GSem Ausarbeitung: Überblick EAM

Chancen für agiles Enterprise Architecture Management durch Aufstrebende Technologische Paradigmen wie Continuous Delivery, DevOps und Microservices

Daniel Kirchner

HAW Hamburg, 20099 Hamburg, Germany

Abstract. Es wird eine Auswahl von neuen technologischen Paradigmen betrachtet, bei denen zur Zeit wachsendes Interesse und erste positive Erfahrungen in dem Betrieb und der Entwicklung von Software beobachtet werden. Aus diesen wird versucht einen allgemeinen Trend zu erkennen und es werden Verknüpfungspunkte zu klassischen Frameworks aus dem Enterprise Architecture Management und der IT Governance vorgeschlagen. Die Ergebnisse werden unter den Gesichtspunkten *lean*, *agile* und *collaborative* diskutiert.

1 Motivation

Informationstechnologie hat sich als lebenswichtiger Aspekt in allen Bereichen großer Unternehmen längst etabliert. Selbst in Unternehmen, deren Kerngeschäft nicht in der Informationstechnologie selbst liegt, werden Geschäftsprozesse - bewusst oder unbewusst - in der Architektur und dem Datenaustausch der betriebenen Systeme abgebildet.

Produkte selbst, ihre Entwicklung und ihr Verkauf, der Kontakt zu Kunden und Lieferanten, Supply-Chain, die Verwaltung der Mitarbeiter und Prozesse aus Controlling und Finanzwesen sind nur wenige Beispiele aus einer sehr langen Liste von Bereichen, die inzwischen von Softwareprodukten mehrerer Generationen unterstützt und teilweise automatisiert werden.

Dabei werden nicht nur Daten erzeugt, verteilt und konsumiert, sondern zunehmend wird auch eine intelligente Verarbeitung der eigenen Daten automatisiert. Dabei werden z.B. Aufgaben aus der Planung, Anomalienerkennung und der Vorhersage automatisiert, die vorher von Menschen übernommen wurde. Dabei entstehen wiederum neue Anforderungen an eine systematische Datenerfassung von neuen - und insbesondere auch von bestehenden - Systemen.

Das Problem, die IT-Landschaft eines Unternehmens ständig neuen Bedürfnissen anzupassen und dabei einen möglichst schlanken und stabilen Betrieb sicherzustellen, ist seit Jahrzehnten Gegenstand von wissenschaftlicher und industrieller Forschung.

In den letzten Jahren haben Techniken aus der Softwareentwicklung mit pragmatischen Ansätzen, kurzen Feedbackzyklen und kleinen Teams gute Resultate in der

Umsetzung komplexer Softwareprojekte erzielt und die Aufmerksamkeit des *Enterprise Architecture Managements* erregt [5].

In diesem Aufsatz werden konkrete Möglichkeiten erläutert aufstrebende technologische Paradigmen in eine *Enterprise Architecture* zu integrieren und Verknüpfungen zum *Enterprise Architecture Management* erläutert.

2 Grundlagen des Enterprise Architecture Managements

2.1 Definition und Scope

Nach [3] ergeben sich folgende Arbeitsdefinitionen von *Enterprise Architecture* und *Enterprise Architecture Management*:

Enterprise Architecture (EA) ist eine Repräsentation der Struktur und des Verhaltens der IT-Landschaft eines Unternehmens mit Bezug auf das geschäftliche Umfeld. Dabei stellt sie die momentane und die zukünftige Nutzung von IT im Unternehmen dar und liefert einen Plan zur Erreichung eines zukünftigen Zustands. Dabei bietet sie

- Einsichten in die IT-Nutzung aus Sicht des Geschäftsbetriebs
- eine Vision für die zukünftige Nutzung von IT im Geschäftsbetrieb
- einen Plan für schrittweise Evolution hin zu einem zukünftigen Zustand

Enterprise Architecture Management (EAM) ist ein strukturierter Ansatz um *EA* zu erzeugen, zu verwalten und anzuwenden um die IT-Nutzung am Geschäftsbetrieb auszurichten. Dabei übersetzt *EAM* die geschäftliche Vision in konkrete Unternehmungen und begleitet das Unternehmen vom jeweils aktuellen EA-Zustand bis zu einem Zielzustand.

2.2 Tools und Frameworks

Um einen gemeinsamen Wortschatz der Beteiligten des *EAM* zu schaffen, aber auch um eine Sammlung von bewährten Richtlinien, Prozessen und Dokumenten verfügbar zu machen wurden verschiedenste *EAM*-Frameworks entwickelt.

