We advocate for the idea to use an object-oriented model as most EA management approaches and tools typically use such models (cf. [AW09, EHH+08, The09]).

[2] As introduced in [BEMS09], we use the stereotype `<<projectAffectable>>`, which can be assigned to a class in order to indicate that this class is actually a subclass of project affectable.

[3] The update of a business application is treated as the introduction of a new version and the retirement of the old one.

Operationalize architectural threats
The object-oriented information model for describing the architectural concepts related to potential architectural threats is up to this point purely qualitative. To make architectural threats explicit, *architectural threat indicators* are introduced into the model as additional attributes of the project concept. These indicators are most likely not directly measurable properties of the architecture but are related to oberservable architectural properties. In operationalizing, two types of threat indicators have to be distinguished:

Quantitative indicators, which quantify certain architectural properties, as e.g. the number of affected concepts. The level of measurement for such indicators is at least *ordinal*, i.e., they support comparisons.

Threat pattern indicators, which indicate that a specific pattern of architectural changes is performed by a project, which is likely to threaten project success. The respective level of measure is *nominal*, although it can be interpreted as ordinal, since the presence of the corresponding pattern can be regarded more negative than its absence.

By this construction, it is assured, that every indicator can be interpreted in terms of *better* and *worse*.

> The architectural threats as identified before are operationalized via two additional properties of the IT project concept – `complexity` and `standard compliance`. Different perspectives on the complexity of a project may exist. As alluded above, the number of associated interfaces contributes to the complexity of an IT project. This calls for a *topologic* complexity measure interpreting the affected part of the EA as a graph.
>
> Whereas the complexity property is classified as a quantitative indicator, the standard compliance property can be classified to be a threat pattern indicator. Thereby, each deviation from given standards either at the beginning or during the execution of the IT project can be seen as an architectural change by the project leading to potential threats.

Formulate indicators
The indicators introduced in the preceding step directly relate to architectural threats for projects, but are not necessarily observable. In contrast, multiple observable properties of the architecture exist, which are reflected in the information model. This step establishes a theory, which relates the observable architectural properties to the architectural threat indicators. This theory is used to define the indicators as aggregations of architectural properties. To incorporate this theory in a concise EA information model, *definitional dependencies* between the observable properties and the corresponding threat indicators are introduced. This procedure especially applies for the *quantitative indicators*, while the *threat pattern indicators* can be formulated as architectural rules; for this different languages may apply, ranging from a textual description of the respective pattern, over an object diagram describing the instances and links in the pattern, to a formal constraint in an appropriate language, e.g. the *Object Constraint Language (OCL)* [OMG06].

In our manufacturing example, the complexity threat was classified as quantitative indicator. In order to calculate this indicator, a meaningful approach should solely regard those constituents of the EA, which are altered by the project. The manufacturing company chooses to use an indicator derived from the *cyclomatic complexity* as introduced in [McC76]. While this complexity measure was initially developed to apply on single applications, indications for its usability on application landscape or even EA level are discussed in [Nie06]. In this respect, the business applications are interpreted as nodes and the interfaces as edges in a graph. Starting with the directly connected business applications also the first degree neighbors are taken into consideration. Counting the nodes and edges the numbers n and e respectively are obtained. Further, the number p of independent sets in this subgraph is determined. From there, a complexity measure N is computed by the calculation rule $N = e + n - p$. This rule is slightly different from the one proposed for the cyclomatic complexity to incorporate the peculiarities of architectural complexity[4].

The standard compliance threat however is formulated as architectural rule as the topic of setting architectural standards is a new but emerging endeavor in the manufacturing company. Thereby, the rule is identified as a textual description: *A business application, which is newly introduced into the EA, should be based on standard technology. Deviation from technology standards is only permitted in exceptional cases and entails a review by the architecture board.*

Prioritize and decide on indicators

In the preceding steps a set of architectural threat indicators was established. To facilitate IT project threat analyses, all indicators have to be taken into account. Nevertheless, one cannot expect these indicators to be of equal importance. Furthermore, computing the values of the different indicators is connected to measuring the values of observable architectural properties, which they are based upon. Performing these measurements does not come without costs. In particular, while especially architectural information on a higher level of abstraction might already be present in the company's EA management tool, other information has to be collected in a costly and time consuming process solely for the architecture radar system. Therefore, the amount of work related to observing a property should – among others – be taken into account, when prioritizing the indicators. Thereby, the architecture radar system has to be designed according to importance and costs of the indicators, i.e., the system can be pareto-optimized according to both criteria. As result of this step, the EA information model is augmented with information of indicator priorities and properties. Additionally, architectural concepts too costly to collect are marked as *not accounted for*. Nevertheless, the latter concepts and properties should not be deleted from the model, as they might serve as valuable input for refining the architecture radar system in a subsequent iteration of the construction process.

