An Assessment Method for Selecting an SOA Delivery Strategy: Determining Influencing Factors and Their Value Weights

Joeri Terlouw[1,2], Linda Terlouw[3,4], and Slinger Jansen[2]

[1] Accenture B.V., Gustav Mahlerplein 90, 1082 MA Amsterdam, The Netherlands,
joeriterlouw@gmail.com
[2] Utrecht University, Padualaan 14, 3584 CH Utrecht, The Netherlands
slinger@cs.uu.nl
[3] Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands
[4] Icris B.V., Martinus Nijhoffhove 2, 3437 ZR Nieuwegein, The Netherlands,
linda.terlouw@icris.nl

Abstract. Organizations should carefully consider which SOA delivery strategy, for instance top-down or bottom-up, to follow in migrating toward a service-oriented environment. Selecting a suboptimal strategy can result in spending more time and money than required, or in complete failure of the SOA project. However, selecting one is not easy. Organizations are often unaware of the existence of the different strategies and their situation-dependent pros and cons. Also, it is impossible for organizations to make a well-founded choice since a method for selecting an SOA delivery strategy is lacking. This paper bridges that gap by proposing an assessment method to select a delivery strategy based on specific characteristics of an organization. The method comprises a matrix that includes the influencing factors with their corresponding value ranges, and a weight calculation to determine their impact. Another contribution of this paper is the elicitation of four different delivery strategies that have never been chartered properly.

1 Introduction

Service-Oriented Architecture (SOA) is not a product one can buy in a store as some software vendors may imply. Neither does an organization become service-oriented overnight. SOA means more than *Just a Bunch of Web Services* (JaBoWS); it is a way of structuring and integrating IT systems. Reusable services replace point-to-point connections and the notion of *orchestration* explicitly links business processes to process logic of automated systems. SOA projects usually span multiple years. During the initial phase the benefits of an SOA project do not outweigh its costs. It takes time and effort to achieve the main premises of SOA, i.e. increased reuse of IT artifacts and increased flexibility. Organizations see these premises as means to achieve cost reduction and increased turnover.

Many roads lead to Rome, when it comes to SOA. We call these paths *SOA delivery strategies*. It is generally not evident what strategy to follow. People

active in the business or IT domains of an organization are often unaware of the existence of these different strategies. Even if they are aware, they lack the means to determine the right strategy upfront. This results in spending more time and money than needed, or, even worse, in the selection of a strategy that leads right back homeward instead of to Rome.

The contribution of this paper is a *Delivery Strategy Assessment Method* (DSAM) that can determine the most appropriate SOA delivery strategy for an organization. The method takes organizational characteristics as inputs. Based on these characteristics the method proposes one of the following delivery strategies: *top-down*, *bottom-up*, *meet-in-the-middle*, or *middle-out*.

In science as well as industry, we see several methodologies for service-orientation. The methods, that are described in most detail and are published in articles, are *Service-Oriented Architecture Framework* (SOAF) [1], developed by the Indian consulting company Infosys Technologies, *Service-Oriented Modeling and Architecture* (SOMA) as proposed by IBM [2], and the method of Papazoglou and Van Den Heuvel [3]. We classify SOAF (or, more precisely: its so-called execution view) as a meet-in-the-middle strategy. SOMA utilizes aspects of multiple delivery strategies, i.e. the top-down, bottom-up, and middle-out strategy. The method from Papazoglou and Van Den Heuvel specifies in a large amount of detail how to execute certain activities, e.g. service interface specification and service deployment. For the realization of processes and services, they explain how to apply top-down, bottom-up, as well as meet-in-the-middle strategies.

This paper is structured as follows. Section 2 describes the research design, which conforms to the *design science research paradigm*. Section 3 provides an overview of the SOA delivery strategies. In Section 4 we propose our assessment method for selecting one of these strategies, and we evaluate the method in Section 5. Section 6 concludes this paper by summarizing and discussing its results.

2 Research Design

Hevner, March, and Park [4] state behavioral science and design science characterize much of the research in the information systems discipline. Behavioral science comes from natural science research methods that explain how and why things are. March and Smith [5] state natural science tries to understand reality, whereas design science attempts to create things that serve human purposes. Rather than producing general theoretical knowledge, design research produces and applies knowledge of tasks or situations in order to create effective artifacts. Its products are of four types, i.e. constructs, models, methods, and implementations. In our research, the artifact is a method for selecting an SOA delivery strategy.

In our research we define, following Brinkkemper [6], a method as an approach to perform a systems development project, based on a specific way of thinking, consisting of directions and rules, structured in a systematic way in development

activities with corresponding development products. In this paper, we apply the method engineering methodology to depict each SOA delivery strategy.

