Analysis of Enterprise Architecture Evolution Using Markov Decision Processes

Sérgio Guerreiro[1,2(✉)], Khaled Gaaloul[3], and Ulrik Franke[4]

[1] Lusófona University, Campo Grande 376, 1749-024 Lisbon, Portugal
sergio.guerreiro@ulusofona.pt
[2] Formetis, Hemelrijk 12c, 5281 PS Boxtel, Netherlands
sergio.guerreiro@formetis.nl
[3] Luxembourg Institute of Science and Technology (LIST), Esch-sur-Alzette, Luxembourg
khaled.gaaloul@list.lu
[4] Swedish Institute of Computer Science (SICS), Isafjordsgatan 22, 164 29 Kista, Sweden
ulrik.franke@sics.se

Abstract. Enterprise architecture (EA) offers steering instruments to aid architects in their decision-making process. However, the management of such a process is a challenging task for enterprise architects, due to the complex dependencies amongst EA models when evolving from an initial to a subsequent state. In this paper, we design, present and analyze an approach supporting EA model evolution. In doing so, we define EA artifacts dependencies and model their corresponding evolutions during change. Then, this model is processed using a feedback control schema to fully inform the EA design decisions. An access control model for an inventory case study is introduced to reason on issues connected to this evolution. The results obtained by a stochastic solution (Markov Decision Processes) are used to argue about the usefulness and applicability of our proposal.

Keywords: Enterprise Architecture · Evolution · Markov Decision Processes

1 Introduction

Enterprise architecture (EA) is a discipline driving change within organizations. EA provides a mechanism for cohesive steering and provides management with appropriate indicators and controls to steer the transformation of an enterprise into the desired direction [1, 2]. In the past, EA practice had focused primarily on the technological aspects of change, but the practice is quickly evolving to use a rigorous business architecture approach to address the organizational and motivational aspects of change as well [3].

The need for enterprises to constantly adapt to ever changing requirements of the environment is a continuing field of research in enterprise engineering. The notion of engineering focuses on applying a systematic approach to enterprise transformation. The transformation of enterprises is engineered by the means of appropriate models and

R. Pergl et al. (Eds.): EOMAS 2016, LNBIP 272, pp. 37–51, 2016.
DOI: 10.1007/978-3-319-49454-8_3

methods [4]. Enterprise modeling provides adequate means for the description of As-is and To-be states of enterprises. Thus, enterprise models integrate the conceptual models of information systems with models representing organizational and technical structures [5].

However, current EA frameworks do not support this transformation appropriately due to inflexible EA models and missing integration of stakeholders in the modeling process [6]. In practice, organizations struggle with transformational change being demanded by the ever-increasing speed of business. In this context, enterprise architects rely on architecture modeling languages to support responsibility and alignment for the new EA models [7], but lack guidelines during EA evolution [3].

The evolution of an EA model itself is just a set of changes to the artifacts contained in this EA model. Architectural artifacts are created in order to describe a system, solution, or state of the enterprise [8]. Thus, EA artifacts document EA components from the business to the IT level. EA provides pragmatic artifacts such as requirements, specifications, and conceptual models; thereby providing information for enterprise architect when dealing with models change and decision-making.

In this paper, we define a model-driven approach to support EA evolution. The main idea is to build an intuitive and powerful paradigm that can help the architect analyze the effects of artifact changes. Such changes trigger events that can be reasoned on, viz. model transitions from As-is to To-be states. As the architect faces several possible, mutually exclusive, To-be design decisions, we offer the prospect of *evaluation* of these decisions in order to identify the best EA model alternatives. These valuations of alternatives are based on observational data and calculations using Markov decision processes. The paper extends our own previous work [18] as detailed in the next section.

This research is based on a simplification of the design-science research (DSR) as proposed by [9, 10]. The methodology applied is divided according to the two processes of design science research in information system: Build and Evaluate. The build process is composed of two stages: model definition and model construction. The first stage encompasses the evolution model based on artifact dependencies in Sect. 1 based on existing research contributions (see Sect. 2). The second stage constructs an organizational dynamics in EA context (inventory case study) to support their evolution process (see Sect. 3). The evaluation process includes a calculation using a linear programming algorithm (Markov decision processes – MDP) in Sect. 4. Finally, we conclude and present future work in Sect. 5.

2 Related Work

One influential strand of research on uncertainty in EA work is that initiated by Johnson *et al.* [11]. Here, the authors introduce three kinds of uncertainties – definitional, causal, and empirical – and propose an extension of influence diagrams to manage them in the EA context. In later work, the same research group has used probabilistic relational models (*e.g.* [12, 13]) and a probabilistic version of the OCL language [14] as tools to describe and manage uncertainty in EA. The same research group has also explored utility theory as a theoretical framework for trading different goods against each other in the EA decision-making context, viz. cost vs. availability.

While our paper is closely related to the work by Johnson *et al.* in the sense that we use probabilities to model EA activities, we differ importantly in our use of the MDP formalism, which explicitly allows us to address problems where the outcomes are only partially under the control of the decision-maker, and, in the partially observable MDP (POMDP) where the outcomes are only partially observable.

