Design Solution Analysis for the Construction of Situational Design Methods

Robert Winter

To cite this version:

Robert Winter. Design Solution Analysis for the Construction of Situational Design Methods. 4th Working Conference on Method Engineering (ME), Apr 2011, Lisbon, Portugal. pp.19-33, 10.1007/978-3-642-19997-4_4 . hal-01562878

HAL Id: hal-01562878
https://hal.inria.fr/hal-01562878

Submitted on 17 Jul 2017

Design Solution Analysis for the Construction of Situational Design Methods

Robert Winter

Institute of Information Management, University of St. Gallen
Müller-Friedberg-Strasse 8, 9000 St. Gallen, Switzerland
Robert.Winter@unisg.ch

Abstract. Situational design methods provide problem solving guidance that can be configured to fit a range of different design goals and contexts. While the formal aspects of situational method engineering are well researched, the specification of method fragment instances and their configurations is often left open and regarded as specific to the respective design problem class. We propose an approach that analyzes variations of existing design solutions to explore the underlying design factors and to identify design situations. This knowledge is then used to derive method fragments and configuration rules that represent the explored variety of design solutions. The proposed approach has been applied for several design problem classes to construct concrete situational design methods. As an illustrative example, the construction of a situational design method for enterprise architecture management is used in this paper. Based on the exploration of eight design factors and three design situations, six method fragments are derived that are combined into four situational method configurations.

1 Introduction: Contingencies, Design and Situational Methods

"At the most abstract level, the contingency approach says that the effect of one variable on another depends upon some third variable" [1]. In an organisational context, contingency theory argues that success or failure of different organizational structures depends on contingency factors such as size, task uncertainty, and task interdependence [2]. Although its validity is questioned by some [e.g. 3], we agree with Donaldson that "overall, empirical studies show that fit positively affects performance, thereby supporting the central idea of contingency theory" [1].

Design Science Research is a research paradigm that has been, among other application domains, successfully deployed to Information Systems (IS). In the following, we will use DSR to abbreviate Design Science Research in Information Systems. At its core, DSR is about the rigorous construction of ***useful*** IS artefacts, i.e. "technology-based solutions to important and relevant business problems." [4, table 1] Technically, IS artefacts can be constructs, models, methods, or instantiations [5]. While instantiations are usually represent a solution to a singular design problem, constructs, models and methods can have different levels of ***generality*** and, as a consequence, "represent [.] a general solution to a class of [design] problems." [6]

Generality is therefore an important quality of an IS artefact [4]. The two design goals of generality and utility are however conflicting. In their research on reference modelling, Becker et al. discuss what they call the [process] ***reference modelling dilemma***: "On the one hand, customers will choose a [process] reference model that […] provides the best fit to their individual requirements and therefore implies the least need for changes. On the other hand, a restriction of the generality of the model results in higher turn-over risks because of smaller sales markets" [7]. This dilemma is not only apparent in [process] reference modelling, but also exists for design methods. With increasing generality, the individual utility of a solution for solving a specific design problem decreases – and vice versa. The overall, cross-organization sum of individual utilities might be increasing when design solutions have a higher generality – but individual organizations might not be interested in this "overall utility". As a solution to this dilemma, Becker et al. [7] propose adaptation mechanisms that instantiate a generic [process] reference model according to the specific design problem at hand.

Both design methods and [process] reference models can be understood as (more or less) general problem solutions. As a consequence, situational methods (e.g. [8, 9, 10, 11]) can, like adaptable [process] reference models, aim at solving the trade-off between solution generality and individual solution utility.

As situational method engineering allows to develop artefacts which are adaptable to different design problem instances within a design problem class, a crucial decision during the method construction phase is to delineate the range of addressed design problems (i.e. to specify the design problem class) and to understand the relevant ***design situations*** within this class. If a design problem class is understood as a set of "similar" design problems, a design situation can be understood as a subset of design problems which are even more similar, i.e. which share certain contingencies. It has been argued that, in situational method engineering, such contingencies can be represented by a certain design goal vector and/or by a certain context [12]. Depending on the degree of desired generality, a design problem class can be partitioned into few, very generic design situations or a larger number of design situations of lesser generality.

The configuration or adaptation of a situational method to a certain design situation can therefore be understood as an application of contingency theory. If relevant contingencies of the respective design problem class are represented correctly by appropriate design situations and appropriate *adaptation/configuration mechanisms*, design solutions can be generated that not only solve the [process] reference modelling dilemma, but also consider contingency factors. To achieve this, however, method engineering must identify the contingencies of a design problem class and correctly derive not only a set of suitable design situations, but also adaptation/configuration mechanisms that combine method fragments into situational methods.

The aim of this paper is to show that method construction and contingency identification can be based on an analysis of a sufficient number of existing design solutions for the addressed design problem class. In section 2 we summarize and extend an existing design solution analysis approach. Section 3 outlines how design solution analysis is used to derive method fragments and fragment configuration rules. As an illustrative example, enterprise architecture management is used as the design problem class, and a respective situational method is derived in section 4. Section 5 summarizes the paper and outlines future research in this domain.