[4] A short discussion leading to the formula and discussing the differences to the cyclomatic complexity formula is given in Section 4.

In our simplified example from the manufacturing industry, we only considered two architectural threats. Nevertheless, a decision about the indicators in respect to a cost/benefit ration has to be made the definition and communication of standards is a time consuming task. Similar considerations hold for gathering information regarding the complexity of an IT project. At the manufacturing company, information about business applications and their interconnections is already presented in a CMDB. Therefore, both indicators are used for the construction of the architecture radar system.

Create analysis model
At the beginning of this step, the information model is reduced to the most important concepts. Similarly, the indicators have been prioritized and decided upon. What remains to be done is, firstly, to convert the *causal dependencies* between the observable properties and the indicators to quantitative computation methods. Secondly, the threat pattern indicators have to be complemented with computable analysis rules, that can be used to match the underlying architectural threat patterns in the EA description. Especially for the latter, object-oriented constraint languages as the OCL can be used, if the threat pattern can be described in terms of a navigation starting from a dedicated concept. For providing the computation methods, different techniques can be used ranging from simple equation models to more sophisticated ones, as e.g. *Bayesian belief networks* (see e.g. [BFH+09]).

While the complexity threat has already been formalized in a preceding step, the standard compliance threat, which is until now only documented utilizing a textual description, needs to be reformulated in an appropriate language to create an analysis model via an OCL constraint:

```
context:Project
derive standardCompliance =
   introduces->forAll(used.standard==true)
```

Having developed an appropriate analysis model for project surveillance, this model is subsequently enacted as part of an integrated project planning and management process. After the completion of a project, the analysis model underlying the architecture radar system can be updated to incorporate the lessons learned from the corresponding project. In particular, projects, which have not been selected for surveillance by an enterprise architect, but which failed due to architectural threats, can provide valuable input to the analysis model. Therefore, the intermediary results of the process for constructing an architecture radar system, as alluded to above, can be of use. This especially applies to the indicators, which were excluded or prioritized low during the process.

4 Exemplary architecture threats

In preceding Section 3, complexity and standard compliance were discussed as representatives of architectural threats. However, there are several other threats arising from different

EA elements. Based on these elements, one may categorize the threats. As done subsequently, we rely on a modified version of the EA structure model as presented in [Wit07] (cf. Figure 3). In terms of this structure model, one can distinguish between architectural threats originating from different layers as *business and organization*, *application and information*, and *infrastructure*, but can also discuss threats related to the cross function *blueprints and patterns*[5].

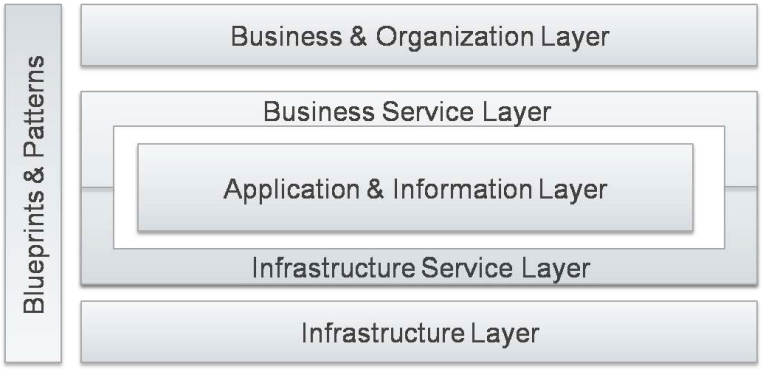

Figure 3: Layers and cross functions of an EA

Subsequently, we give a non-exhaustive list of thinkable architectural threats and exemplify the classification along the EA structure model. We thereby also consider the threats as discussed in the running example of Section 3 and elicit some difficulties in operationalizing a threat to an indicator.