Another method proposed in the literature for addressing uncertainty in EA is real options analysis [15]. However, while Mikaelian *et al.* [15] address the problem of using real options holistically, to avoid sub-optimization in organizational silos, we address inherent uncertainties in EA work by means of MDP valuation to choose the best option.

It should also be noted that EA frameworks most often contain mechanisms for dealing with uncertainty, albeit not in a formal and quantitative manner. On the one hand, Quartel *et al.* [16] describe the well-known TOGAF Architecture Development Method (ADM) precisely as a means to address uncertainty and change, in particular the kind of uncertainty inherent in bridging requirements and actual solution. On the other hand, the Information Technology Infrastructure Library (ITIL) describes processes supporting change management to be applied by an organization during service transition [17]. However, such qualitative approaches remain generic using best practices, and thus, differ from our quantitative approach.

To summarize, we address an important problem in a novel way. This paper extends our own previous work [18] in the following aspects *(i)* enforcement of an informed decision-making solution applied for access control governance and *(ii)* evaluation of the rigor of the delivered MDP calculation results using DSR methodology.

3 Modeling Enterprise Architecture Evolution

Why is EA decision-making and evolution such a hard problem? One reason is the complex organizational setting. Almost by definition, EA decision-making takes place at the highest organization level, and aims to align and synchronize a variety of separate processes and organizational entities, each with their own expertise and their own agendas. As noted by [15], this entails a substantial risk of sub-optimization in organizational silos. Another reason is that the problems of EA evolution do not exist a priori, well defined and waiting to be solved, but rather have to be constructed and structured into well-defined problems before they can be solved [19]. A third reason, as alluded to above, is the prevalence of uncertainty. Uncertainty being the rationale for our choice of the MDP formalism, it is worth expanding on: Causal uncertainty [11] is about the effects of actions taken: Even if EA decisions and actions are guided by proper theory, there is always some degree of causal uncertainty involved. If a company switches to a more reliable IT service, what will their resulting availability be? 99.998 %? 99.999 %? Theories addressing such questions typically include uncertainty (*e.g.* [20]). Empirical uncertainty [11] is about the data used. The information used for enterprise decision-making is uncertain, *e.g.* because it is obsolete [21], measured with an imperfect instrument, or subject to some unknown bias. This kind of uncertainty is an important reason for extending the MDP into the POMDP, where the environment is only partially observable. Event uncertainty outside the enterprise is essentially the

kind of uncertainty, which is at the core of standard decision theory: Will supplier A go bust, will product X form a working software ecosystem, and will there be a need to retract thousands of deployed embedded systems for security upgrades? The uncertainty of being successful in changing an EA (from As-Is to To-Be) is a classical problem that is shown in the case study (*cf.* next Sect.), and could occur because of bad implementation, bad interpretation or even a malicious action, or deception, taken by an organization actor. For instance, implementing a non-secure EA model could lead to substantial financial losses.

It is against the background of causal, empirical and event uncertainty that we find the MDP methodology useful as a means for uncertainty management in EA.

The rest of this section follows the DSR methodology. We define the first stage of the build process where the relevant concepts and relations are identified, in order to build a model describing EA evolution and their corresponding operations. Here, we introduce our concrete running example, intended to illustrate the uncertainties described, more generally, above.

We assume that the overarching EA is composed of a multitude of EA models, each being concurrently edited by different modelers (*i.e.* enterprise architects). These architects have different responsibilities and may not be fully aware of the dependencies between the models and their artifacts. (In this sense, the problem addressed is one of separation of concern, which is an important rationale for EA work, *cf.* [22]) This may lead to creating flaws and inconsistencies in EA models, when changes made on given EA models indirectly impact other EA models (see Fig. 1). A typical example of this is when a modeler X modifies the credentials or permissions assigned to a given role, in order to update a business process model Y, but is unaware that this has an impact on another business process model Z where some security requirement is broken by this change (*e.g.* a conflict of interest violation).

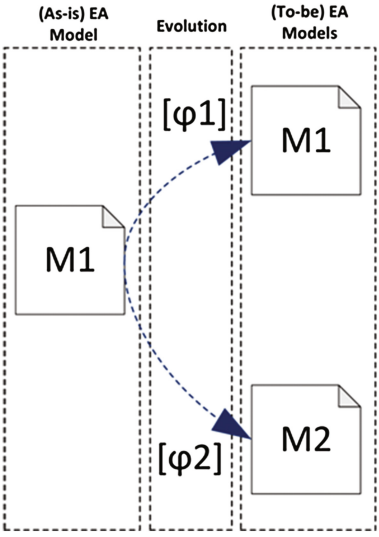

Fig. 1. Model-driven EA evolution

Describing EA evolution will enable us to reason on different alternatives for EA evolutions and thus decide upon alternatives or analyze potential evolutions from a given EA state in an informed manner. Our objective is to assist enterprise architects in deciding which EA evolutions are fully compliant and which ones are not compliant or should be considered as suspicious and need be more thoroughly analyzed by an EA expert. This expert will then have the responsibility of taking a final decision whether the EA evolution should be committed to or reverted.