2 Analysis of Existing Design Solutions

Winter [13] extended Bucher and Klesse's design problem analysis procedure [14] by differentiating more components and assuming that, in general, only adaptable, situational solution artefacts are constructed. In the following, Winter's proposal is refined and illustrated:

1. A ***rough idea*** about the delineation of the design problem class is developed. Results of this step are definitions, a description of the system under analysis and ideas about design goals for the respective class of design problems.
   In section 3 we illustrate our approach for the design problem class Enterprise Architecture Management (EAM). It is delineated by defining architecture, enterprise architecture and EAM, by defining the scope of relevant artefacts, and by differentiating potential EAM goals.
2. A ***literature analysis*** is conducted in order to identify potential contingency factors for the respective class of design problems, i.e. factors which might have influence on how such design problems are solved in practice.
   For EAM, such an analysis yields factors like 'EAM's main sponsor is IT or business', 'EAM's main deliverable is maps, analyses or project support', 'EAM's main goal is transparency, consistency, simplification, or flexibility', or 'EAM's role is active or passive'.
3. A ***field study*** is conducted in order to analyze how design solutions for this class of design problems in practice are actually related to what contingencies. Using ***principal component analysis*** on the field study data, the list of potential contingency factor candidates from step 2 is reduced to a smaller set of relevant "design factors". Design factors are usually aggregates of several relevant contingency factors and therefore need to be semantically interpreted.
   For EAM, principle component analysis on EAM practice solutions yielded eight design factors (like IT operations support, integrative role, business strategy support, or design impact) which aggregate 54 statistically relevant contingencies (see section 4). E.g., the design factor 'integrative role of EAM' aggregates the contingencies 'EAM takes place in an interdisciplinary team', 'EAM team and business departments continuously exchange information (e.g. in architecture boards)' and 'EAM team and IT departments continuously exchange information (e.g. in architecture boards)'.
4. In a multi-dimensional room where every dimension corresponds to a design factor, every observed solution can be understood as a point. The design problem class now should be ***redefined*** by specifying value ranges for the design factors identified in step 3. This means that "outlier" design solutions are excluded from

further analysis in order to ensure a useful degree of homogeneity of solutions.
For EAM step 4 leads to the exclusion of few observed solutions which can be clearly recognized as outliers, i.e. which clearly cannot be associated with any cluster in the above described multi-dimensional room.

5. For the vast majority of observations, ***ultrametric distances*** can now be computed that represent the similarity (or dissimilarity) between design solutions. Metrics for ultrametric distances are usually based on Euclidian distance. The observations and their distances can be visualized using a dendrogram-like tree graph. The (dis)similarity of two design solutions corresponds to the generality level of their link. If two design solutions are very similar, their link is represented on a low level of generality. If two design solutions are very different, their link is represented on a very high level of generality. The linkage can be interpreted as generalization of the linked specific solutions.
Figure 1 [adapted from 13] illustrates such a tree graph that represents 33 observed solutions (C1...C33, on the bottom of the tree diagram) in design problem class C as well as their ultrametric distances. The generalization of solutions C11, C12, C13, C14 and C15 is represented as generic solution C11...C15 on some level of generality. The generalization of solutions C1 through C15 is represented as even more generic solution C1...C15 on a higher level of generality. The maximum generic solution of design problem class C is found at the top of the tree diagram.

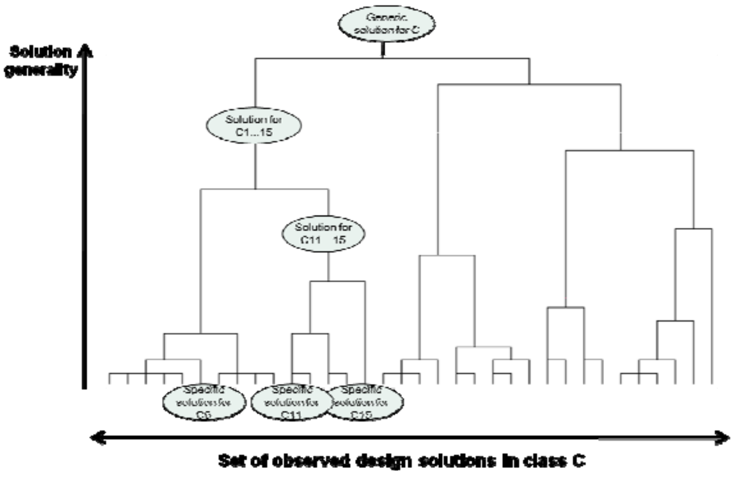
Generic solution for C
Solution generality
Solution for C1...15
Solution for C11...15
Specific solutions for C6
Specific solutions for C11
Specific solutions for C15
Set of observed design solutions in class C

Fig. 1 Ultrametric tree visualization of observed design solutions for design problem class C

6. In order to not only visualize, but characterize generic solutions in C, a ***clustering algorithm*** can be applied to the observation data. By agglomerative clustering, solutions can be specified at any generality level between “full detail” (i.e. one cluster per original observation) and “one size fits all” (i.e. one generic solution description for the entire design problem class). Analyzing the clustering error in relation to the number of clusters, an optimal level of generality (i.e. an optimal

number of clusters) can be determined.
Empirical data can be used to determine the optimal number of clusters, i.e. the number of different 'approaches' that should be differentiated, e.g. for a certain number of companies that implemented EAM. If the set of observations is large and diverse enough, this finding might be applied to EAM in general.