Business and organization layer
For this first category, threats originating from the organization structure can be investigated. IT projects affecting a number of business applications, which are operated by different organizational units are more likely to run into interest conflicts, e.g. concerning the applications' planned ex ante architecture. In this respect, a complex organizational environment may pose an architectural threat to project completion. This threat is similarly also discussed in [Mer09], where an unclear distribution of power and entitlement among a high number of boards and committees disposing of different visions and interests, is regarded a project threat. In addition, the business processes supported by the business applications might represent a project threat. In particular, a business application supporting a number of highly important or time-critical business processes is likely to be difficult to adapt, as a mere auxiliary application.

Application and information layer
The business application's history gives rise to a potential architectural threat. With an increasing number of changes, that have already been applied to the business application, it may develop and increase its *resistance to change* [MWF08]. This might especially be

[5]The original structure model introduces additional cross functions, which are not considered here as they cannot be considered as sources for architectural threats but instead provide means to e.g. assess architectural threats (cf. cross function KPIs and metrics).

the case, as the application's internal architecture is more likely to clump together. Nevertheless, this history of changes for a business application is not that easy to analyze, as major reworks of the application must be clearly separated from sole maintenance operations, as the former, at least as a side effect, reduce resistance to change. In contrast, IT projects of the latter type only occasionally reduce the resistance to change but are very likely to complicate the business application's internal architecture. The aforementioned considerations only take EA level information into account, i.e., they use information on a business application's change history to make predictions on the quality of the internal architecture of the application. This might be a sensible method as an in-depth assessment of the architecture quality of all business applications in an enterprise may not be feasible due to their sheer number.

Complexity, as discussed above, is a core architectural threat at the application layer. Different indicators for operationalizing this threat have been sketched such as the connection degree of the business applications [Bru05], their degree of clustering [AW09], and coupling [EHH+08]. As alluded to in Section 3 also in related disciplines, complexity measures such as McCabe's cyclomatic complexity [McC76] exist. This measure was used in a slightly adapted manner in the application example, as the cyclomatic complexity targets at determining the complexity of a method, where low connectness, i.e., a high number p indicates a high complexity. In contrast, the environments of a project's affected business applications might be more easy to change, if the impact is partitioned to several disconnected portions of the overall application landscape. The graph-based complexity measure from the example in Section 3 accounts for this aspect.

Blueprints and patterns crossfunction
Drilling deeper into the architecture of a business application, the non-conformity of application components to standards (as mentioned in [MWF08]) turns out to be a hazardous threat for successful project completion (cf. running example from Section 3). In particular, additional know-how for changing these components has to be hold ready, but also capable employees implementing changes. Both might be less likely to apply for components, which use standard technologies. Similar to the resistance to change discussion from above, only EA level information is accounted for in the standard conformity assessment. More detailed information on the used technologies and adhered standards can be gathered on an application architecture level, although collecting such information is much more extensive. In particular, a similar argument as with the resistance to change here also applies: due to their sheer number, it might not be feasible to gather information on the application architecture of each business application in the enterprise.

Infrastructure layer
Beside threats from the business and application layer, the infrastructure may also threaten the completion of IT projects. Again, an increased diversification of used technology rises the variety of skills that must be available. Furthermore, the longevity of infrastructure components is most likely to impact the project execution. Older infrastructure components might be at risk of running out of support, resulting in an IT project that sets a new or greatly reworked business application on an aging or almost outdated infrastructure. In particular, such projects are in danger to conflict with maintenance or upgrade projects for the infrastructure. If this is not the case, they are nonetheless most likely to cause follow-up projects, which could have been avoided by careful planning the infrastructure.

While some of above described threats may be considered as being irrelevant in one enterprise, other organizations should take them serious, when deciding which IT projects should be provided with additional guidance by an enterprise architect. Since the initial position of each enterprise differs, it is impossible to make a general statement about those architectural threats, which should be necessarily in scope of close project surveillance.