4 EA Evolution Decision: A Case Study Using Models

In this section, the DSR methodology is applied to the second stage of the build process. Here we use organizational dynamics in EA to process the evolution model as introduced in Sect. 3.

In order to illustrate the need and the benefits to support the EA evolution decision, we present a case study in the field of access control models, in specific, the role-based access control (RBAC) model [23]). RBAC relies on user authentication, which in turn relies on identity management and defines relationships between the main concepts of Users, Roles and Permissions. RBAC's constraints restrict permissions depending on contextual information such as segregation of duties (*SoD*) [24].

To represent the models, DEMO[1] (Design and Engineering Methodology for Organizations) is used. DEMO is a methodology and a theory founded in language action perspective (LAP), and aims at the design, engineering, and implementation of organizations [5]. On the one hand DEMO is compatible with the communication and production, acts and facts that occur between actors in business processes. A DEMO business transaction model has two distinct worlds: *(i)* transition space and *(ii)* state space. On the one hand, the DEMO transition space is grounded in a theory named as Ψ-theory (PSI), where the standard pattern of a transaction includes two distinct actor roles: the Initiator and the Executor. Figure 2 depicts this basic transaction pattern.

The transactional pattern is performed by a sequence of coordination and production acts that leads to the production of the new fact. In detail, it encompasses: *(i)* order phase that involves the acts of request (rq), promise (pm), decline (dc) and quit (qt), *(ii)* execution phase that includes the production act of the new fact itself (depicted by the diamond) and *(iii)* result phase that includes the acts of state (st), reject (rj), stop (st) and accept (ac). Firstly, when a Customer desires a new product, he requests it. After the request for the production, a promise to produce the production is delivered by the Producer. Then, after the production, the Producer states that the product is available. Finally, the Customer accepts the new fact produced. DEMO basic transaction pattern aims at specifying the transition space of a process that is given by the set of allowable sequences of transitions. Every state transition is only dependent on the current states of all surrounding transactions.

The usage of a business transaction oriented methodology has the benefit of narrowing the domain of EA models to a single and self-contained set of models.

[1] http://www.demo.nl/.

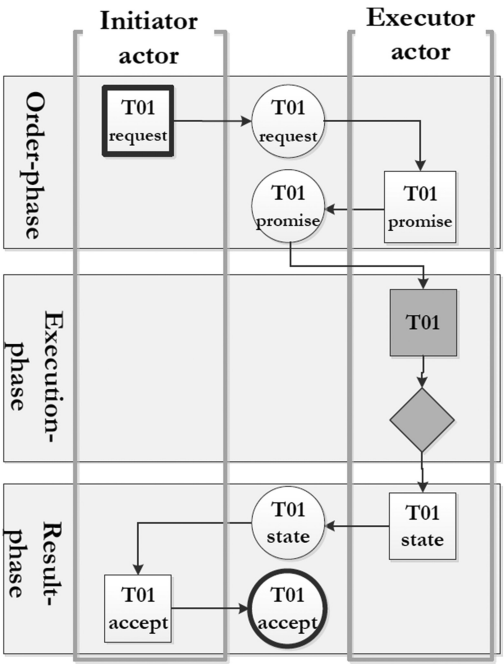

Fig. 2. The DEMO standard pattern of a transaction between two actors with separation between communication and production acts (Adapted from [5]).

4.1 Case Study Explanation

For explanation, the evolvable EA proposal is exemplified using an inventory case study. One person orders goods from suppliers, and another person logs the received goods in the accounting system. This keeps the purchasing person from diverting incoming goods for his own use. To that end, a segregation of duties (SoD[2]) between both users' roles is enforced.

Figure 3 depicts a DEMO Process Model (PM) [5], where each business transaction is an abstraction represented graphically by the cylinders. The goal of performing such a transaction pattern is to obtain a new fact.

As depicted by the PM in Fig. 3-(Model 1), the user U1 (Client) who order the goods (T1) is assigned with role 1 (R1). R1 inherits read/write permissions. The user U2 (Accounter) of the accounting system is assigned with role 2 (R2) to perform log account (T2). R2 inherits read permission. The aforementioned permissions define operations on the accounting system database. Roles and responsibilities definition are dependent on organizational dynamics during EA change.

[2] *SoD* concept prohibits the assignment of role/responsibility to a single person for the acquisition of assets, their custody, and the related record keeping.