7. For the level of solution description generality chosen in step 6, each cluster represents one ***design situation***. The situations should not only be defined formally (i.e. by specifying value ranges of the respective design factors), but also should be interpreted semantically ("design problem types").
EAM clusters differ in particular with regard to their values for the design factors IT operations support, integrative role, design impact, enterprise-wide focus and IT strategy support. These differences are used to characterize one cluster as 'balanced, active EAM', one as 'business analysis' and one as 'IT focused, passive EAM'. As a consequence, situational method engineering should provide three situated methods. Since the clusters are similar with regard to some (but not all!) design factors, these situated methods will share certain method fragments.

There is some, but not much, related work on how to identify design situations for situational method engineering [15]: Some suggest to differentiate "project size", "number of stakeholder groups" or "applied technology" for every situational method (e.g. [16] and [17] according to [18]), while others specify situations on a case-by-case basis [e.g. 10, 12]. The procedure proposed here allows systematic and reliable identification and specification of design situations for any given class of design problems as long as a sufficient number of problem solutions can be observed and analyzed for this problem class.

But identifying design factors and design situations is only the first part of constructing a situational design method. Design problems need to be identified and linked to design situations, and a set of method fragments needs to be specified whose combinations will constitute useful solutions for such design problems. The next section proposes a procedure for this second part of situational design method construction.

It should however be noted that even both construction procedure parts are not necessarily sufficient to solve every design problem in C. Situated methods might need to be adapted to provide a useful design solution to a specific design problem. Referring to fig. 1, a combination of method fragments might solve the generic design problem for situation C1..15 sufficiently, but still might need to be adapted to design problem C15 in order to solve this specific problem instance effectively.

3 Derivation of Method Fragments and Configuration Rules

For typical design problem classes, between four and eight design factors can be identified which explain the variance of the observed design solutions sufficiently [19, 20, 21, 22, 23, 24]. These design factors span up a solution room where between three and six design situations are differentiated.

The crucial step is to qualitatively interpret the n design factors and m design situations that have been quantitatively created as principal components of the data set and clusters in the *n*-dimensional design factor space, respectively. For that purpose, it is necessary for every design situation to identify the subset of *p* design factors ($p \leq n$) that best characterizes the respective design situation, i.e. whose factor values are particularly high or low in a cluster and/or whose factor values have only a small standard deviation in a cluster.

Understanding the problem-oriented relations between design factors and design situations is essential for the construction of respective methods: Each method fragment can then be interpreted as an "***elementary movement***" in the *p*-dimensional design factor sub-space. The situated method aggregates certain fragments and therefore constitutes a complex, multi-dimensional movement in the *n*-dimensional design factor space.

8. For every design situation ***characterizing design factors*** need to be identified. In EAM, only the design situation 'IT focused, passive EAM' is characterized by high values of the design factor IT operations support and low values of the design factors enterprise-wide focus, integrative role and design impact. The EAM design situation 'balanced, active EAM', in contrast, exhibits much smaller values for IT operations support, but much higher values for enterprise-wide focus, integrative role and design impact. With regard to information supply, business support and IT strategy and IT governance support, these two design situations are not very different, so that these factors are not useful to characterize them.
9. Now characterizing design factors need to be linked to ***design problems***. All preceding procedure steps analyze existing design solutions. Since these design solutions were created purposefully, they are qualitatively interpreted and linked to design problems. For 'IT focused, passive EAM', the characterizing design factors 'integrative role', 'enterprise-wide focus' and 'design impact' can be associated with an EAM setup where the main EAM sponsor is the CIO, the main EAM customer is the IT function, EAM is primarily performed within the IT function and EAM is widely ignored by business units. 'Business analysis', in contrast, can be associated with an EAM setup where business is the main stakeholder and executor and where implementation considerations are widely neglected. Most EAM setups can be easily linked to major EAM challenges as often described in the literature. E.g., missing business involvement and missing business value creation of EAM correspond to the first EAM setup, while missing 'grounding'/'execution' and too much 'locality' of EAM correspond to the latter EAM setup.
10. Elementary problem-solving actions are now derived by comparing design solutions (steps 1-8) with design problems (step 9). These elementary problem-solving actions constitute ***method fragments***. If e.g. the design problem is that business stakeholders are not sufficiently involved in EAM sponsorship and/or EAM delivery, and that EAM recommendations seem not to create sufficient business value, the as-is EAM setup is close to 'IT focused, passive EAM' while the to-be EAM setup is likely to be 'Balanced, active EAM'. Elementary problem-solving activities can be derived from the respective characterizing design factors 'integrative role', 'enterprise-wide focus' and 'design impact'. A suitable method fragment should include, among others, 'EAM alignment with business goals', 'architects

