Harmonisation of EAM and Agile Methods

Summary of a Breakout Session at TEAR 2018 Workshop

Workshop Trends in Enterprise Architecture Research (TEAR 2018)
Stockholm (Sweden), 16th October 2018

Introduction

Agile methods for application development are gaining popularity in modern software engineering organisations. There are prominent approaches (like Scrum, eXtreme Programming or Kanban) being implemented in organisations and demonstrate being successful with respect to development time and quality. Nevertheless, those approaches are rather applicable in a limited amount of projects with small teams (typically up to ten programmers). Recent frameworks (like SAFe or LESS) offer management approaches that allow for managing several agile projects but there are still uncertainties for companies with adopting them as they are quite new and not evaluated in a broader sense.

Enterprise Architecture Management (EAM) methods and tools are developed as well as successfully applied since many years. One of the main purposes of EAM is the alignment of software systems with business needs and therefore managing corporate application landscapes which also involves development of new applications. However, today's organisations having implemented EAM are struggling with adopting agile methods. Reasons for this are manifold (cf. Figure 1). The administrative overhead for evaluating and approving changes can screw up the time horizon of agile initiatives. They are typically aiming at finishing a new extension within one to six weeks. The time dimension can be seen as one of the general differences as EAM initiatives are addressing long-term concerns over many years. Different objectives and stakeholders also imply varying participants and skill sets. An excellent programmer might also be a good business analyst or system architect but not used to realising over-arching concerns and enforcing that the application landscape supports a corporate strategy following economic constraints.

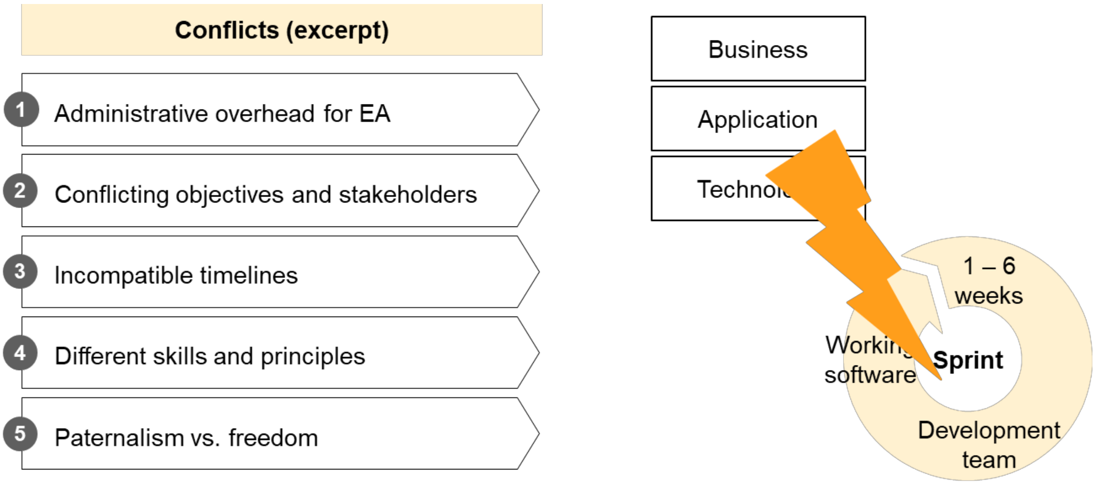

Conflicts (excerpt)
1 Administrative overhead for EA
2 Conflicting objectives and stakeholders
3 Incompatible timelines
4 Different skills and principles
5 Paternalism vs. freedom
Business
Application
Technolo
1 – 6 weeks
Workin software
Sprint
Development team

