International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 6 Issue 03, March-2017

Establishing Architecture for Large Enterprise Solutions in Agile Environment

Sujatha Dantuluri
Software Architecture
Karsun Solutions LLC
Herndon, USA

***Abstract*— Companies are adopting Agile, Scaled Agile Framework (SAFe), Lean to deliver software faster to the market. These stacks of methodologies ensure early delivery and faster time to recover from mistakes. While developing software iteratively we need to ensure design and quality are not compromised to speed. SAFe proposes to use emergent architecture and design which develops iteratively. However, for large, complex systems with multiple system integration points and high volume of transactional data, doing architecture and design iteratively will lead to excessive rework, redundancy and compromise of the architectural principles of scalability, interoperability, and performance. Proper architecture at both Enterprise level and system level is required to develop such large enterprise systems. SAFe proposes Enterprise level Architecture, also termed as intentional architecture, to be used along with iterative Emergent Architecture. Many Enterprise Architecture frameworks are available in today's market. Can they be used for establishing the software architecture for Large Enterprise level software systems in Agile world? Can they be efficiently be applied to system level architecture? This paper explores these questions and establishes a process by which enterprise and system level architecture can be applied to the development of large scale software systems in an Agile environment. The paper uses two enterprise architecture frameworks, Zachman and TOGAF, to define Architecture while adopting SAFe Methodology of software development.**

Keywords*— *Agile; Scaled Agile Framework(SAFe); Fragile; Architecture Runway; Emergent Design; Intentional Architecture; Software Development; Architecture; Zachman; TOGAF;

I. INTRODUCTION

Software Architecture is very important to ensure software is developed with quality and results can be repeated. It ensures systems developed, are scalable, interoperable, and manageable and guarantees performance. In order to establish an architecture for a system, we need to know the purpose and scope of the application, its importance and relationship to the overall strategic goals of the organization based on known requirements of the system being developed. However in an Agile environment where the requirements are groomed iteratively and change is constant, establishing architecture and design would be a daunting task. Especially if the system being developed is large enterprise level solution, iterative architecture and design could result in a lot of rework and redundancy. If there are many integration points and functionality spans across multiple applications, defining enterprise level architecture helps the software being developed to align with strategic goals of the enterprise, helping business capabilities to be shared across organizations. But establishing such architecture in an Agile environment would be challenging. This paper addresses this issue and establishes a process for architecture development for enterprise level applications. The paper defines a process to develop a software solution which spans across multiple legacy applications using Agile principles'. The approach explained in the paper has been adopted for the development of software which modernizes legacy data capture and is intended to establish a framework for enterprise level services in the future. The solution spans across 6 legacy software systems. By adopting Scaled Agile Framework we delivered the solution in an iterative manner and incorporated feedback easily. Given the strategic nature of the software being developed and its complexity, it was very important to develop the software to align with the enterprise architecture of the organization. SAFe proposes an iterative emergent architecture and design which evolves with the system. At the same time it proposes intentional Architecture to be defined for large scale systems. This paper discusses the approach used to develop both enterprise level architecture and system level architecture using Enterprise Architecture Frameworks. These frameworks were originally designed to work with traditional waterfall development, where requirements are elicited, software architecture is defined, design is completed and then the implementation is done. However, these frameworks can be tailored to deliver architecture iteratively as part of an Agile software development. This paper explains the efficient use of enterprise architecture frameworks in an agile environment. Zachman and TOGAF, two preeminent enterprise Architecture frameworks were used to demonstrate such use of Enterprise Architecture Framework in Agile Architecture development

II. DEFINITIONS

A. *Agile*

"Agile Software Development is an umbrella term for a set of methods and practices based on the values and principles expressed in the Agile Manifesto. Solutions evolve through collaboration between self-organizing, cross-functional teams utilizing the appropriate practices for their context" (Agile Alliance, 2015).

Scrum, Kanban, XP are some of the methodologies defined as part of Agile.

SAFe, is a framework which provides comprehensive guidance for achieving the benefits of Lean-Agile development at enterprise scale (A Scaled Agile, Inc. White Paper, July, 2016). It proposes that by balancing upfront intentional Architecture with iterative emergent architecture, also termed as Agile Architecture, we can build complex enterprise solutions.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 6 Issue 03, March-2017

B. *Enterprise Architecture*

"An architecture is defined as system fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution" (ISO/IEC/IEEE 42010, 2011)

It is important to have architecture defined for an organization so that we can produce repeatable quality products and to ensure we meet the scalability, interoperability, security and compliance factors derived from non-functional requirements of the project.