To construct our artifact we follow the general design cycle [7]. The problem we aim to solve is an *efficiency problem* as well as an *effectiveness problem*. Selecting an SOA delivery strategy that does not fit the organization's circumstances leads to one of the following consequences: (i) the organization needlessly spends extra money and time that could have been avoided by selecting a more efficient strategy, (ii) the organization does not end up with a service-oriented environment, since a more effective strategy should have been selected. Suggestions come from the fields of *method engineering* and *SOA delivery strategies*. We use the meta-process modeling and meta-deliverable modeling techniques as a means for comparing the SOA delivery strategies. Our design contains the concept of a *Delivery Strategy Factor Matrix* (DSFM) that plots the values of certain organizational characteristics against delivery strategies. The research was conducted in three different phases. In the first phase we identify the *delivery strategies* by literature research, and model them using the scientific discipline of method engineering. During the development phase, a round of interviews with SOA consultants and enterprise architects leads to a selection of the *factors* (i.e. the organizational characteristics that influence the choice for a delivery strategy) and their *value ranges* (i.e. the values that a factor can have). In the second phase, the obtained values of each factor were plotted against the identified strategies to construct the DSFM. A second interview round with a different set of SOA consultants and enterprise architects follows, where *weights* are assigned to the DSFM indicating the impact of factors, and their value weights. We use these weights for the construction of the DSAM. The method takes the factor values of a specific organization as inputs and delivers an output. This output is an absolute score between 0% and 100% for each delivery strategy indicating to what extent it matches the situation. In the third and final phase, we evaluate our method by applying the DSAM to real-life SOA projects. We asked the representatives about their experiences with the actual adopted strategy, and whether they would change strategies if they had the chance.

3 Delivery Strategies

This section discusses the four delivery strategies that organizations can choose for adopting SOA, viz. top-down (3.1), bottom-up (3.2), meet-in-the-middle (3.3), and middle-out (3.4). We provide process-deliverable diagrams depicting the activities and deliverables for each strategy. We apply the method engineering discipline in which activities are represented by rounded rectangles that are colored. They can contain additional sub activities that are white. Deliverables, the end-products of activities, are depicted by squared rectangles. They are linked to their corresponding activity with a broken arrow. Activities can also have their sub activities or deliverables presented elsewhere. A process-deliverable diagrams exhibits these "collapsed" activities and deliverables by a rectangle laying on top of a white one. The sequence of activities is displayed

by arrows that indicate if activities are performed sequentially, in parallel, or depend on a condition.

3.1 Top-down

Figure 1 depicts the activities and deliverables of the top-down strategy as described by Erl [8]. It is an "analysis" first approach; this means it starts with defining one or more enterprise business models. These models can either represent the current or desired state of business operations. In other words, the model shows the ongoing recurring activities involved in the current or future running of a business for the purpose of producing value for the stakeholders. The actual physical or technical environment that the organization relies on, and its constraints, are not taken into consideration according to Marks and Bell [9]. They describe top-down similarly, but use different terms, i.e. examine, design, build, apply, and realize instead of perform top-down analysis, service-oriented analysis, service-oriented design, and service code activities.

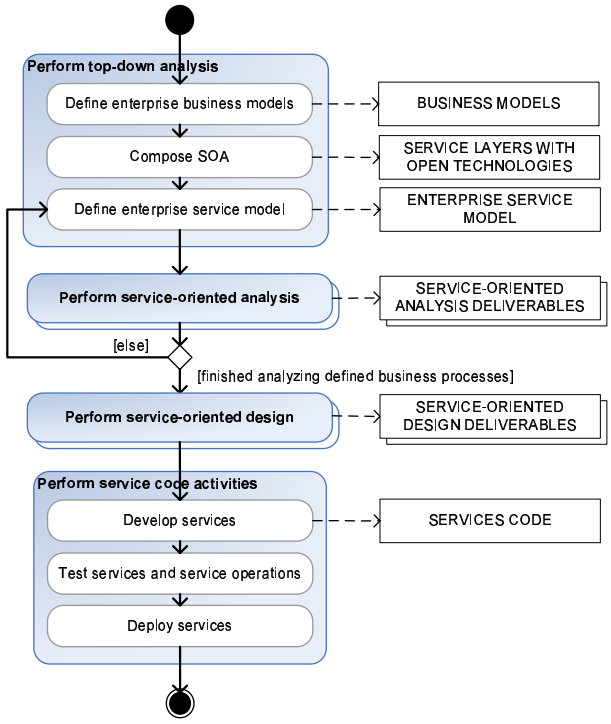

Fig. 1: Activities and Deliverables of Top-Down Strategy

Figure 2 exhibits the decomposition of the service-oriented analysis, which includes service-oriented modeling activities. Figure 3 depicts service-oriented de-