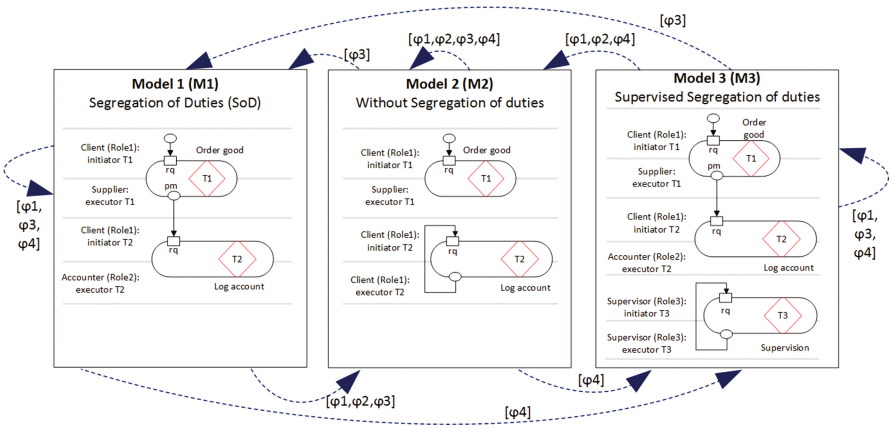

Fig. 3. Mapping the set of possible evolutions (φ) from/to Models 1, 2 and 3: a non-deterministic finite automaton representation.

However, because of the occurrence of non-expected situations, *e.g.* fraud, deception, misunderstanding or misinterpretation, there is a risk of malicious or wrongful change in the inventory scenario.

4.2 EA Evolution Options

We explain how to anticipate the impacts of each decision and consequently to avoid security failures using ex ante calculation. Two possible model transformations (M2 and M3) starting from an initial M1 (see Fig. 3-(Model 1)) are considered. Figure 3-(Model 2) represents one the one hand, a first model design. There exists a risk in changing role's hierarchy when evolving RBAC model in EA. For instance, the inventory unit may be extended with additional tasks (*e.g.* audit). The architect needs to model this change and may misinterpret the role R2 as a responsibility to supervise role R1 orders. In this case, role R2 will go up in the hierarchy and be senior to role R1. In RBAC, this means role R2 will inherit role R1's permissions. This situation may present a fraud risk and is represented in Fig. 3-(Model 2) when U1 is empowered with T1 and T2 concomitantly.

On the other hand, Fig. 3-(Model 3) represents a second model. Here, a new role 3 (R3) is added to the model in order to supervise the orders and the logging transactions that have been executed. M3 is the response to a previous deception successfully attempted, guaranteeing for some time an extra level of operational control. In this context, the role R3 is assigned with the responsibility of initiating and performing the Supervision transaction (T3).

An EA model transformation is triggered whenever the enterprise architect decides to evolve the organization with a known purpose. In this context, the following set of evolutions (φ) decisions is considered: φ1 - do not take any action; φ2 – remove *SoD*; φ3 – add *SoD*; and φ4 – add extra transaction with supervised *SoD*.

The mapping between the known EA models (M) and EA model evolutions (φ) is depicted in Fig. 3 as a non-deterministic finite automaton. The mappings are derived from the enterprise transformation planning produced by the architect. Each model may evolve, whereas different evolution options are available at each moment. Each evolution drives the EA to a new model. The maximum number of possible evolutions is given multiplying φ by M. Figure 3 presents the evolutions that will likely happen. We remark that due to probabilities division, the same φ may drive to more than one M, *e.g.*, φ1 will be simulated with shared probability p and 1 − p to evolve respectively from M1 to M1 and M2. The full probabilities used in the calculation are presented in Table 1 and will be discussed in Sect. 4.2.

Table 1. Transition matrix (P_{ij}^{q}) containing the set of possible evolutions (φ) from/to Models 1, 2 and 3.

φ1	From	To			**φ3**	From	To		
		Model 1	Model 2	Model 3			Model 1	Model 2	Model 3
	Model 1	p	$1-p$	0		Model 1	p	$1-p$	0
	Model 2	0	1	0		Model 2	p	$1-p$	0
	Model 3	0	$1-p$	p		Model 3	p	0	$1-p$
φ2	From	To			**φ4**	From	To		
		Model 1	Model 2	Model 3			Model 1	Model 2	Model 3
	Model 1	0	1	0		Model 1	$1-p$	0	p
	Model 2	0	1	0		Model 2	0	$1-p$	p
	Model 3	0	1	0		Model 3	0	$1-p$	p

4.3 Experimenting Enterprise Architecture Evolution Decision

This section is about the experimental design in DSR. The approach is evaluated and simulated using a linear programming algorithm (Markov Decision Process) to instantiate the theoretical conceptualization introduced in Sects. 3 and 4. The MDP is simulated and the achieved results are argued. Markov Decision Process (MDP) are used make informed design decisions by computing the best EA model alternatives. Alternatives are evolvable models and depend on roles' transformation as depicted in the inventory case study of Fig. 3. This corresponds to the execution of a given type of change operation. Moreover, the decision depends on the dependency between the model and the set of possible evolutions available for checking whether the fulfillment of segregation of duty (*SoD*) constraint is being endangered by the evolution or not.

From the probabilities theory literature, a Markov process is a stochastic process that satisfies the Markov property [25]: if the transition probabilities from any given state depend only on the actual state and not on previous history. Four classes of Markov models are usually distinguished. A Markov chain refers to a process, which has a countable and discrete set of state spaces, but is not controllable. A Markov decision process (MDP) is able to solve the problem of calculating an optimal policy in an accessible and stochastic environment with a known transition model [26].