have an extensive network within the company', 'EAM team and business departments continuously exchange information (e.g. in architecture boards)', 'EAM takes place in an interdisciplinary team' and 'EAM has an impact on business architecture design'. If, as another example, the design problem is that EAM is not sufficiently 'implemented' and creates not enough impact, the as-is EAM setup is close to 'business analysis' while the to-be EAM setup is likely also to be 'Balanced, active EAM'. Elementary problem-solving activities can be derived from respective characterizing design factors 'IT governance and IT strategy support', 'IT operations support' and 'EAM governance'. Hence a suitable method fragment should include, among others, 'Results of EAM are used for IT strategy development', 'Architecture data is centralized with the EAM department' and 'Results of EAM are used for IT development'.

11. Based on the set of identified design problems and specified method fragments, now ***method configuration rules*** need to be derived. Basically the (reusable) method fragments identified in step 10 need to be related to respective design situations. The fewer characterizing design factors and the fewer design problems have been identified, the simpler the fragment configuration will be – and vice versa. For the EAM example, four situated methods are configured from, depending on the design situation, up to four method fragments out of a total number of six reusable method fragments (see section 4).

4 Enterprise Architecture Management - An Illustrative Example for Design Solution Analysis

The ANSI/IEEE Standard 1471-2000 defines architecture as "the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution" [25]. For enterprise architecture, relevant system views are strategic positioning, organizational structure, process organization, information flows, and implementation by means of software systems and data structures [26]. EAM can provide systematic support to organizational change that affects business structures as well as IT structures by providing constructional principles for designing the enterprise [27]. The development and evolution of EAM need to be based on appropriate design methods. EAM methods typically comprise strategic design of an architectural vision, development and maintenance of as-is architecture models, development and maintenance of to-be architecture models, migration planning, implementation of enterprise architecture, and analysis of enterprise architecture on the basis of architecture models [28].

Aiming at a deeper understanding of the constituent factors that influence EAM, there has been some scientific effort to analyze contingency factors of EAM. Aier et al. [29] have identified models, data, and organizational penetration as potential contingencies. They did however not explicitly consider management aspects of EAM. Leppänen et al. [30] made a first step towards a complex contingency framework for an engineering method for enterprise architecture. Ylimäki [31] conducted several studies in order to identify potential critical success factors for EAM, yielding commitment, governance, methodology, enterprise architecture models, project management, training and education, organizational culture, IT investment strategy, assessment and evaluation, business-driven approach, communication, and scope. These success factors give a first insight into possible design factors of EAM.

4.1 Procedure Steps 1 through 7: Design Solution Analysis

This subsection summarizes Aier et al.'s empirical analysis of EAM design solutions [32] that applied steps 1 through 7 of the proposed procedure – although not elaborating the procedure itself. Aier et al. use Ylimäki's set of EAM success factors as a starting point. In order to distinguish different EAM approaches, the first part of their questionnaire asks for a company's general EAM understanding. Then, the EAM positioning is analyzed using questions on EAM integration in the organization and on the way how organizational units, teams and roles are involved in EAM processes. Other important aspects in this context are the scope of EAM processes, the penetration of EAM processes / EAM results throughout the organization as well as the level of continuity and controlling of EAM processes. In the third and final part of the questionnaire, it is asked what types of EAM results are used by which organizational units. All in all, 54 questions are used to assess current EAM design solutions in companies. At four EAM practitioner events, a total of 119 data sets were collected that did not reveal substantial extent of missing data (10% at maximum).

In order to identify design factors for EAM, Aier et al. apply an exploratory factor analysis. The original study [32] documents two quantitative techniques that support the suitability of the data set for Principal Component Analysis. Using Varimax rotation with Kaiser normalization, eight design factors are identified that comprise a total of 38 questionnaire items. 16 questionnaire items were deleted because they were intentionally designed as control items or did not seem to contribute to the factor identification [33]. Due to some incomplete questionnaires, missing values were excluded pair wise during the factor analysis. This resulted in a total number of 109 cases contributing to the factor analysis. The items selected for the factor analysis explain 67.63% of the variance in total. The original study [32] also documents a quantitative technique that supports the validity of the factor analysis.

With regards to the interpretation of the factors, factor loadings from 0.3 to 0.4 are considered to be the minimal level [33], while factor loadings from at least 0.5 are considered as sufficient for an unambiguous assignment to one factor. For some items that showed identical factor loadings for more than one factor, the factor was chosen that best matched the respective factor from an EAM literature perspective.