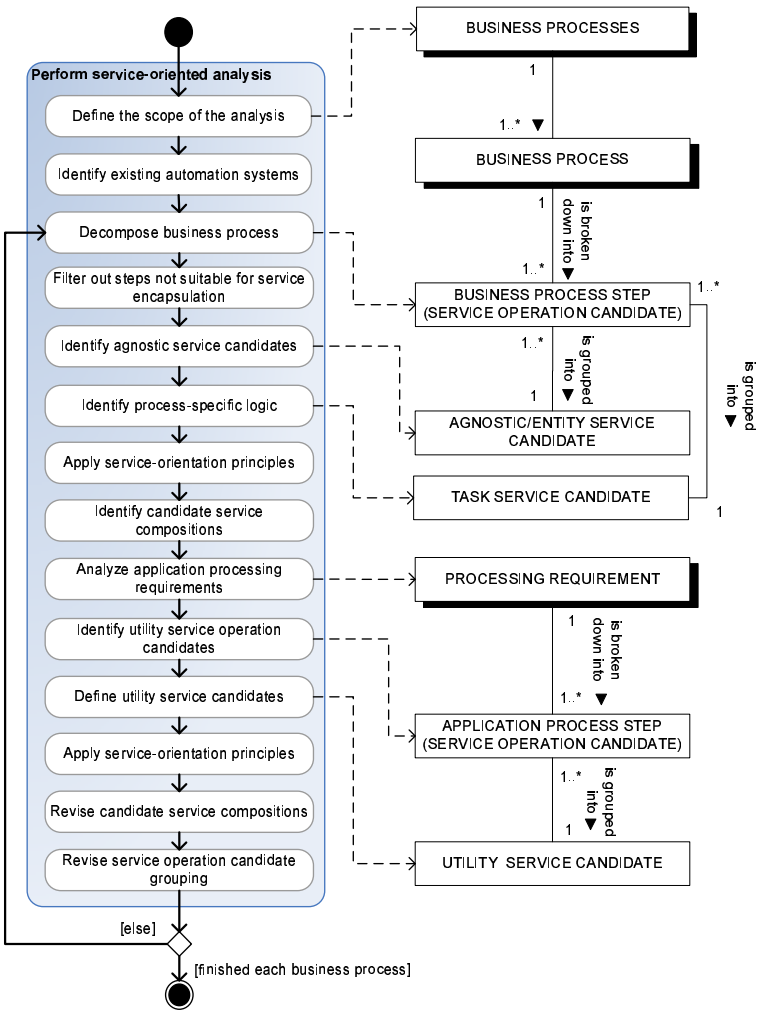

Fig. 2: Activities and Deliverables of Service-Oriented Analysis

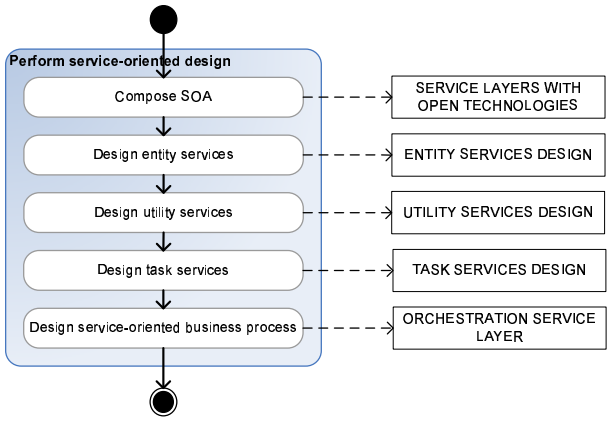

Fig. 3: Activities and Deliverables of Service-Oriented Design

sign. We have depicted them in separate figures since they play a part in both the top-down, and the meet-in-the-middle strategy. Service-oriented analysis first defines what business processes should be automated to conform to current mature and defined business requirements. Each of the business processes is decomposed into granular steps that are afterward grouped into candidate services. A step can be assigned to a *task candidate service* that specifically belongs to the business process. Otherwise, it is unaware (agnostic) of the process, and often linked to a certain business entity, like an invoice or employee. These *agnostic* (or *entity*) *candidate services* are likely to be used in multiple business processes. *Utility service candidates* are also created that could encapsulate granular processing steps of application requirements resulting from the analysis of business process steps. The decomposition activity is repeated for every business process, leading to revised task, entity, and utility service candidates.

Service-oriented design begins with defining abstraction layers that can encapsulate certain services, depending on their type of logic, potential reuse, and relation to existing domains within the organization. The entity, utility, and task services derived from the final service candidates are designed. Depending on its necessity an additional orchestration layer can be created, representing a business process definition hosted within an orchestration platform. This results in the formal, executable definition of work flow by creating a Web Services Business Process Execution Language (WS-BPEL) process definition. As a last phase, top-down deals with service code.

3.2 Bottom-up

The bottom-up strategy encourages the creation of services as a means of fulfilling application-centric requirements. Marks and Bell [9] state it is a progressive process of building services or assembling existing technologies to provide business solutions. The bottom-up strategy can tie services to their originating technology environments. This leads to tight coupling. Figure 4 depicts the steps that are performed according to Erl [8] during a bottom-up approach.