Figure 1: Potential conflicts between EAM and agile

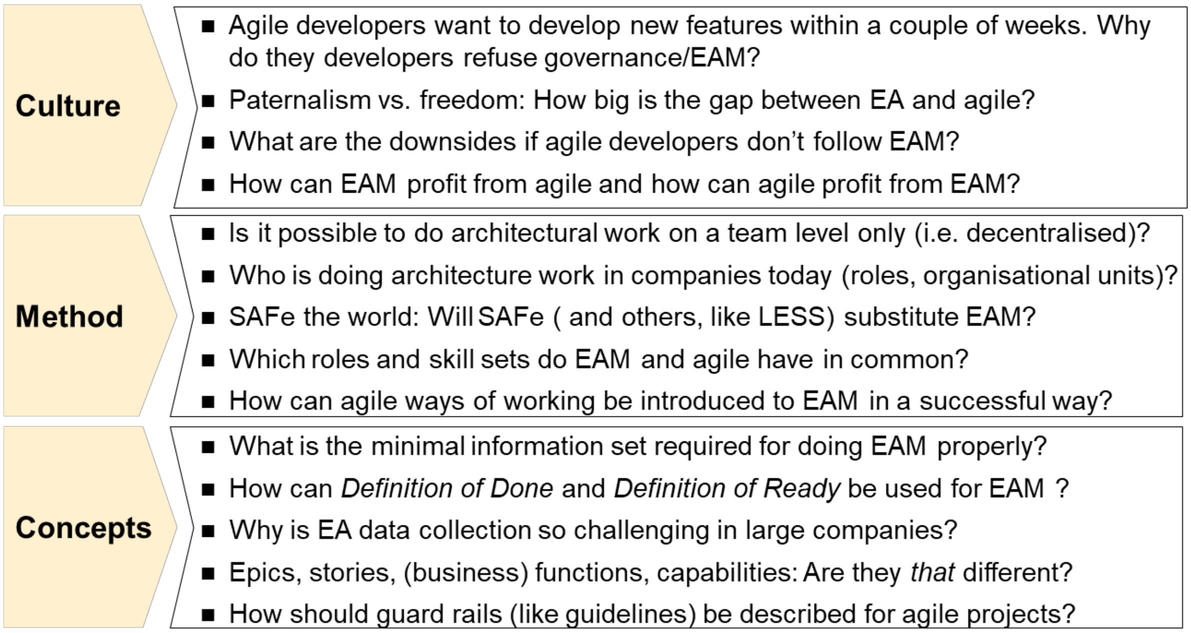
In order to discuss the topic in a broader audience and find potential solutions, a one-hour breakout session has been conducted during TEAR 2018 workshop in Stockholm. The paper at hands provides an overview on the results from discussions within the group. First, the setup of the session will be described in the following section.

Workshop setup

A one-hour session has been conducted during the workshop after a corresponding full paper presentation [1]. The session had the following schedule (plus a ten minutes buffer):

Topic	Duration
Introduction and grouping	10’
Sessions • Group 1: Culture • Group 2: Method • Group 3: Concepts	20’
Summary and discussion	20’

Topic, objective and session format have been introduced to the participants at the beginning. The participants have then been separated into three groups each of which discussed one aspect of EAM and agile harmonisation. Discussions were triggered by some questions addressing typical issues or concerns (Figure 2). These questions were not mentioned to be answered directly by the participants but representing some basic ideas for starting the discussions



Culture

- Agile developers want to develop new features within a couple of weeks. Why do they developers refuse governance/EAM?
- Paternalism vs. freedom: How big is the gap between EA and agile?
- What are the downsides if agile developers don’t follow EAM?
- How can EAM profit from agile and how can agile profit from EAM?

Method

- Is it possible to do architectural work on a team level only (i.e. decentralised)?
- Who is doing architecture work in companies today (roles, organisational units)?
- SAFe the world: Will SAFe (and others, like LESS) substitute EAM?
- Which roles and skill sets do EAM and agile have in common?
- How can agile ways of working be introduced to EAM in a successful way?

Concepts

- What is the minimal information set required for doing EAM properly?
- How can *Definition of Done* and *Definition of Ready* be used for EAM ?
- Why is EA data collection so challenging in large companies?
- Epics, stories, (business) functions, capabilities: Are they *that* different?
- How should guard rails (like guidelines) be described for agile projects?

Figure 2: Example questions for triggering discussions

Results

Even though each group was provided with one of the topics, discussion were not that focussed. It is hard to discuss one topic without having the other in mind. This section will provide an overview on the results of the discussions within each group and the overall discussion wrapping up the breakout sessions.

Can EA work be decentralised?

Working with architecture on a team level can be done if no intersections with other teams exist. The closer teams are connected the more coordination is needed. As architecture is typically close to strategic work it covers long-term decisions and solutions, which should be solved on an organization-wide level. All architectural discussions on a micro level can be made on a team level.

Architecture roles in companies

We often find solution architects at larger organisations, whereas the role of the enterprise architect is not so strongly established. The location within the organization varies. Enterprise Architects are typically located organization-wide, (agile) teams sometimes have their own (solution) architects.

Relevance of agile frameworks

Agile Frameworks can be seen as a complement to EAM. They focus on value delivery and how organizations should be structured. However, they are not so specific and prescriptive on the architecture work. For example topics such as information- or application architecture are not addressed by the agile frameworks. Therefore EAM is still necessary and will coexist with new ways of working.

Information need

There was a discussion on whether agile projects require information from EA or the other way around. It was advocated that it is rather the Enterprise Architect relying on data on progress and results from software development projects as it is needed for documenting and planning the application architecture. The situation was seen similar to corporate accounting. There might be a central charts of accounts as required legal and economic reasons. However, subsidiaries can have their own chart of accounts as they have a different scope, might follow differing objectives or the corporation is just of a mixed form, having divisions for completely different businesses or markets. The standard chart of account is not pushed to the divisions but a corporate reporting is implemented for provided required financial figures—bottom-up.

Incentives

The discussion started with asking for concepts (e.g. models, artefacts, information) for aligning agile projects with EAM. Nevertheless, it should be more relevant focussing on incentivising the participants for doing the right thing instead of forcing them to maintain standardised documents. Example: If re-use is important, people should be encouraged by KPI for re-using instead of just documenting artefacts for re-use.