The study yields the following eight EAM design factors:

a) ***IT operations support:*** The use of results for IT operation tasks and by IT departments for their daily job characterizes this factor. Considering the items' loadings on this factor it becomes obvious that usage of EAM results as well as the perception of EAM within the organizational units concerned with IT operations exert a conjoint effect on the overall assessment of EAM.
b) **S*upport of management tasks by EAM*:** This is again expressed by the usage of

EAM results by management tasks as well as by the perception of EAM in the management board. This factor constitutes the "antipole" to factor (a) and reveals that EAM can serve both IT and management purposes, but that these purposes are most probably not highly interrelated. It can be assumed that a high degree of realization for factors (a) and (b) might distinguish different EAM approaches fundamentally.

c) ***Governance of EAM***: EAM governance consists of model and process assessment and maintenance and a central supervision of EA models and data.
d) ***Support of IT strategy and IT governance tasks:*** EAM and its results are considered to be an essential part of IT strategy development and IT governance.
e) ***Information supply*****:** This design factor reflects the service function that EAM can fulfil both for business and IT departments. Moreover the support of business/IT alignment is an essential part of this factor.
f) ***Integrative role*****:** The integrative role of EAM can be realized by interdisciplinary teams and a continuous exchange between EAM roles. It can be assumed that the existence of an architecture board is part of such an organizational structure for EAM.
g) ***Design impact*****:** EAM can impact IT or infrastructure, application or business architecture. The degree of design impact reflects the penetration of the EAM approach throughout the organization as well as its active role.
h) ***Business strategy support:*** In contrast to design factor (b), items in this design factor describe the support of strategic tasks that are not management tasks - like e.g. enterprise development and product planning. Most probably, high degrees of realization of this factor correspond to a high realization of design factor (b).

Three different groups of EAM design factors were found: Design factors (a), (b), (d), (e) and (h) characterize the ***concern*** of the EAM approach (i.e. if the approach supports IT operations, management tasks, IT strategy, Business/IT alignment or business strategy). Design factors (f) and (g) describe the ***role*** of the EAM approach within the company (moderator or innovator). Finally, design factor (c) describes the ***governance*** of the EAM approach itself.

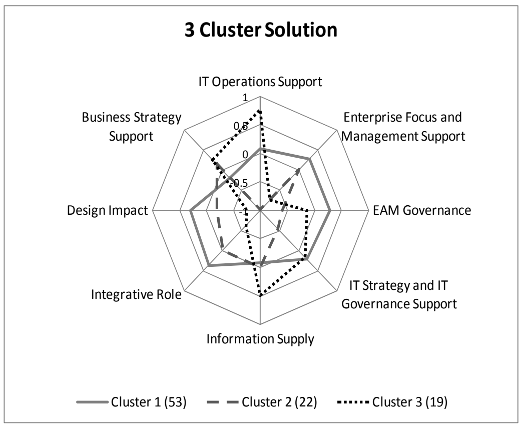

Fig. 2. EAM design situations (and their representation in the data set) [32]

Considering results from preliminary cluster analyses on the data, one case was eliminated as it showed significant outlier behaviour [33]. Excluding cases with missing factor loadings, 94 cases were used for the cluster analysis. Aier et al [32] applied the *Average Within-Group Linkage* cluster algorithm provided by SPSS and *Squared Euclidean Distance* as the distance measure. They identified three EAM design solution clusters that can be regarded as EAM design situations.

In fig. 2, the three identified EAM design situations are illustrated as a cobweb diagram. The eight EAM design factors are used as dimensions; Each cluster's centroid value was used to represent the respective cluster's values. The cobweb diagram nicely visualizes the characteristics of each EAM design situations.
The EAM design situations can be described as follows: [32]

- ***Design situation 1: Balanced, active EAM:*** The first cluster (solid line in fig. 2) presents a rather balanced approach to EAM. For most factors this cluster shows the highest or at least average values. Especially the similar values for the factors IT operations support and enterprise-wide focus lead to the conclusion that organizations within this cluster focus neither on IT support nor on management support.
  In contrast to the other clusters, the high support of IT operations, management, IT strategy as well as the focus on design impact, the integrative role and EAM governance argue for a high degree of integration within the organization. In particular the values for design impact, integrative role and EAM governance are by far the highest between all three clusters. It can therefore be presumed that these organizations have a rather high level of maturity in their EAM approach.
  It should be noted that this cluster includes 53 out of 94 organizations, which lead to the supposition that this cluster represents a "mainstream" approach.
- ***Design situation 2: Business analysis:*** The second cluster (dashed line in fig. 2) groups 22 organizations that have an apparent focus on business support in their EAM approach. The factors IT operations support as well as IT strategy and IT governance support are clearly assigned with comparatively low values. Comparing the mean factor values to those of cluster 1, the overall low values imply that the organizations in this cluster do not show a high degree of EAM implementation in any dimension. Two conclusions can be derived from this fact: Firstly, the organizations could have decided to apply a minimalist EAM approach, focusing on management support without putting resources in EAM governance or an active role of EAM. Second, the introduction of EAM could only recently be initiated by management and is not very mature yet. For both cases, literature suggests that a sustainable EAM approach can only be established by realizing an effective EAM governance [34, 35].
- ***Design situation 3: IT focused, passive approach:*** Organizations assigned to this cluster (dotted line in fig. 2) clearly emphasize the use of EAM for IT operations as well as the information supply by EAM. In contrast, values for management support are by far the lowest compared to the other clusters. As the factors design impact as well as integrative role are not focused by this approach, it can be described as a passive approach that is most probably realized very locally within the organization.
  Obviously, this small cluster, which includes only 19 of the 94 organizations,

represents a specialized IT-centred EAM approach that primarily takes a documentation role. It can be presumed that the EAM approach was initiated by IT departments and has not been disseminated throughout the organization yet.