The first bottom-up sub activity consists of modeling the application requirements that can be fulfilled through the use of services. The second sub activity focuses on the design of these utility services, which can come into existence in several ways. They may be delivered by third-party *wrapper services* or *auto-generated proxy services*. Wrapper services can be used for legacy system integration purposes that expose legacy functionality to service requesters. However, custom utility services may also be constructed, which require a design process where existing design standards are applied. This could lead to more service-oriented utility services, because of their potential reusability. It should be noticed that Erl [10] recommends that, at minimum, a high-level service inventory blueprint must be defined prior to creating physical service contracts. Finally, these utility services are developed, tested, and deployed in essentially the same way as in the top-down strategy.

Marks and Bell [9] also mention a *bottom-up design and analysis approach*. Unlike the approach mentioned by Marks and Bell, the bottom-up delivery strat-

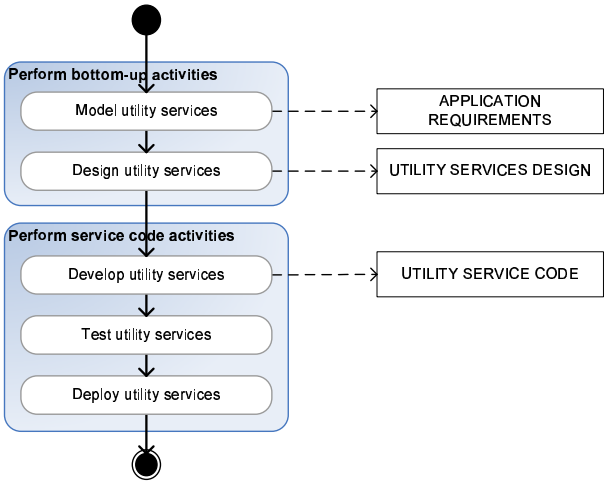

Fig. 4: Activities and Deliverables of Bottom-Up Strategy

egy of Erl [8] assumes the business requirements have already been collected. Furthermore, Marks and Bell refer to application requirements as business requirements, and assume the incorporation of design standards. They also assume business logic to be incorporated, and call the development "design and construct". Together with the absence of service deployment and testing, these points make it hard to relate it to the steps as proposed by Erl.

3.3 Meet-in-the-middle

The method as proposed by Marks and Bell [9] advocates a delivery strategy using both a top-down and bottom-up approach, applied in an iterative fashion. The services identification process is conducted in a top-down fashion, with a focus on *candidate business services* as in Figure 2. In this case, current physical or technical environments should be disregarded. Instead, attention should be given to the organization's operating units, and the relationships between those units. Marks and Bell propose to perform service construction itself in a bottom-up fashion. An iterative process follows to bring both approaches together. The approach of Marks and Bell resembles the meet-in-the-middle delivery strategy as proposed by Erl [8]. Figure 5 depicts the steps of this strategy.

Meet-in-the-middle starts with a top-down approach. The top-down analysis differs from the top-down delivery strategy in the sense that it is an ongoing effort to further achieve the enterprise-wide analysis goals. When the analysis is sufficiently progressed, service-oriented analysis is also initiated using the available business models, and other top-down analysis results. Service-oriented design and service code activities follow like in the top-down strategy. Since an ongoing top-down analysis is executed during these steps, services are subject to

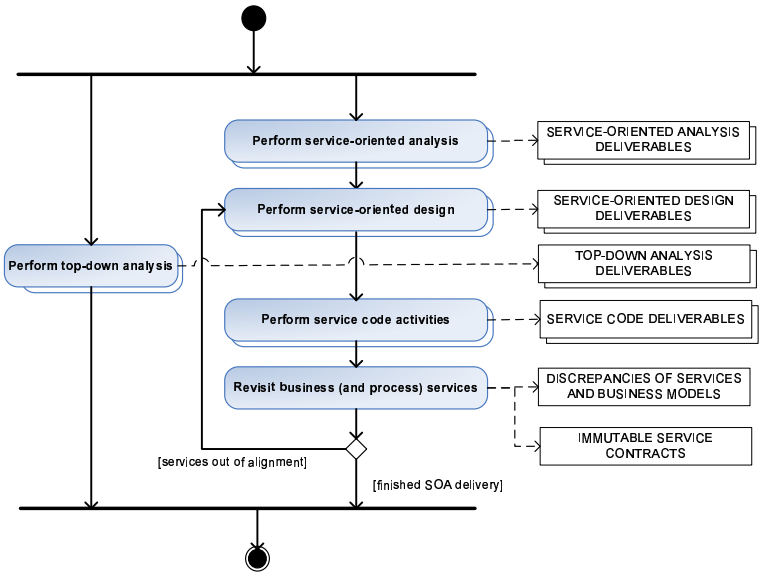

Fig. 5: Activities and Deliverables of Meet-in-the-Middle Strategy

revision. Meet-in-the-middle therefore requires an extra activity, in which periodic reviews are performed to compare the design of services against the current state of the business models. From this point on, an iterative process follows. The service-oriented design and service code activities are repeated for the services out of alignment, followed by another revision.