III. INTRODUCTION TO AGILE PROJECT IMPLEMENTED

The project on which the following methodology was implemented is a federal government project, which allows vendors to submit the products/services and their pricing, against solicitations in the form of an offer. Once the offer is submitted, it is reviewed and awarded by Contracting officer (CO). Offer now becomes a contract on which vendor can submit modifications. These modifications are reviewed and approved by Contracting Officer. The applications to submit an offer and for CO to review and approve, already exists. The project is allowing capture of formatted product data, which was captured in the form of documents. The data is captured in the form excel uploads, form entry and Electronic Data Transmission (EDI). Infrastructure for EDI already exists and is not part of the scope. Since data captured is large and is transactional in nature, the important architectural concerns for the project are the performance, data consistency and scalability. Along with these, other architecture attributes required for the project are security, integrity of the system, availability and user friendliness. This is a public internet facing application available 24/7, so the above mentioned attributes are very important.

The project is implemented as an Agile project spanning across a year. Since the project caters to large data and is highly transactional in nature, we may need to establish some upfront architecture and since the requirements are groomed iteratively, it was important to establish a framework which that works for the Agile environment.

Since the project is delivered using agile not all requirements are available right away to establish the proper architecture. This project has many integration points with other legacy systems and these cannot be defined unless some level of architecture is defined upfront.

IV. SAFE AND ENTERPRISE LEVEL SOFTWARE SOLUTION

For large scale enterprise level strategic solutions, we need both iterative system level architecture as well as an enterprise level upfront architecture initiative which is enhanced in parallel to system level architecture. SAFe proposes establishing Architecture Runway, which is a set of purposeful, planned architectural initiatives that enhance solution design, performance, and usability and provide guidance to cross team design and implementation synchronization (Scaled Agile, Inc, 2016). These architectural goals also known as enablers in SAFe terminology can be achieved using Enterprise Architecture Frameworks. It also proposes that, for the project success, right amount of Architecture runway is required. At the same time, Architectural Runway needs to evolve along with the project. Establishing Architecture runway and maintaining is done using TOGAF and Zachman is used to derive various architecture artifacts. Agile proposes minimum documentation and hence all minimal architecture artifacts were created so that the necessary upfront architecture is ready for implementation. Let's first understand how Zachman was used to define architecture artifacts and then we will understand the process followed to derive those artifacts. The approach is adopted for developing a solution for the above mentioned project.

V. ZACHMAN FRAMEWORK FOR ARCHITECTURE ARTIFACTS

In the Agile, user stories define the user/system requirements. Bigger user stories are classified as epics and they are divided into multiple smaller user stories. A set of user stories/epics make a feature of a system. They are prioritized and groomed as per end-user needs. In order to design optimized software, teams must look ahead and analyze possible impacts to the system. (Software Engineering Institute, 2017).

According to John Zachman, the author of Zachman Enterprise Architecture framework, by answering six questions Who, What, When, Where, Why and How we can derive answers to any other question about the subject (or object) being described that anybody can construct (Zachman, The Zachman Framework: A Primer for Enterprise Engineering and Manufacturing, 2003)

In Agile the requirements of the software system are represented using user stories. They depict the requirement from the user perspective and describe the features or functionality user wants to achieve. A user story is usually written in the format below

As a <type of user>, I want <some goal> so that <some reason>. (Mountain Goat Software, 1998-2017)

Apparently, a user story represents "who", "what" and "why" of the system, and can help define Architecture of the system.

Enterprise Architecture, as per Zachman is a two dimensional classification scheme, perspectives, and abstractions, and it classifies the design artifacts (descriptive representations, the product descriptions, and the engineering documentation) of an Enterprise (Zachman, The Zachman Framework: A Primer for Enterprise Engineering and Manufacturing, 2003)

The perspective classifies the artifact by the audience, providing different architecture views and the abstraction classifies the content or subject focus of the artifact providing different viewpoints. The principal perspectives defined by Zachman framework are- The Owner's Perspective, the end user perspective, Designer's Perspective, the engineer, the Architect perspective, providing what is technically possible based on owner's perspective and Builder's Perspective, the Engineer producing the end product (Zachman, The Zachman Framework: A Primer for Enterprise Engineering and Manufacturing, 2003).

From the user stories we can derive owner's perspective and based on them and known available requirements, we can derive the designer's perspective and builder's perspective.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 6 Issue 03, March-2017

A. User Story and Zachman Framework mapping

The following method has been used to derive various artifacts of Zachman. In an Agile environment, since it is difficult to establish the Architecture first, and then design, this method of deriving artifacts was used along with Togaf to iteratively enhance the artifacts. The process is explained in the next section. This section only provides the mapping.