After delineating the design problem class, collecting empirical design solution data, identifying and interpreting design factors, and identifying and interpreting design solution clusters, the 'analysis' portion of the proposed procedure is completed. The new steps 8 through 11 address the construction of a situational design method on that basis.

4.2 Procedure Steps 8 through 11: Design Method Construction

The description of design situation 1 shows that this situation rather constitutes a mature, to-be state rather than a design problem. Compared to situation 1, situations 2 and 3 exhibit clear gaps and therefore can be considered as EAM design problem sub-classes. Companies that have not systematically realized EAM at all will however not be included in any of the clusters so that, in addition to the above two sub-classes, an EAM design method needs to address a third problem sub-class. If we assume that no direct transformation from "no systematic EAM at all" to the quite mature state in situation 1 is feasible, two alternatives of this third sub-class have to be regarded: from nothing to situation 2 vs. from nothing to situation 3.

In the following, we characterize each design problem sub-class (not design situation!) by assigning characterizing design factors, and we derive design method fragments from that assignment:

- ***Design problem sub-class I (from situation 2 to situation 1)***: IT operations support, IT strategy and IT governance support as well as EAM governance need to be strengthened. Since IT topics and EAM governance constitute widely different measures, two method fragments (designated as A and B) should be differentiated to achieve that transformation.
- ***Design problem sub-class II (from situation 3 to situation 1)***: Design impact, integrative role and enterprise-wide focus need to be strengthened. Since design impact and IT/business alignment issues constitute widely different measures, two additional method fragments (designated as C and D) should be differentiated to achieve that transformation.
- ***Design problem sub-class III (from nothing to situation 3)***: If an IT focused approach is favoured, IT operations support, IT strategy and IT governance support, information supply and business strategy support need to be developed foremost. In addition to fragment A (see above: IT operations, IT strategy and IT governance support), two additional fragments should be defined: fragment E to address business strategy support and fragment F to address information supply.
- ***Design problem sub-class IV (from nothing to situation 2)***: If a business analysis approach is favoured, enterprise-wide focus as well as information supply and business strategy support need to be developed foremost. Such a transformation is represented by the fragments D, E and F.

Based on EAM design factors (a)...(h) and EAM design situations (1)...(4), EAM design problem sub-classes (I)...(IV) have been derived that can be addressed by different configurations of EAM method fragments (A)...(F). It should be noted that (A)...(F), although designated as fragments here due to their configuration into situational methods, are quite complex and would be designated as method components or even methods in a different context (e.g. D as a method for EAM-based IT/business alignment). As a result from applying steps 8 through 11 of the proposed procedure, the following situational EAM method is constructed:

- ***Method for problem sub-class I (from business analysis to balanced, active EAM)***: combine method fragments A and B
- ***Method for problem sub-class II (from IT focused, passive to balanced, active EAM)***: combine method fragments C and D
- ***Method for problem sub-class III (initial development of IT focused, passive EAM)***: combine method fragments A, E and F
- ***Method for problem sub-class IV (initial development of business analysis)***: combine method fragments D, E and F

Although not advised because a big maturity leap is necessary, it is possible to combine method fragments A, D, E and F into a method for initial development of a balanced, active EAM.

5 Conclusions and Outlook

While many method engineering approaches claim to incorporate situational factors, they do nearly never detail what these situational factors exactly are and how they can incorporated into method fragment design and fragment configuration rules. This paper is based on Winter's analysis procedure [13] that has been applied to EAM design solution analysis in [32]. Since earlier extensions of the proposed analysis procedure for constructing situational methods suffer from too simplistic design situations and design problems [cf. 36], we have used here the EAM data that allows a more realistic illustration of the proposed procedure extensions. The proposed procedure guides not only the identification of relevant context and project type factors, examines their occurrences in practice, and classifies them into design solution situations. Based on an evaluation of solution maturity, design problems can be specified and associated with matching design factors. From that association, transformation fragments and their configuration into situational methods (one for each design problem sub-class) can be derived.

The question arises how general the proposed situational method engineering procedure is. While its construction portion has only been applied to IT/business alignment [36] and EAM so far, its analysis portion has been applied in many cases:

- Leist [23] uses it to identify eight enterprise modelling situations. Based on that analysis, she investigates which meta modelling approaches are best suited in which situation.