3.4 Middle-out

Figure 6 depicts the middle-out strategy according to Rosen et al. [11]. They state iterations take place within and between each activity, instead of them being linear. The path on the other hand depends on whether new business capabilities are created or existing capabilities are used. Since they do not mention how, the figure provides every possible iteration.

The middle-out delivery strategy produces both higher-level business and information architecture and design artifacts, and working and deployed services. Rosen et al. explicitly mention roles and state every activity (i.e. business modeling, design, etc.) focuses on a special goal, like modeling the business context, or enabling services for use in solutions. An activity should not address or be influenced by concerns of other activities. Therefore, middle-out has no sequential activities, unlike the previous strategies. Performing independent activities is enabled by a stable center, called the *reference architecture*. This extra element provides the context for what the different architectures (business, information, application and technology) describe, what they look like, how they are related to each other, and how they work together to meet overall business goals. The

reference architecture provides an overall taxonomy that defines the different types of services together with service groupings. Furthermore, responsibilities are assigned to each activity to help shape the overall service road map. It also provides proven design patterns, different types of applications and services, and technology standards and mappings for service implementations. The reference architecture provides the link between top-down and bottom-up aspects.
Since the terminology of activities by Rosen et al. differs from Erl, method comparison of van de Weerd et al. [12] was applied to display how the activities of both authors relate to each other. Method comparison shows meet-in-the-middle has an additional activity, which produces a reference architecture in an early stage or updates it at a later time to make it current. This reference architecture ensures that activities with separate concerns can be performed in a parallel fashion, without the need for intermediate activities that link those concerns as found in other strategies. It therefore has less activities.

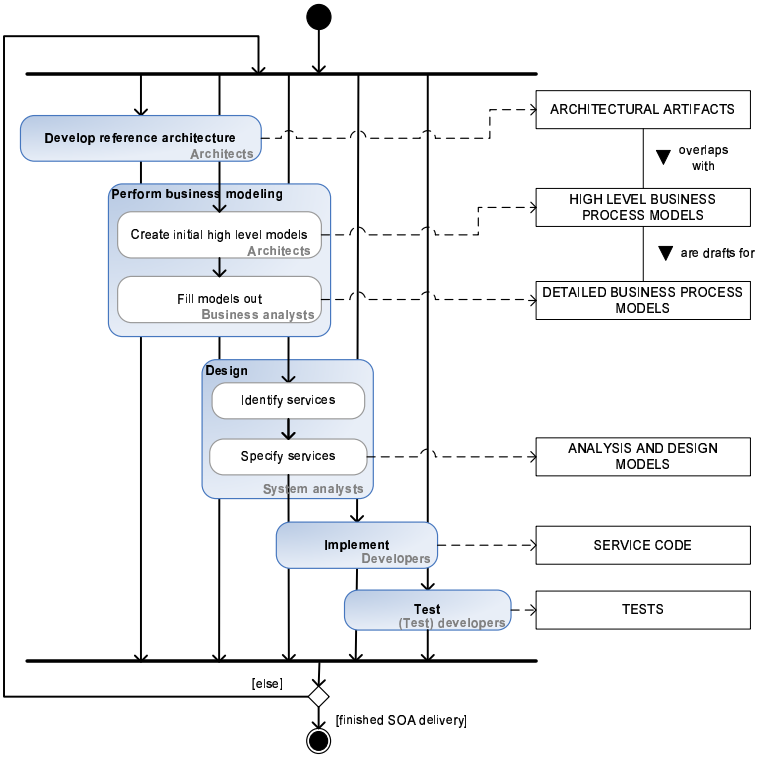

Fig. 6: Activities and Deliverables of Middle-Out Strategy

4 Delivery Strategy Assessment Method

The result of our design science development phase is the DSAM. The two bases for this method are the DSFM and the weights assigned to the combinations of factor value and delivery strategy. During a first round of interviews, SOA consultants and enterprise architects indicated which factors can possibly influence the success of a certain delivery strategy. In Table 1 we see the results of this interview round in the form of twelve boxes (the factors) and the possible values a factor can have for a certain delivery strategy (the list in the box). During the second interview phase, other SOA consultants and enterprise architects weighted each combination of factor value and delivery strategy, resulting in the numbers depicted in Table 1.