- The "goal" of a user story or system and project vision was used to determine abstractions related to "what". The artifacts related to these abstractions are business entities from business owner's perspective which eventually get transformed into data and system entities from the Architect's and builder's perspective. Data entities were defined and ER models were derived based on the goal of the system. Most of the data elements to be captured were available at the beginning of the project and some rules were also available and hence determining the data models and ER models during the first two iterations helped establishing initial data layer
- User system "goals" usually can be used to determine the business connections and hence were used to derive the technology connections related to "Where" abstraction. Vendor can submit formatted data. Existing vendor facing system, which is used for electronic submission of offer was enhanced to meet the requirement. The document should be archived for auditing purposes, so Document repository is used and this document needs to be reviewed and hence the internal CO review system is enhanced for review of the document. This helped determine business and system connections. Various stories related to the epics also help establish these connections.
- Process flows and functional/nonfunctional specifications can be derived based on various stories related to the epics and features which specify the stories required for the development of the feature and they depict the abstraction- "How". High level epics were used to derive business flows, evaluating the inputs and outputs for those flows. These models were eventually used to derive the system input/output and technology input/output. . One of the epic and its goal, available at the start of the project, was to support the upload of the files which has more than 600k records. This helped determine the nonfunctional requirements and establish technology and system input/outputs.
- "Users" in the user story determine the people and their workflow models eventually help determine system roles and technology roles. These artifacts provide the "Who" abstraction. Zachman suggests User's perspective as Enterprise perspective however, we are using "User" of the Agile to help determine system roles. They help establish business roles and thereby help determine the system/technology roles.
- The "reasons" of user story provide the "Why" abstraction and help derive artifacts related to this abstraction. To continue with the example given earlier. The reason auditing purposes helps determine the system means and technology means and we can derive that we need a document repository.

The below table shows mapping between various part of user story and Zachman Artifacts

Classification with Agile relevance/ Audience perspective	Goal of a user story (What)	Goals of system (Where)	User stories in an epic (How)	Users in user story (Who)	Reasons of User Story (Why)
Business Owner Perspecitve	Business entities	Business Connections	Business Transform/ Input/ Output	Business Roles	Business Means
Architect's Perspective	System/Data Entities	System Connections	System Input/Output	System and technology roles	System Means
Builder perspective	Entity Relations	Technology Connections	Technology/ Input/Output	Technology roles	Technology Means

VI. TOGAF AND ARCHITECTURE RUNWAY DEVELOPMENT PROCESS

Establishing a process for developing architecture is very important. Architecture should be developed by the Value team in parallel with implementation of features, carried out by Agile release teams. However we need to establish some upfront/intentional architecture before starting implementation. Architecture Runway developed as a result should comply with EA principles and should be constantly evaluated and maintained. As mentioned earlier, architecture is developed following TOGAF.

As defined in the TOGAF 9.1 documentation, TOGAF consists of three main parts:

1. The TOGAF Architecture Development Method (ADM), which explains how to derive an organization-specific enterprise architecture that addresses business requirements.

2. The Enterprise Continuum, which is a "virtual repository" of all the software architecture assets - models, patterns, architecture descriptions, etc.

3. The TOGAF Resource Base, which is a set of resources - guidelines, templates, background information, etc. - to help the architect in the use of the ADM (The Open Group, 2011).

TOGAF Architecture Development method, ADM is iterative and the below process explains how ADM can be used in Agile software development . Different phases of TOGAF help software architecture to evolve along with the project. We divided the phases to help us with the use of architecture in our Iteration development.

The ADM consists of Preliminary phase and 8 iterative phases. It also has requirements phase, which is shared across all 8 phases.

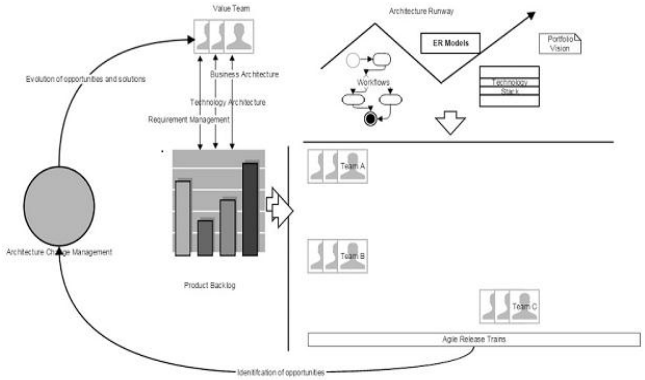
Project Vision and Scope should be identified during the project definition and planning phase. If an organization has an Enterprise Architecture, EA, already established, it should feed into architecture scope and vision for the solution being developed. In addition, project resources, their roles, and responsibilities are also identified during this phase. At this point a Value team consisting of Architects and Business Analysts, is defined. Architecture vision should comply with the strategic plan of the organization. The strategic goal of the