Zu den wichtigsten (siehe [3]) Frameworks gehören das Zachman Framework[1], das Open Group Architecture Framework (TOGAF)[2] und die Gartner Methodologies[3].

[1] https://www.zachman.com/about-the-zachman-framework, abgerufen am 03.06.2015
[2] https://www.opengroup.org/togaf/, abgerufen am 03.06.2015
[3] http://www.gartner.com/technology/research/methodologies/methodology.jsp, abgerufen am 03.06.2015

3 Übersicht der betrachteten Technologischen Paradigmen

3.1 Continuous Delivery

Continous Delivery ist eine Erweiterung von *Continous Integration*, die als Methodik der Softwareentwicklung bereits erfolgreich in Unternehmen etabliert ist ([6]).

Wesentliches Ziel ist eine schnelle Rückmeldung für eingepflegten Code durch möglichst unmittelbare Auslieferung der Änderung bis in das laufende System zu erreichen. Dabei wird eine Automatisierungslinie (*continuous delivery pipeline*) geschaffen, die Build, Tests und Deployment des geänderten produktiven Systems soweit automatisiert, dass möglichst schnell Werte (im Sinne von produktiv eingesetzten Features) geschaffen werden (Vgl [7]).

3.2 Microservices

Mit dem *Microservices*-Architekturparadigma wird eine Art der Entkopplung von Softwarekomponenten beschrieben, bei denen die Komponenten in eigenen betriebssystemartigen Umgebungen existieren.

Durch die Weiterentwicklung von Technologien zur Erzeugung und Verwaltung solcher Anwendungscontainer, wie *Docker*[4] und *Rocket*[5], ist es möglich die automatische Erzeugung solcher Container - und damit das Deployment von Softwarekomponenten - drastisch zu vereinfachten. Dieses ist ein wesentliches Element bei dem Übergang von *Continuous Delivery* zu *Continuous Deployment* ([7]).

Zudem ist es möglich mit dieser Art des Anwendungsbetriebes Architekturen umzusetzen, die den Wartungsaufwand und die horizontale Skalierbarkeit von Anwendungen deutlich verbessern ([8]).

3.3 DevOps

DevOps bezeichnet ein Paradigma der engen Verknüpfung zwischen Verantwortung für die Entwicklung und dem produktiven Betrieb von Softwarekomponenten.
Bei der Umsetzung einer *Continuous Delivery Pipeline* ist für die Produktionstauglichkeit der Software neben den üblichen Tests auf Korrektheit der Implementation besonders die Performance im realen Betrieb ein kritischer Punkt ([9]).

4 Vorschlag zur integrierten Umsetzung der Paradigmen auf Unternehmensebene

Setzt man die betrachteten technologischen Paradigmen in einer idealisierten Architektur um, könnte sich etwa ein Modell wie in Abb 1 dargestellt ergeben.

[4] https://www.docker.com/, abgerufen am 03.06.2015
[5] https://coreos.com/blog/rocket/, abgerufen am 03.06.2015

Es wird eine vertikale Aufteilung der IT-Landschaft vorgenommen, die fast ausschließlich an den geschäftlichen Kategorien orientiert ist (Ausnahme ist der Betrieb von technischer Infrastruktur). Dabei werden über *Continuous Delivery* möglichst kurze Feedbackwege über den gesamten Stack von den geschäftlichen Anforderungen bis zum Betrieb und der Performance einzelner Dienste im Backend geschaffen.

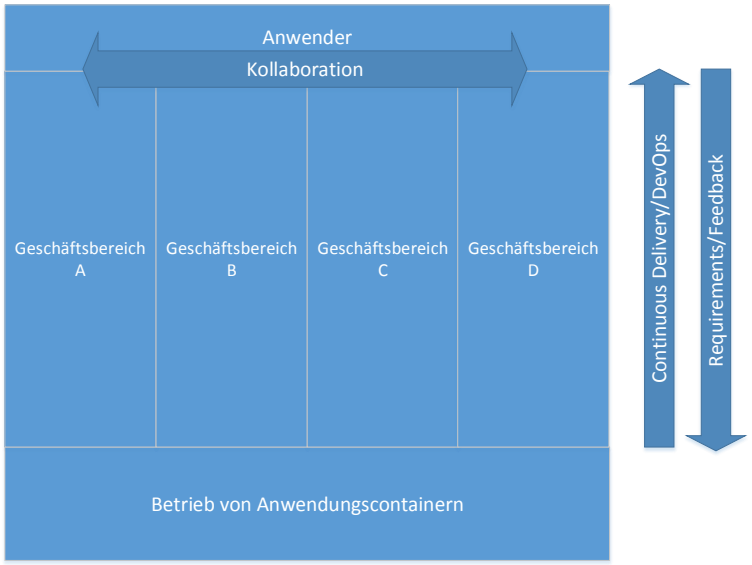

Fig. 1. Naive Sicht auf eine Continuous-Delivery-zentrierte Unternehmensarchitektur