4.1 Factors and Value Ranges

Let us have a closer look at these factors and their value ranges (reading the table from left to right and top to bottom). The *CMM maturity level* refers to the maturity of processes related to software development. The possible values range from level 1 to level 5 as described by the SEI [13]. From the interviews we learned that it matters whether an organization chooses for product leadership or operational excellence. We have named this *value discipline* following Treacey and Wiersema [14]. From their work, we found one missing factor value, i.e. customer intimacy. We have classified the *organization size* based on the amount of people working for the organization. The value range is based on headcount levels of the Commission of the European Communities [15]. The next factor, *flexibility/stability*, is concerned with two types of flexibility/stability. On the one hand we look at the flexibility related to the organization's internal changes, on the other hand at changes caused by its clients' changing wishes or demands. We created three values for the factor *enterprise architect involvement*, i.e. there is no architect present, he is sometimes present, and he is always present. For the *influence of sponsors/stakeholders* we can distinguish between projects initiated from the management side, technical side, or both. Another factor is the currently existing application landscape within an organization, called *legacy systems*. Within an SOA project legacy systems are either required, useful, or not present at all (greenfield situation). Also, *structural change* has an impact on choosing the delivery strategy. Though every SOA needs an organizational change at the end of the day, there are different ways with implementing this change. *SOA focus* refers to the preferred way of distributing SOA throughout the organization. One possible focus is to start out with making a solid service foundation at one department, and gradually spreading the SOA environment toward other departments until it is enterprise-wide. Another focus is developing services enterprise-wide to support a large scope of business processes, and improving the services at a later stage. Because often external consultants are involved in SOA projects, we add the factor *client relationship*. This factor describes the amount in which the organization is willing to implement the

consultants' advice. *Technical maturity* refers to the "newness" of the technology of the organization. We have used a scale ranging from obsolete to bleeding edge [16]. The final identified factor is concerned with whether an organization delivers tangible products or intangible products (also called services, though in a different meaning than in the rest of this paper). This is called *economic sector* in the factor matrix.

4.2 Weights

During the second phase of the research a second round of interviews was conducted, resulting in Table 1. We have assigned weights to the factors themselves and to factor value and delivery strategy pairs by averaging the individual answers of the interviewees. The weights represent the amount of influence on a five-point Likert scale, ranging from 1 (very small influence) to 5 (very big influence). We see that four factors have a weight higher than 3.5. For ruling out very high or very low individual responses, we also calculated the standard deviations (not depicted in the table). We grouped them into the categories "low consensus" (with a high standard deviation) and "high consensus" (with a low standard deviation). Based on these calculations we can say the most influential factors are respectively the presence of an enterprise architect in an SOA project, whether the sponsors or stakeholders are from the management or technical side, and whether the organization is willing to change its current organizational structure. Though flexibility/stability of an organization has the same average as the willingness to change, it shows a low consensus among respondents unlike the other three factors.

Looking at the average weights and the standard deviations for combinations of factor value and delivery strategy, we can extract some general findings about which delivery strategy to apply in a certain situation. Table 2 exhibits the four SOA delivery strategies and the organizational characteristics to which they have the best fit. The asterisk indicates that although the average weight is high, the standard deviation is also high.

5 Evaluation

We have evaluated the DSAM by applying the method to five real-life SOA projects. Taking the organizational characteristics as inputs, the method proposed an optimal strategy. Because it is impossible to execute the project again using this optimal strategy (in case it differs from the actual taken approach), we took the following approach. We clarified the different strategies to the people representing the SOA projects. Then we proposed the optimal strategy according to the DSAM and explained how we got to this advice. In case the proposed optimal strategy equals the actual taken approach, we asked the representative whether he would, looking backwards, take the same approach. Otherwise, if the actual taken and the proposed optimal strategy differ, we asked the respondent the same question, and we additionally we asked whether he regarded the proposed solution as a better strategy.

Table 1: Delivery Strategy Factor Matrix

Influence of CMM maturity level on choosing a delivery strategy:				
3.2	T	B	M	O
Level 1: Initial	0.0	2.2	0.8	0.0
Level 2: Managed	1.0	3.0	1.0	0.2
Level 3: Defined	1.8	1.0	2.2	0.4
Level 4: Quantitatively Managed	0.2	0.2	2.0	2.0
Level 5: Optimizing	1.0	0.0	0.0	3.0

Influence of value discipline on choosing a delivery strategy:				
3.0	T	B	M	O
Product leadership	2.4	0.0	0.0	0.0
Operational excellence	1.2	0.6	0.6	1.2
Customer intimacy	0.6	0.0	1.8	1.8

Influence of organization size on choosing a delivery strategy:				
3.0	T	B	M	O
Micro (<10 headcount)	2.0	2.6	0.4	0.0
Small (<50 headcount)	2.0	2.0	1.0	0.0
Medium-sized (<250 headcount)	0.6	0.0	1.2	2.4
Large (250 headcount or more)	1.0	0.0	0.6	2.4

Influence of flexibility/stability on choosing a delivery strategy:				
3.6	T	B	M	O
Flexible to customers and business	0.4	0.6	0.8	3.0
Flexible to customers	1.6	0.4	0.8	0.8
Flexible to business	0.0	1.8	0.8	1.8
Stable to customers and business	2.4	0.4	0.8	0.0

Influence enterprise architect on choosing a delivery strategy:				
4.4	T	B	M	O
No EA present	1.6	2.0	0.0	0.8
Partly an EA present	1.0	0.0	1.6	2.8
Always an EA present	1.8	0.0	1.0	3.6

Influence of sponsors/stakeholders on choosing a delivery strategy:				
4.2	T	B	M	O
Project from management side	3.2	1.0	0.0	0.0
Project from technical side	0.0	2.4	0.0	2.6
Project from both sides	1.8	0.0	0.6	3.2