5 Outlook

In this paper, we presented a method for designing an enterprise-specific architecture radar system for the purpose of assisted project surveillance. Based on related work proposing indicators for project management as well as for the EA, an iterative process for constructing an architecture radar system was developed and exemplified utilizing a running application example from practice. Preparing the iterative process, the paper introduces a distinction between *classical* project preparation and execution threats, as widely discussed in project management literature, and architectural threats, arising from the part of the EA, which a project affects. With its focus on the latter threats, the process for constructing an architecture radar system provides a contribution to the field of EA planning and execution. Complementing the construction process, we further identified, sketched, and categorized different architectural threats. This by far non-exhaustive list provides a good starting point for deriving additional indicators and rules for analyzing parts of the EA, as changed by a project, in respect to potential architectural threats.

As the present document is a research in progress paper, no real world case studies were provided to validate the suggested approach. Nevertheless, we are currently using the method at one of our industry partners, where first results demonstrate its suitability. However, to generalize and validate the presented method, additional case studies have to be conducted in different fields of industry. In a second step, the method has to be supplemented by a role model to support the transformation towards a more comprehensive approach.

This article discussed architectural factors threatening project success and the operationalization of the threats by rules and indicators in presenting two simplified examples – complexity and standard conformance. In this respect, additional research has to be carried out in order to identify meaningful and significant indicators and suitable rules. On the one hand, those concepts (indicators and rules) should reflect architectural threats tying in with the implementation of IT projects, on the other hand they should be suitable for the architecture radar system, hence being easy to collect, to maintain, and to interpret. Moreover, appropriate methods and languages for documenting and operationalizing rules as well as indicators are subject to future research. Thereby, methods how to augment informal documentations of the EA to support calculations of formal indicators is an area of research that will need further investigation although the works of [AS08] and [FHKS08] provide good starting points. Concerning the formalization of rules on EA level the situation is even more complicated, as no widely accepted method has been proposed yet.

References

[AS08] Jan Stefan Addicks and Ulrike Steffens. Supporting Landscape Dependent Evaluation of Enterprise Applications. In Martin Bichler, Thomas Hess, Helmut Krcmar, Ulrike Lechner, Florian Matthes, Arnold Picot, Benjamin Speitkamp, and Petra Wolf, editors, *Multikonferenz Wirtschaftsinformatik*, pages 1815 – 1825, GITO-Verlag, Berlin, Germany, 2008.

[AW09] Stephan Aier and Robert Winter. Virtual Decoupling for IT/Business Alignment Conceptual Foundations, Architecture Design and Implementation Example. *Business & Information Systems Engineering*, 1(2):150–163, 2009.

[BEMS09] Sabine Buckl, Alexander M. Ernst, Florian Matthes, and Christian Schweda. An Information Model for Managed Application Landscape Evolution. *Journal of Enterprise Architecture (JEA)*, 5(1):12 – 26, 2009.

[BFH+09] Sabine Buckl, Ulrik Franke, Oliver Holschke, Florian Matthes, Christian. M. Schweda, Teodor Sommestad, and Johan Ullberg. A Pattern-based Approach to Quantitative Enterprise Architecture Analysis. In *15th Americas Conference on Information Systems (AMCIS)*, San Francisco, USA, 2009.

[BMM98] William J. Brown, Raphael C. Malveau, and Thomas J. Mowbray. *AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis*. Wiley, New York, USA, 1998.

[Bru05] Ralph Brugger. *IT-Projekte strukturiert realisieren: Situationen analysieren, Lösungen konzipieren – Vorgehen systematisieren, Sachverhalte visualisieren – UML und EPKs nutzen*. Vieweg+Teubner, Wiesbaden, Germany, 2^{nd} edition, 2005.

[CG06] David L. Cleland and Roland Gareis. *Global Project Management Handbook: Planning, Organizing and Controlling International Projects, Second Edition*. McGraw-Hill Professional, 2^{nd} edition, 2006.

[EHH+08] Gregor Engels, Andreas Hess, Bernhard Humm, Oliver Juwig, Marc Lohmann, and Jan-Peter Richter. *Quasar Enterprise – Anwendungslandschaften serviceorientiert gestalten*. dpunkt.verlag, Heidelberg, Germany, 2008.