Aus dieser Betrachtung ergibt sich in erster Näherung eine mögliche Architektur für IT-Landschaften von Unternehmen, die sich den agilen Werten[6] annähern könnte:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

[6] http://agilemanifesto.org/, abgerufen am 03.06.2015

5 Zusammenfassung und Ausblick

Es wurde erläutert, dass aus einer sehr oberflächlichen Betrachtung heraus durchaus die Möglichkeit bestehen könnte durch konkrete Einflechtung der genannten technologischen Paradigmen einen erhöhten Reifegrad (*Maturity Level*) bezüglich Agilität und Kollaboration durch und innerhalb von Unternehmens-IT zu erreichen.

Es stellen sich jedoch viele offene Fragen, die in weiterer Literaturrecherche und auch im Experiment behandelt werden müssten, beispielsweise:

- Wie wird eine sinnvolle Aufteilung in Geschäftsbereiche gemessen?
- Welche Metriken existieren?
- Wie groß ist der Einfluss von Duplikation von Komponenten?
- Wie geht das vorgeschlagene Modell mit Korrosion seiner Struktur um?
- Welche Artefakte von üblichen EAM-Frameworks lassen sich noch anwenden, welche müssen angepasst werden?
- Wie könnte eine Transition von einer bestehenden Architektur aussehen?
- Welchen Einfluss hat das Modell auf Sicherheits- und Verfügbarkeitsrisiken?

Das *HAW Labor für Anwendungsintegration* ist dabei ein Bereich, der für die Untersuchung vieler dieser Fragen einen akademisch geschlossenes und dennoch realitätsnahes Umfeld bietet.

Ein konkreter Einstieg könnte sein, im Rahmen eines Grundprojektes die technischen Vorraussetzungen für eine *Continuous Delivery Pipeline* in einem technisch und geschäftlich heterogenen Umfeld zu schaffen.

Durch zweimal jährlich wechselnde Studentengruppen (in der Rolle der Softwareentwickler) und die Abwesenheit des Risikos geschäftsvernichtender, katastrophaler Systemausfälle sind dann die Rahmenbedingungen für weiterführende Untersuchungen und Experimente gegeben.
[**?**] [1]

References

1. Clarke, F., Ekeland, I.: Nonlinear oscillations and boundary-value problems for Hamiltonian systems. Arch. Rat. Mech. Anal. 78, 315–333 (1982)
2. Kim, S., Park, S.: Automated Continuous Integration of Component-Based Software: An Industrial Experience ASE '08 Proceedings of the 2008 23rd IEEE/ACM International Conference on Automated Software Engineering Pages 423-426
3. Bente, S.: Collaborative Enterprise Architecture 2011 Morgan Kaufmann Publ.
4. Haynes, S., Skattebo, A.: Collaborative architecture design and evaluation DIS '06 Proceedings of the 6th conference on Designing Interactive systems Pages 219-228
5. Buckl, S., Matthes, F.: Towards an Agile Design of the Enterprise Architecture Management Function 2011 15th IEEE International Enterprise Distributed Object Computing Conference Workshops Pages 322-329

6. Fitzgerald, F., Stol, K.-J.: Continuous software engineering and beyond: trends and challenges RCoSE 2014 Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering Pages 1-9
7. Wolf, D.: Continuous Delivery: Der pragmatische Einstieg 2014 dpunkt.verlag GmbH
8. Toffetti, G., Brunner, S.: An architecture for self-managing microservices AIMC '15 Proceedings of the 1st International Workshop on Automated Incident Management in Cloud Pages 19-24
9. Waller, J., Ehmke, N.: Including Performance Benchmarks into Continuous Integration to Enable DevOps ACM SIGSOFT Software Engineering Notes archive Volume 40 Issue 2, March 2015 Pages 1-4