- Baumöl [19] uses it to identify four types of transformation projects. Based on that analysis, she constructs project type specific recommendations which general and type specific transformation management instruments should be used.
- Bucher and Winter [20] use it to identify four Business Process Management (BPM) realization situations and five types of BPM transformations. Based on that design problem analysis, they construct a situational BPM method.
- Klesse and Winter [21] use it to identify four organizational designs for data warehouse service providers. Based on that analysis, they give recommendations for consistent data warehousing service provisioning and identify dynamic patterns (maturity).
- Lahrmann and Stroh [22] use it to identify three approaches to organize and implement information logistics in companies. Based on that analysis, they derive guidelines and reference models for information logistics strategy design.
- Aier, Gleichauf, Riege and Saat [24] use it to verify a hypothesis that all integration projects in companies can be assigned to one of only four fundamental types.

Although the generality of the proposal needs to be demonstrated yet, there is some evidence that the procedure in general can be applied to a wide variety of IS related design problems in organizations.

Four broad categories of research opportunities exist: Firstly, the demonstration example lacks tradeoffs between design goals and design activities: Since implementation impact and business involvement should be achieved equally, the derivation of fragments and their aggregation into situated methods therefore was straightforward. If tradeoffs have to be observed, both the fragment specification and the fragment aggregation become much more complex. Secondly, an interesting feature of many design solution analyses that yield a larger number of design factors is that the first factor is often representing many and quite diverse problem aspects that are sometimes not easy to interpret qualitatively. With regard to design solution analysis and method construction, we interpret this “technically” overloaded design factor as a problem independent aggregation of “generalized” properties and the respective solution fragment as a basic set of domain-independent problem solution activities like e.g. general project/transformation management. This aspect of our approach does certainly need additional research attention. Thirdly, the proposed approach does not explicitly cover yet the adaptation of situated methods to specific design problems. On the one hand, we consider this extension not too problematic because there is a plethora of adaptation knowledge on reference models which promises to be generalizable. On the other hand, adaptation efforts might depend on problem properties and influence the “optimal” level of method generality that up to now is determined using “technical” homogeneity/heterogeneity metrics only. Finally, another and probably the most important extension of the proposed approach would be the inclusion of not only adaptation effort, but also other “economical” properties like the absolute number of design problems in a class or even their attractiveness in terms of economic gains. This is probably the most interesting – and challenging – avenue for further research.

6 References

1. Donaldson, L.: The Contingency Theory of Organizations, Thousand Oaks: Sage (2001)
2. Graubner, M.: Task, firm size, and organizational structure in management consulting. An empirical analysis from a contingency perspective, Wiesbaden: DUV (2006)
3. Pfeffer, J.: New Directions for Organization Theory: Problems and Prospects, New York: Oxford University Press (1997)
4. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design Science in Information Systems Research. MIS Quarterly, vol. 28, no. 1, pp. 75-105 (2004)
5. March, S.T., Smith, G.F.: Design and Natural Science Research on Information Technology. Decision Support Systems, vol. 15, no. 4, pp. 251-266 (1995)
6. Baskerville, R.L., Pries-Heje, J., Venable, J.: Soft design science methodology. In Proceedings of DESRIST 2009, ACM (2009)
7. Becker, J., Delfmann, P., Knackstedt, R.: Adaptive Reference Modeling: Integrating Configurative and Generic Adaptation Techniques for Information Models in Jörg Becker and Patrick Delfmann, ed., *Reference Modeling*, Heidelberg: Physica, 2007, pp. 27-58.
8. Brinkkemper, S., Saeki, M., Harmsen, A.F.: Meta-Modelling Based Assembly Techniques for Situational Method Engineering. Information Systems, vol. 24, no. 3, pp. 209-228 (1999)
9. Harmsen, A.F., Brinkkemper, S., Oei, H.: Situational Method Engineering for Information System Project Approaches, in Proceedings of the IFIP 8.1 Working Conference on Methods and Associated Tools for the Information Systems Life Cycle. 1994, North-Holland: Maastricht. pp. 169-194.
10. Mirbel, I., Ralyté, J.: Situational method engineering: combining assembly-based and roadmap-driven approaches. Requirements Engineering, vol. 11, no. 1, pp. 58-78 (2006)
11. Ralyté, J., Rolland, C.: An Approach for Method Reengineering, in 20th International Conference on Conceptual Modeling. 2001, Springer: Berlin. pp. 471-484.
12. Bucher, T., Klesse, M., Kurpjuweit, S., Winter, R.: Situational Method Engineering – On the Differentiation of "Context" and "Project Type", in IFIP WG8.1 Working Conference on Situational Method Engineering – Fundamentals and Experiences (ME07). 2007, Springer: Berlin. pp. 33-48.
13. Winter, R.: Problem Analysis for Situational Artefact Construction in Information Systems *(forthcoming)*, 2011.
14. Bucher, T., Klesse, M.: Contextual Method Engineering, Research Report BE HSG/EIW/03, University of St. Gallen, Institute of Information Management (2006)
15. Winter, R., Gericke, A., Bucher, T.: Method versus Model – Two Sides of the Same Coin?, in Advances in Enterprise Engineering II, Jan Dietz and Antonia Albani. 2009, Amsterdam.
16. Kornyshova, E., Deneckère, R., Salinesi, C.: Method Chunks Selection by Multicriteria Techniques: an Extension of the Assembly-based Approach, in IFIP WG8.1 Working Conference on Situational Method Engineering – Fundamentals and Experiences (ME07). 2007, Springer: Berlin. pp. 64-78.
17. van Slooten, K., Hodes, B.: Characterizing IS Development Projects, in Proceedings of the IFIP TC8, WG8.1/8.2 Working Conference on Method Engineering. 1996, Chapman & Hall: Atlanta. pp. 29-44.
18. Rolland, C.: A Primer for Method Engineering, in Proceedings of the Informatique des Organisations d'Information et de Décision (INFORSID). 1997, Toulouse.
19. Baumöl, U.: Strategic Agility through Situational Method Construction. In Proceedings of the European Academy of Management Annual Conference (2005)
20. Bucher, T., Winter, R.: Taxonomy of Business Process Management Approaches: An Empirical Foundation for the Engineering of Situational Methods to Support BPM. In Jan vom