However, in only partially accessible environments, or whenever the observation does not provide enough information to determine the states or the associated transition probabilities, then the hidden Markov model (HMM) or the partially observable Markov decision process (POMDP) solutions should be considered. The difference is that HMM is applied to uncontrolled systems and POMDP to controlled systems.

In our case study, the models and evolutions are considered as observable, and when an evolution is taken it will be successful. By other words, the system in Fig. 3 is observable and controllable, and therefore, a MDP is chosen to solve the problem of defining the optimal evolutions that maximizes value for the organization.

4.4 Enterprise Architecture Evolution Decision

The goal is to decide if the evolution contains any change that will influence adversely the model. In a real operational environment, many (and concurrent) evolutions are attempted; therefore the enforcement of a continuous process to steer the EA evolutions is demanded. Considering α as the artifact to be the aforementioned roles, then the evolution process is instantiated by the following five steps, and the challenge posed to the architect is to choose the evolution that maximizes the value to the organization:

1. *Observation*: the set of α that are being attempted at operation are observed and collected;
2. *Intelligence*: this step is equal to (1) if a full observation is considered. However, if *(i)* uncertainty about the α exists, or if *(ii)* due to manual task-based environments is not possible to automatically collect α, or if *(iii)* different perceptions coexist within the organization in regard to α; then a partial observation solution should be considered. In the EA context, the different kinds of uncertainty described by Johnson et al. merit further research into Partially observable Markov decision processes (POMDP) [26–28], to estimate the belief α.
   In this case study, however, we merely assume that all the artifacts are observable and employ a Markov decision processes (MDP). MDP evaluates a given EA transformation process maximizing the expected value *(v)* after discounting the decay throughout time. A MDP is usually defined by the tuple (S,A,P,R,γ) where:
   $S = \{S_1,\ldots,S_n\}$ is a set of states, representing all the possible underlying states the process can be in (our case study, the states of S are the models *M1–M3*);
   $A = \{A_1,\ldots,A_n\}$ is a set of actions, representing all the available control choices at each point in time (our case study represents A by the evolutions φ);

$$P_{ij}^{a} = \begin{array}{c} \\ i_0 \\ i_1 \\ \ldots \\ i_k \end{array} \begin{array}{c} \begin{array}{cccc} j_0 & j_1 & \cdots & j_k \end{array} \\ \begin{bmatrix} p_{00} & p_{01} & \cdots & p_{0k} \\ p_{10} & p_{11} & \cdots & p_{1k} \\ \cdots & \cdots & \cdots & \cdots \\ p_{k0} & p_{k1} & \cdots & p_{kk} \end{bmatrix} \end{array}$$

is a transition matrix that contains the probability of a state transition, whereas i is the actual state and j is the final state if a given action a that is being used;[3]
$R = \{R_1,\ldots,R_n\}$ is an immediate reward function, giving the immediate utility for performing an action that drives the system towards each state[4];
Finally, γ is a discounted factor of future rewards, meaning the decay that a given achieved state suffers throughout time.

3. *EA re-design*: in regard to a possible *SoD* violation, the enterprise architect need to re-design a new set of evolution (α,α') pairs, *e.g.*, adding an auditing task such as an extra supervision task. If partial observations are occurring, then the new (α,α') pair depends on the belief α obtained in step (2);
4. *Choose best EA re-design*: a qualitative and/or quantitative valuation of the best evolution to take. This step is the responsibility of enterprise architect. To support the architect the MDP is solved. There are many solutions available to solve the MDP. Our goal is to use MDP using a well-known solution with stable results. Therefore, to obtain the maximized V, we solve the MDP as specified by the following recursive Eq. 1:

$$V(s) := \sum_{s'} P_{\pi(s)}(s,s')\left(R_{\pi(s)}(s,s') + \gamma V(s')\right) \tag{1}$$

where:

$$\pi(s) := arg\ \max_a \left\{ \sum\nolimits_{s'} P_a(s,s')(R_a(s,s') + \gamma V(s')) \right\};$$

5. *Enforce new EA model*: it is equal to result in (4) if a full actuation is considered. Whether operational environment is not completely controllable then α' will be only partial enforced.

Next, the results of an exemplification of this MDP approach is presented to foresee the support that could be delivered to the architect while choosing the best EA evolution.

4.5 Evolvable Enterprise Architecture Results

The obtained results emphasize the rationale behind our approach to deliver valuation when the architect faces different EA model evolution options. In fact, this rationale is

[3] Clearly, how to elicit these probabilities is a key issue for applying the method. Having developed the formalism in this paper, probability elicitation is an important future work to be addressed in industrial case studies. Here, let us just note that it will probably involve a combination of manual analytical methods such as identifying forbidden transitions and setting their probabilities to 0, data-driven methods such as using historical data to find probabilities and expert methods such as surveys (cf. [29] for an example in the EA context) and interviews (cf. [30] for an example in the EA context).