Influence of legacy systems on choosing a delivery strategy:				
3.0	T	B	M	O
Greenfield situation (no legacy)	3.0	0.0	0.0	0.6
Could make use of legacy	1.0	1.4	1.4	2.0
Must make use of legacy	0.0	0.6	1.4	3.0

Influence of structural change on choosing a delivery strategy:				
3.6	T	B	M	O
Willing to change structure	1.6	0.0	1.2	3.6
Partly willing to change structure	0.4	0.8	2.0	2.4
Not willing to change structure	1.2	1.6	0.8	0.0

Influence of SOA focus on choosing a delivery strategy:				
3.2	T	B	M	O
Focused on services foundation	0.0	1.6	1.8	0.6
Focused on processes automation	3.2	0.0	1.0	0.8
Focused on both	0.6	0.0	0.4	3.2
Focused on neither	0.0	0.6	0.4	0.4

Influence of client relationship on choosing a delivery strategy:				
3.2	T	B	M	O
Implements consultants' advice	0.0	0.8	0.4	1.8
Uses advice to decide itself	0.0	0.0	0.8	1.4
Resistant toward advice	0.6	1.6	0.8	0.0

Influence of technical maturity on choosing a delivery strategy:				
2.6	T	B	M	O
Obsolete	0.6	0.6	0.0	1.2
Dated	1.4	0.6	0.0	0.4
State of the art	0.0	0.4	0.6	2.4
Leading edge	0.6	1.8	0.0	1.6
Bleeding edge	0.6	1.4	0.0	0.0

Influence of economical sector on choosing a delivery strategy:				
2.8	T	B	M	O
Product organization	1.4	0.4	0.0	0.4
Service organization	1.2	0.0	0.8	2.6
Product and service organization	0.0	0.0	1.6	2.6
Neither	0.0	0.0	0.0	0.0

1 Very small influence
2 Small influence
3 Average influence
4 Big influence
5 Very big influence

T Top-down delivery strategy
B Bottom-up delivery strategy
M Meet-in-the-middle delivery strategy
O Middle-out delivery strategy

Table 2: SOA Delivery Strategies and Their Matching Factor Values

Factor	**Top-down**	**Bottom-up**	**Meet-in-the-middle**	**Middle-out**
CMM maturity		level 1* or 2*	level 3* or 4*	level 4* or 5*
Value discipline	product leadership			
Organizational size	micro or small	micro or small		medium-sized or large
Flexibility/stability	stable to both*			flexible to both*
Enterprise architect involvement		not present*		partly or always present*
Sponsors/stakeholders	management side	technical side*		technical side* or both sides
Legacy systems	greenfield			could make use of legacy* or must make use of legacy
Structural change			partly willing to change*	partly willing to change* or willing to change
SOA focus	processes			both services and processes
Client relationship				
Technical maturity				state of the art
Economical sector				services or both

* indicates a low consensus among interviewees (high standard deviation)

The adopted strategy of a first telecom company was middle-out. Likewise, the proposed strategy was middle-out. Its score exceeded the other delivery strategies by a factor of two to three. Currently, the project is still running, but not expanding to enterprise-wide level. The respondent claimed the expansion problems are due to the change of people involved in the project and it was not caused by a wrong choice of delivery strategy. The respondent still supports the choice for the middle-out strategy, and he would not change strategies if he could start over again.

For a second telecom company a middle-out delivery strategy was proposed by the DSAM while the adopted strategy was bottom-up. The middle-out strategy scored higher than each of the other delivery strategies by a factor of two to four. The project was canceled, due to a constant changes of people involved in the project. It is however unknown if the project would have failed with the strategy proposed by the assessment method, i.e. the middle-out delivery strategy. The respondent indicated that, if he could start over again, he would take either the bottom-up or the middle-out strategy.

The transportation company has followed a bottom-up approach. The DSAM proposed a middle-out strategy. Due to the company size (>250 headcount) the company ended up with a large amount of services. Because the services were "just" created from legacy systems without thinking about the overall architecture, the situation soon became difficult to manage. Would they do the project over again, the respondent would take either a middle-out or a meet-in-the middle approach. The company did not consider a top-down approach, since they have a lot of legacy systems to take into account. Also, their SOA focus is on a solid service foundation and not on enterprise-wide process support.

While the industrial company has taken a top-down approach, the DSAM proposed a middle-out strategy. In this case the top-down approach was the second choice (with a difference of 10%). For this company the top-down approach led to the expected results. The respondent did not think a middle-out strategy would have resulted in reduction of costs or project duration.

The governmental organization applied a meet-in-the-middle approach. The proposed solution was top-down, having middle-out as a very close runner up (3% difference). According to the respondent the meet-in-the-middle approach did not work for the organization. They got into trouble when aligning the candidate services from the initial top-down phase with the bottom-up created services. Reconsidering, the respondent would have followed a top-down approach, because the middle-out strategy resulted in unnecessary alignment problems in their greenfield situation (they did not have legacy systems to worry about).