[FHKS08] Ulrich Frank, David Heise, Heiko Kattenstroth, and Hanno Schauer. Designing and Utilising Business Indicator Systems within Enterprise Models – Outline of a Method. In *Modellierung betrieblicher Informationssysteme (MobIS 2008) – Modellierung zwischen SOA und Compliance Management 27.-28. November 2008*, Saarbrücken, Germany, 2008.

[FK06] Quentin W. Fleming and Joel M. Koppelman. *Earned Value Project Management*. Project Management Institute, Pennsylvania, USA, 3^{rd} edition, 2006.

[Fra02] Ulrich Frank. Multi-perspective Enterprise Modeling (MEMO) – Conceptual Framework and Modeling Languages. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS 2002)*, pages 1258–1267, Washington, DC, USA, 2002.

[HV99] John Charles Henderson and N. Venkatraman. Strategic alignment: leveraging information technology for transforming organizations. *IBM Systems Journal*, 38(2-3):472–484, 1999.

[Int09] Standish Group International. The Chaos Report 2009. http://www.standishgroup.com/newsroom/chaos_2009.php (cited 2009-05-05), 2009.

[Jac98] Peter Jackson. *Introduction to Expert Systems*. Addison Wesley, Harlow, England, 3^{rd} edition, 1998.

[JJSU07] Pontus Johnson, Erik Johansson, Teodor Sommestad, and Johan Ullberg. A Tool for Enterprise Architecture Analysis. In *11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007), 15-19 October 2007, Annapolis, Maryland, USA*, pages 142–156, IEEE Computer Society, Annapolis, Maryland, USA, 2007.

[KN91] Robert S. Kaplan and David P. Norton. The Balanced Scorecard - Measures That Drive Performance. *Harvard Business Review*, 70(1):pages 71–79, 1991.

[Küt06] Martin Kütz. *IT-Steuerung mit Kennzahlensystemen*. dpunkt.verlag, Heidelberg, Germany, 2006.

[Lan08] Josef Lankes. *Metrics for Application Landscapes – Status Quo, Development, and a Case Study*. PhD thesis, Technische Universität München, Fakultät für Informatik, Munich, Germany, 2008.

[Luf03] Jerry N. Luftman. *Competing in the Information Age – Align in the Sand*. Oxford University Press, New York, USA, 2^{nd} edition, 2003.

[McC76] Thomas J. McCabe. A Complexity Measure. *IEEE Transactions on Software Engineering*, 2(4):308–320, 1976.

[Mer09] Peter Mertens. Schwierigkeiten mit IT-Projekten der öffentlichen Verwaltung. *Informatik Spektrum*, 1(32):42 – 49, 2009.

[MP07] Peter Morris and Jeffrey K. Pinto. *The Wiley Guide to Project, Program, and Portfolio Management (The Wiley Guides to the Management of Projects)*. Wiley, Hoboken, USA, 2007.

[MWF08] Stephan Murer, Carl Worms, and Frank J. Furrer. Managed Evolution. *Informatik Spektrum*, 31(6):527–536, 2008.

[Nie06] Klaus D. Niemann. *From Enterprise Architecture to IT Governance – Elements of Effective IT Management*. Vieweg+Teubner, Wiesbaden, Germany, 2006.

[OMG06] OMG. Object Constraint Language (OCL) Available Specification, version 2.0 (formal/06-05-01), 2006.

[Sch08] Jaap Schekkerman. *Enterprise Architecture Good Practices Guide – How to Manage the Enterprise Architecture Practice*. Trafford Publishing, Victoria, BC, Canada, 2008.

[The09] The Open Group. TOGAF "Enterprise Edition" Version 9. `http://www.togaf.org` (cited 2009-07-10), 2009.

[WBFK07] Robert Winter, Tobias Bucher, Ronny Fischer, and Stephan Kurpjuweit. Analysis and Application Scenarios of Enterprise Architecture – An Exploratory Study. *Journal of Enterprise Architecture*, 3(3):33–43, 2007.

[Wit07] André Wittenburg. *Softwarekartographie: Modelle und Methoden zur systematischen Visualisierung von Anwendungslandschaften*. PhD thesis, Fakultät für Informatik, Technische Universität München, Germany, 2007.

[WM07] Hans W. Wieczorrek and Peter Mertens. *Management von IT-Projekten. Von der Planung zur Realisierung*. Springer, Berlin, Germany, 2^{nd} edition, 2007.