[4] The need to estimate rewards could also be found in literature, e.g., [31] proposes an EA support tool where the score of a given architecture solution should be indicated by the architect. See also [32] for a discussion of utility in EA evaluation.

more important than the particular results obtained for the case study at hand. Moreover, this approach is to be used recursively: observing the reality, simulating different options, enforcing new models and then restarting the loop.

The MDP is computed by a *Matlab*© toolbox[5] using a linear programming algorithm. The transition matrix with the probabilistic estimation between the evolutions (φ) required to transit from a model (M_{actual}) to other (M_{final}) is presented in Table 1. Each cell of Table 1 accounts the previous defined $P^{\varphi}_{M_{actual}M_{final}}$. Let p be the probability of φ be succeed, and for calculation purposes, p is tested in the range [0.1,…,1.0] with small steps of 0.1 each. A positive value is attributed to a cell if and only if an evolution exists in Fig. 3.

Moreover, the reward matrix R, when achieving the desired M_{final}, is defined in Table 2. For all φ, Model 1 and 3 contain an access control model; therefore they have an higher reward. In the specific situation of Model 3, the sum of rewards for all φ is higher because it has a more sophisticated access control model (supervised SoD). Model 2 has zero reward because it should be avoided. Moreover, using the same previous rationale, φ1 (*do not take any action*) has a positive reward, because achieving an access control model without effort is valuable; φ2 (*removing SoD*) has a zero reward because it is not desirable; φ3 (*add SoD*) is the considered as the best commitment cost/benefit for this organization; and finally, φ4 (*add extra transaction with supervised SoD*) is valuable, but, because of implementation effort to enforce a new transaction the reward is less than φ3.

Table 2. Reward matrix (R) containing the set of rewards when achieving a model through each evolution (φ).

Achieved model	Evolution			
	φ1	φ2	φ3	φ4
Model 1	*2*	*0*	*4*	*0*
Model 2	*0*	*0*	*0*	*0*
Model 3	*2*	*0*	*3*	*3*

For example, the probability to be in state *M3* at time $t + 1$, starting in state *M3* and choosing φ1 at time t, is $P(3,3,1) = p$ and the associated reward is $R(3,1) = 2$. Translating for the case study language, this denotes that keeping the model supervised *SoD* after doing no action has a probability p and offers the third higher reward.

Figure 4 depicts the result from the MDP calculation using three distinct representations. In the top left corner, the value function is presented at each stage k, and each p value is separated. We observe that increasing p drives to increased value function. In the top right corner, the elicited evolutions to fulfill the value function are depicted. For each p the set of evolutions differ (each set correspond here to a different color). In the bottom, for each p, we represent the percentage of time spent in each model. Here, we observe that changing p implies change in the percentage of time spent in each model.

[5] Toolbox public available at http://www7.inra.fr/mia/T/MDPtoolbox.

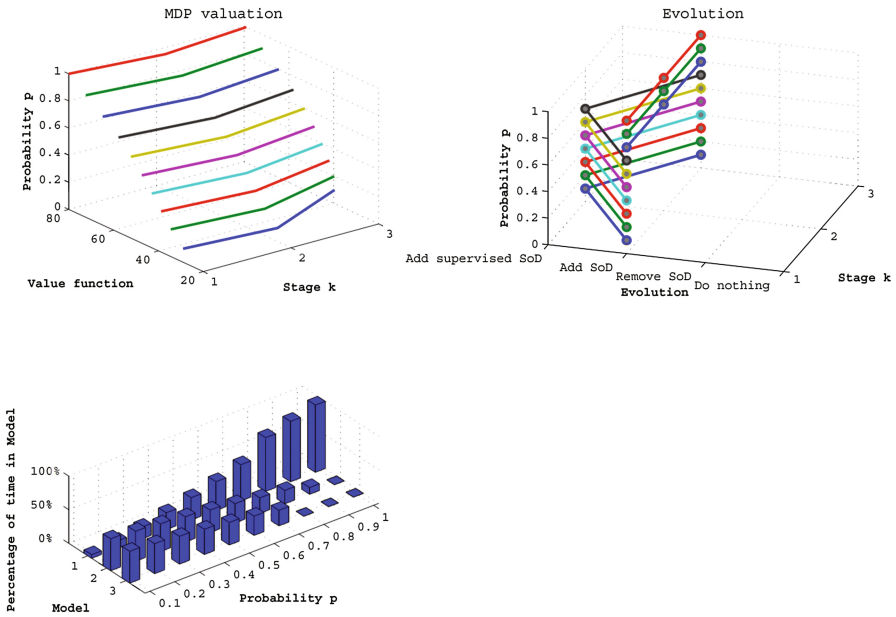

Fig. 4. MDP calculation: P_{ij}^{a} cf. Table 1; R cf. Table 2; γ = 0.95 and p ∈ [0.1,…,1.0:0.1].