Brocke and Michael Rosemann, ed., *Handbook on Business Process Management*, Springer, 2010.

21. Klesse, M., Winter, R.: Organizational Forms of Data Warehousing: An Explorative Analysis. In Proceedings of Proceedings of the 40th Hawaii International Conference on System Sciences (HICSS-40), Los Alamitos, IEEE Computer Society (2007)
22. Lahrmann, G., Stroh, F.: Towards a Classification of Information Logistics Scenarios - An Exploratory Analysis. In Proceedings of Proceedings of the 42nd Hawaii International Conference on System Sciences (HICSS-42), Los Alamitos, IEEE Computer Society (2009)
23. Leist, S.: Methoden zur Unternehmensmodellierung - Vergleich, Anwendungen und Diskussionen der Integrationspotenziale, Habilitation, Institut für Wirtschaftsinformatik Universität St. Gallen, 2004.
24. Aier, S., Gleichauf, B., Riege, C., Saat, J.: Empirische Validierung von Integrationstypen am Beispiel unternehmensübergreifender Integration. In Proceedings der 9. Internationalen Tagung Wirtschaftsinformatik, Band 1, Wien, Österreichische Computer Gesellschaft, pp. 99–108 (2009)
25. IEEE: IEEE Recommended Practice for Architectural Description of Software Intensive Systems (IEEE Std 1471-2000), IEEE Computer Society, New York, NY (2000)
26. Winter, R., Fischer, R.: Essential Layers, Artifacts, and Dependencies of Enterprise Architecture. Journal of Enterprise Architecture, vol. 3, no. 2, pp. 7-18 (2007)
27. Dietz, J.L.G.: Enterprise Ontology – Theory and Methodology, Berlin, Heidelberg: Springer (2006)
28. Aier, S., Riege, C., Winter, R.: Unternehmensarchitektur – Literaturüberblick und Stand der Praxis. Wirtschaftsinformatik, vol. 50, no. 4, pp. 292-304 (2008)
29. Aier, S., Riege, C., Winter, R.: Classification of Enterprise Architecture Scenarios – An Exploratory Analysis. Enterprise Modelling and Information Systems Architectures, vol. 3, no. 1, pp. 14–23 (2008)
30. Leppänen, M., Valtonen, K., Pulkkinen, M.: Towards a Contingency Framework for Engineering an Enterprise Architecture Planning Method. In Proceedings of 30th Information Systems Research Seminar in Scandinavia (IRIS 2007) (2007)
31. Ylimäki, T.: Potential Critical Success Factors for Enterprise Architecture. Journal of Enterprise Architecture, vol. 2, no. 4, pp. 29–40 (2006)
32. Aier, S., Gleichauf, B., Winter, R.: Understanding Enterprise Architecture Management Design – An Empirical Analysis, to appear in Proc. Wirtschaftsinformatik 2011, ACM (2011)
33. Hair Jr, J.F., Black, W.C., Babin, B.J., Anderson, R.E., Tatham, R.L.: Multivariate Data Analysis, 6 ed., Upper Saddle River, New Jersey: Pearson Prentice Hall (2006)
34. Aziz, S., Obitz, T., Modi, R., Sarkar, S.: Enterprise Architecture: A Governance Framework - Part I: Embedding Architecture into the Organization, Infosys (2005)
35. Aziz, S., Obitz, T., Modi, R., Sarkar, S.: Enterprise Architecture: A Governance Framework - Part II: Making Enterprise Architecture Work within the Organization, Infosys Technologies Ltd. (2006)
36. Winter, R.: Design of Situational Artefacts – Conceptual Foundations and their Application to IT/Business Alignment. Will appear in Proc. ISD 2010 (2010)