Roles

It was suggested to think about how existing roles can be extended in order to support the coordination with other agile projects. Programmers already claim also being good business analysts or system architects so that there is no dedicated staff required to provide these skills in an agile project team. In the same way, existing team members can cover for typical EA tasks like for example fostering re-use, overall data integrity, system integration or interface design. Also the Product Owner can be extended so that it supports re-use, alignment on interface design, optimal application landscape as well as adherence to corporate objectives and constraint. This might, surely, put a lot of tasks onto the team which can be hard to manage in real life projects. Nevertheless, it needs to be examined whether EAM always needs to be centralised or can also be distributed over (partially) autonomous teams. Further frameworks life SAFE or LESS should be evaluated with respect to extended roles and coordination mechanisms.

Available information

There were some suggestions for shared information, too:

- *Contact details*: If you do not have sufficient information about an application, system or business, it might be good to know who to ask about it. In fact, this kind of information is crucial as existing information might be ambiguous, incomplete or out-of-date. As not each information need can be anticipated for future uses, contact details will always be required. Popular EAM tools all have a field for this kind of information and it is recommended to have this as a mandatory field.

- Interface specification: One common EAM purpose lies in supporting application integration by maintaining documentation about system interfaces. Such an interface specification lists services exposed by a system and also data types for data exchange. Such a document will be required by agile team as well for connected their application to others by using existing services.
- KPI: Management of a company relies on Key Performance Indicators (KPI) that measure whether or by which extent corporate objectives are achieved. Instead of prescribing how work should be executed, a KPI only focusses on the quality of the outcome and leaves details about how to do it to those responsible for execution. Consequently, instead of standardising documentation required for achieving a specific objective (for example reusability) a KPI might be defined (e.g. a measure for redundancies or reuse). However, KPI need to be very specific as well as measurable and defining them in a way that they drive desired behaviour (and avoiding misuse) is still a challenging task.

Further standard concepts of agile methods (e.g. epic, story, DOD, DOR) as well as EAM (e.g. objective, constraint, business, capability and requirement) have not been discussed in more detail.

Necessity for EAM

It was argued that not all projects needs to be guided (or guarded) by EAM as many have a very limited and clear scope or do not affect (or depend on) other systems. The operation system for e new product or the control software for a novel production machinery has a clear scope and the interaction with surrounding systems is usually straight forward. It might be hard to justify administrative overhead for coordination with other project teams as it also slows down development speed. However, this was put into perspective as it is usually hard to tell whether an application is completely self-contained or has some dependencies that will only get obvious during project execution. A new product also requires customer service support and reusing an existing Customer Relationship Management (CRM) suite should be preferred over implementing complaint management functionality into the product's operating system. Also further analysis of the machine control system might reveal dependencies on existing systems for Master Data Management (MDM), order management or financial applications.

Summary

Participants have been very active within the groups but also during the overall result discussion. This was also reflecting the fact that there is still a broad range of interpretations on agile methods but also how they can be harmonised with existing Governance approaches like EAM. Participants also found the workshop format with given topics and questions as too restrictive as it was rather hampering discussions within the groups. However, the results still reflect a large spectrum of ideas. These ideas do not represent final approaches for integrating agile methods with EAM but are a first collection of ideas that need to be evaluated further.

Participants

- Ulrik Franke, RISE SICS, Stockholm, Sweden (Coordinator)
- Robert Lagerström, KTH Royal Institute of Technology, Sweden (Facilitator)
- Markus Buschle, zeb Consulting, Stockholm, Sweden (Facilitator)
- Jürgen Jung, Frankfurt University of Applied Sciences, Germany (Facilitator)
- Daniel Ritter, University of Vienna
- Kevin Andrews, Ulm University
- Simon Hacks, RWTH Aachen University
- Emmanuel Nowakowski, University of Innsbruck

- Sylvain Hallé, Université du Québec à Chicoutimi
- Eike-Christian Ramcke, HAW Hamburg
- Pontus Johnson, KTH Royal Institute of Technology
- Mathias Ekstedt, KTH Royal Institute of Technology
- Raphael Schilling, University of St. Gallen
- Erik Proper, Luxembourg Institute of Science and Technology
- Adina Aldea, University of Twente
- Ulrike Steffens, Hamburg University of Applied Sciences
- Ömer Uludag, Technische Universität München
- Alexander Jourkovski, KTH Royal Institute of Technology
- Mert Canat, KTH Royal Institute of Technology
- Svetlomir Petrov, KTH Royal Institute of Technology
- Ola Dagberg, Modellum
- Stefan Varga, KTH Royal Institute of Technology

References

[1] M. Canat and et al., “Enterprise Architecture and Agile Development: Friends or Foes?,” in *2018 IEEE 22nd International Enterprise Distributed Object Computing Workshop (EDOCW)*, Stockholm (Sweden), 2018, pp. 176–183.