Aggregating the previous results, the solution that maximizes the value function, is to always keep M1, however, it demands $p = 1$. The interpretation is straightforward and intuitive – if the organization can be fully controlled with no risk of going astray, then it is easy to maximize value. Unfortunately, due to the occurrence of workarounds, it is not expected that any organizational operation will behave 100 % as prescribed [33, 34]. Moreover, when $p \in [0.8,\ldots,1.0[$ then it is possible to avoid φ4 (add extra transaction with supervised *SoD*) which imposes implementation costs. Yet, if p [0.1,…,0.8] then φ4 is required and extra costs are incurred. Therefore, in this case study, the probability (p) to succeed with an evolution (φ) seems to be the relevant variable to maximize the value function.

Furthermore, following the rigor imposed by DSR methodology, these results are analysed using the four principles of *(i)* abstraction, *(ii)* originality, *(iii)* justification and *(iv)* benefit as proposed by [35]:

1. *Abstraction* (the proposed solution must be applicable to a class of problems) – the solution may be used to evaluate other EA models, considering the fact that MDP evaluation depends on the estimation process that is defined for each reality.
2. *Originality* (the proposed solution must substantially contribute to the advancement of the body of knowledge) – by gathering the related literature combined with a stochastic approach, a novel solution is presented, representing a contribution over and above what has been found in the related work Sect.
3. *Justification* (the proposed solution must be justified in a comprehensible manner and must allow validation) – the presented solution depends on *(i)* EA modeling, then *(ii)* converting EA models into a non-deterministic finite automaton

representation, and finally *(iii)* parameter estimation. The calculation results are repeatable using any MDP computational environment.

4. *Benefit* (the proposed solution must deliver benefit, immediately or in the future for the respective stakeholder groups) – the solution explores the benefits of using stochastic approaches supporting EA architect decisions. This goal can be achieved if engineers are empowered with all pertinent information to forecast the impacts of their decisions in the near future of the organization. With this proposal, the architect is able to simulate different configurations (and evaluate them) before its implementation, and subsequently understand the impact of actions in the operation of the organization.

5 Conclusions

This paper proposes an EA-driven organizational evolution process based on MDP calculations. Our goal is to support EA evolution decisions with an informed decision-making process, and thus enable better-informed transformational changes. We argue that the benefit of having a fully informed decision-making solution is the capability to empower the organizational decisions with tools to forecast the impacts in the near/middle/long -terms for the organization. Subsequently, the organization will be able to decide upon which is the best, and the most timely, action to be enacted.

Our solution is illustrated using a stochastic approach that is grounded in Markov Decision Processes theory. Three distinct EA models are considered and four distinguishable evolutions are available afterward. Therefore, this illustration raises the challenge of choosing between twelve different EA evolution options. In this sense, the challenge is to identify the EA evolution option that maximizes value. We remark that a stochastic approach does not address unknown exception situations; however, it covers a significant part of the reality of how actors behave within their social and human interactions. Moreover, this solution is able to show the valuation throughout the intermediate EA evolution stages. Therefore, the organization is able to forecast not only the final valuation to be achieved, but also the value that will be returned throughout time.

The main weakness of the method presented is, of course, that it has not yet been applied to a real case. Clearly, this constitutes the most important direction for future work, where not least the elicitation of probabilities and the full complexity of EA evolution options will be important challenges to overcome. By finding suitable industry partners, we hope to develop an informed decision-making approach that works, side-by-side, with humans taking dynamic decisions. Some potential alternatives and complements are business intelligence, business analytics, process mining, and event calculus. In such real-world environments, the richness of detailed data available might also call for the use of simulation methods that go beyond the analytic solutions demonstrated here.