From these five cases we derive that making the delivery strategy choice explicit is itself a contribution to any SOA delivery project. The DSAM selects the right strategy in each of the five cases, although other critical success factors play a large role in the outcome of these projects. We consider further validation of the framework as future work.

6 Conclusions

In this paper we presented the DSAM, a method for selecting the optimal SOA delivery strategy for an SOA delivery project. We obtained the factors determining the suitability of a certain strategy, and their value ranges in a first round of interviews with SOA consultants and enterprise architects. In a second round of interviews a different set of consultants and architects gave weights to each SOA delivery and factor value combination. Based on average weight of the factors, the most influential factors for selecting a delivery strategy are: (i) the presence of an enterprise architect, (ii) whether the sponsors or stakeholders are from the management or technical side, and (iii) whether the organization is willing to change its current organizational structure. To rule out the effects of extreme high or low weights from individual interviewees, we also calculated the standard deviation of the weights. We have classified the weights in two groups: the first having a low standard deviation, the second having a high standard deviation.

Three factors in the matrix are in our eyes subject to discussion: enterprise architect involvement, client relationship, and organizational size. The first two were often regarded as highly influential. This high ranking may be due to subjectivity, because the interviewees fulfill the roles of SOA consultant and enterprise architect (and thus ranking the importance of their own work). A possibility is to also interview other stakeholders in SOA projects like CEOs and CIOs. The third discussion point concerns the organizational size. We classified the size in five categories according to the headcount levels of the Commission of the European Communities [15]. When evaluating the DSAM we found that this is probably not the best scale to use. It can make a difference whether the company is a large company of 300 people or a multinational of 30.000 people. At this

moment, both companies have the same value for the factor company size. We intend to change this in a next version of the factor matrix.

The DSAM aids organizations in selecting an SOA delivery strategy. Based on characteristics of the organization the DSAM proposes the optimal SOA delivery strategy for the organization. The basis on which the decision is made is completely traceable, giving the organization insight into the selection path and the relative importance of influencing factors.

References

1. Abdelkarim Erradi, Sriram Anand, and Naveen N. Kulkarni. Soaf: An architectural framework for service definition and realization. In *IEEE SCC*, pages 151–158. IEEE Computer Society, 2006.
2. A. Arsanjani et al. Soma: a method for developing service-oriented solutions. *IBM Syst. J.*, 47(3):377–396, 2008.
3. Michael Papazoglou and Willem-Jan van den Heuvel. Service-oriented design and development methodology. *Int. J. Web Engineering and Technology*, 2(4):412–442, 2006.
4. Alan R. Hevner, Salvatore T. March, and Jinsoo Park. Design Science in Information Systems Research. *MIS Quarterly*, 28:75–99, 2004.
5. Salvatore T. March and Gerald F. Smith. Design and natural science research on information technology. *Elsevier Science*, Decision Support Systems 15:251–266, 1995.
6. Sjaak Brinkkemper. Method engineering: engineering of information systems development methods and tools. *Inf. and Software Technology*, 38(4):275–280, 1996.
7. Hideaki Takeda, Paul Veerkamp, Tetsuo Tomiyama, and Hiroyuki Yoshikawa. Modeling design processes. *AI Mag.*, 11:37–48, 1990.
8. Thomas Erl. *Service-Oriented Architecture (SOA): Concepts, Technology, and Design (The Prentice Hall Service-Oriented Computing Series from Thomas Erl)*. Prentice Hall PTR, 2005.
9. Eric A. Marks and Michael Bell. *Service-Oriented Architecture (SOA): A Planning and Implementation Guide for Business and Technology*. Wiley, 2006.
10. Thomas Erl. *SOA Principles of Service Design (Prentice Hall Service-Oriented Computing Series from Thomas Erl)*. Prentice Hall PTR, 2007.
11. Michael Rosen et al. *Applied SOA: Service-Oriented Architecture and Design Strategies*. Wiley, 2008.
12. Inge van de Weerd, Stefan de Weerd, and Sjaak Brinkkemper. *Situational Method Engineering: Fundamentals and Experiences*, chapter Developing a Reference Method for Game Production by Method Comparison, pages 313–327. Springer Boston, 2007.
13. CMMI Product Team. Cmmi for systems engineering/software engineering/integrated product and process development/supplier sourcing, version 1.1. Technical report, Carnergie Mellon Software Engineering Institute, 2002.
14. Michael Treacy and Fred Wiersema. *"The Discipline of Market Leaders: Choose Your Customers, Narrow Your Focus, Dominate Your Market"*. Basic Books, 1997.
15. European Commission, editor. *The New SME Definition: User Guide and Model Declaration*. Office for Official Publications of the European Communities, 2005.
16. E.T.B. Johnston. Editorial. *Potentials, IEEE*, 26:4–4, 2007.