References

1. Lankhorst, M.M.: Enterprise Architecture at Work – Modelling, Communication and Analysis. The Enterprise Engineering Series, 4th edn. Springer, Berlin (2013)
2. Greefhorst, D., Proper, H.A.: Architecture Principles – The Cornerstones of Enterprise Architecture. Enterprise Engineering Series. Springer, Heidelberg (2011)
3. Gaaloul, K., Guerreiro, S.: A decision-oriented approach supporting enterprise architecture evolution. In: 2015 IEEE 24th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), pp. 116–121. IEEE (2015a)
4. Aier, S., Gleichauf, B.: Application of enterprise models for engineering enterprise transformation. Enterp. Model. Inf. Syst. Architect. **5**(1), 58–75 (2010)
5. Dietz, J.L.: Enterprise Ontology: Theory and Methodology. Springer, Berlin (2006)
6. Roth, S., Hauder, M., Matthes, F.: A tool for collaborative evolution of enterprise architecture models at runtime. In: 8th International Workshop on Models at Runtime, Miami, USA. IEEE Computer Society (2013)
7. Bernard, S.A.: An Introduction to Enterprise Architecture: 3rd edn. Published by AuthorHouse (2012)
8. TOGAF: The Open Group – TOGAF Version 9. Van Haren Publishing, Zaltbommel (2009)
9. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Q. **28**(1), 75–105 (2004)
10. Winter, R.: Design science research in Europe. Eur. J. Inf. Syst. **17**(5), 470–475 (2008)
11. Johnson, P., Lagerström, R., Närman, P., Simonsson, M.: Enterprise architecture analysis with extended influence diagrams. Inf. Syst. Front. **9**(2–3), 163–180 (2007)
12. Sommestad, T., Ekstedt, M., Johnson, P.: A probabilistic relational model for security risk analysis. Comput. Secur. **29**(6), 659–679 (2010)
13. Närman, P., Buschle, M., König, J., Johnson, P.: Hybrid probabilistic relational models for system quality analysis. In: 2010 14th IEEE International Enterprise Distributed Object Computing Conference (EDOC), pp. 57–66. IEEE, October 2010
14. Johnson, P., Ullberg, J., Buschle, M., Franke, U., Shahzad, K.: An architecture modeling framework for probabilistic prediction. Inf. Syst. e-Bus. Manag. **12**(4), 595–622 (2014)
15. Mikaelian, T., Nightingale, D.J., Rhodes, D.H., Hastings, D.E.: Real options in enterprise architecture: a holistic mapping of mechanisms and types for uncertainty management. IEEE Trans. Eng. Manag. **58**(3), 457–470 (2011)
16. Quartel, D., Engelsman, W., Jonkers, H., Van Sinderen, M.: A goal-oriented requirements modelling language for enterprise architecture. In: IEEE International Enterprise Distributed Object Computing Conference, 2009, EDOC 2009, pp. 3–13. IEEE, September 2009
17. Stuart Rance: ITIL Service Transition. The Stationery Office (2011). ISBN 978-0113313068
18. Gaaloul, K., Guerreiro, S.: A risk-based approach supporting enterprise architecture evolution. In: Ralyté, J., España, S., Pastor, Ó. (eds.) PoEM 2015. LNBIP, vol. 235, pp. 43–56. Springer, Heidelberg (2015). doi:10.1007/978-3-319-25897-3_4
19. Bock, A.: The concepts of decision making: an analysis of classical approaches and avenues for the field of enterprise modeling. In: Ralyté, J., España, S., Pastor, Ó. (eds.) PoEM 2015. LNBIP, vol. 235, pp. 306–321. Springer, Heidelberg (2015). doi:10.1007/978-3-319-25897-3_20
20. Franke, U., Johnson, P., König, J.: An architecture framework for enterprise IT service availability analysis. Softw. Syst. Model. **13**(4), 1417–1445 (2014)
21. Aier, S., Buckl, S., Franke, U., Gleichauf, B., Johnson, P., Närman, P., Schweda C.M., Ullberg, J.: A survival analysis of application life spans based on enterprise architecture models. In: EMISA, pp. 141–154 (2009)

22. Lankhorst, M.M., et al.: Enterprise architecture modelling—the issue of integration. Adv. Eng. Inform. **18**(4), 205–216 (2004)
23. Ferraiolo, D.F., Sandhu, R., Gavrila, S., Kuhn, D.R., Chandramouli, R.: Proposed NIST standard for role-based access control. ACM Trans. Inf. Syst. Secur. **4**(3), 224–274 (2001)
24. Botha, R.A., Eloff, J.H.P.: Separation of duties for access control enforcement in workflow environments. IBM Syst. J. **40**(3), 666–682 (2001)
25. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Artificial Intelligence, 3rd edn. Prentice Hall, Upper Saddle River (2010)
26. Puterman, M.L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley, New York (1994)
27. Guerreiro, S.: Decision-making in partially observable environments. In: 2014 IEEE 16th Conference on Business Informatics (CBI), vol. 1, pp. 159–166 (2014)
28. Guerreiro, S.: Engineering the decision-making process using multiple Markov theories and DEMO. In: Aveiro, D., Pergl, R., Valenta, M. (eds.) EEWC 2015. LNBIP, vol. 211, pp. 19–33. Springer, Heidelberg (2015). doi:10.1007/978-3-319-19297-0_2
29. Franke, U., Johnson, P., König, J., von Würtemberg, L.M.: Availability of enterprise IT systems: an expert-based Bayesian framework. Softw. Qual. J. **20**(2), 369–394 (2012)
30. Närman, P., Holm, H., Johnson, P., König, J., Chenine, M., Ekstedt, M.: Data accuracy assessment using enterprise architecture. Enterp. Inf. Syst. **5**(1), 37–58 (2011)
31. Ameller, D., Franch, X.: Assisting software architects in architectural decision-making using quark. CLEI Electron. J. **17**(3), 2 (2014)
32. Österlind, M., Johnson, P., Karnati, K., Lagerström, R., Välja, M.: Enterprise architecture evaluation using utility theory. In: 17th IEEE International Enterprise Distributed Object Computing Conference Workshops, Vancouver, BC, pp. 347–351 (2013)
33. Guerreiro, S., Tribolet, J.: Conceptualizing enterprise dynamic systems control for run-time business transactions. In: European Conference on Information Systems (ECIS) 2013, paper 5 (2013)
34. Alter, S.: Theory of workarounds. Commun. Assoc. Inf. Syst. **34**(55), 1041–1066 (2014)
35. Österle, H., et al.: Memorandum on design-oriented information systems research. Eur. J. Inf. Syst. **20**(1), 7–10 (2011)