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 6 Issue 03, March-2017

project is to create a database for the products and services offered to the agency and eventually use this database for price analytics. Therefore it was decided to use service oriented architecture. As the project requires lots of upfront architecture to be defined, Value Team consisting of two Architects and two BA's was established during the initiation phase. Value Team works with the product owner and affected teams to groom the backlog and define the Architecture based on the backlog. The Value team would work as part of scrum of scrums. Furthermore, they were to ensure cross team communication is established. In addition, to the vision and scope, we should also define the software architecture development process during the initiation phase. The application supposed to support big volumes of transactional data and hence our emphasis was on the performance and scalability. Four V's volume, velocity, variety and veracity are very important. This helped establish the vision and scope of the Architecture.

Iteration 0 and Iteration 1 were termed as architecture sprints, where value team was grooming the initial set of requirements and establishing architecture. Since the enterprise already had EA, the artifacts developed for enterprise level were utilized while establishing the solution. The solution needs to consider the fact this database will eventually become the central repository for the enterprise and hence the services provided should cater various consumers of the organization, so service oriented architecture was adopted. These two iterations helped establish an architecture roadmap. During architecture iterations, a very high level roadmap for the project is defined and user stories for next iterations are prioritized and groomed. . During this period some of the crucial epics were groomed defining what of the system. Thereby establishing the business models and system models. The goals of the system were also used to define the business and system connections. Initial set of Business Roles and System Roles were also defined during this time At this point, business architecture should be established for next few iterations and owner's perspective for the what, how, who and where should be defined and their respective abstractions should be defined. The Value team working on architecture iterations should help define the designer's perspective for the same. ER and data models were established during the first iterations. System interatctions were defined based on the critical epics of the system. The "reasons" of the user story helped with the determination of technologies to be used and Tools were evaluated based on the project needs and organizational goals. Proof of concept was executed to evaluate the tools and technology required for achieving the goal of uploading huge data in the form of excel or EDI. Reuse is an important attribute considered while laying down the architecture. Agile proposes minimum documentation hence minimum and essential documentation required for iteration team is developed. Information system architecture and technical architecture development help define the abstractions for the designer perspective. Furthermore, Architecture change management should be established in this phase.

By the end of first two iteration some architecture artifacts were developed for release teams to start implementing and Architecture Runway has been established. The image below depicts the process of creating Architecture Runway using TOGAF



Epics business models, process flow and functional specifications were defined depicting the various integration points. The Value team works ahead of the development iterations grooming the functional and nonfunctional requirements of the project's next iterations. From next N iterations, a development cycle is established with Agile Release teams headed by scrum master. The scrum Master or development lead in some cases worked with the Value team to define technical abstractions for the Builder's perspective. During the planning meeting of the iteration, the value team explains the business requirements, business architecture, and system architecture. Scrum team should discuss any factors which could lead to software architecture change and they will be incorporated as part of architecture change management. During iteration development the Agile Release teams (builders) build from architecture and design specification. The Value team would work on next iteration user stories. Development Team Lead /Scrum Master would work closely with the Value team to help implement the architecture artifacts. He/She should help build the builder perspective of abstractions.

In the initial stages, only 2 release teams were established and as the architecture runway matured, Team C is added. During iteration implementation builder artifacts are developed. If any opportunities are identified, then they are communicated to the value team and architecture goes through the change management and it evolves accordingly.

In this model, the value team works in parallel with iteration team to develop the necessary upfront architecture for the next iteration and ensures that the software architecture properly aligns with principles of organization/project and enhancing the architecture roadmap. Thus the Zachman framework was efficiently used for deriving both Enterprise architecture and system architecture level artifacts. On the other hand TOGAF established the process to define these artifacts in a systematic manner

VII. CONCLUSION

Architecture is very important to ensure a repeated quality of software is produced and incorporating the Zachman and Togaf framework in the model above, we can ensure Software Architecture is properly defined even in an Agile environment. The iterative model proposed have helped develop architecture runway, which evolves with every iteration, ensuring the quality of the solution. By developing the right amount of upfront architecture, also termed as intentional architecture and developed during the first few iterations of the project, we could reduce rework and redundancy eliminating technical debt. By

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 6 Issue 03, March-2017

and large, TOGAF ADM being iterative in nature fits well with an Agile development methodology and helps with the iterative evolution of architecture. Moreover Zachman framework efficiently fits with Agile development in defining the architecture artifacts. The Architecture developed in this way, will therefore be in alignment with the strategic plan of the organization. In conclusion, using SAFe and Zachman and TOGAF frameworks together, architecture can be efficiently be developed for large scale enterprise solutions.

A. Further Reading

- http://www.scaledagileframework.com/introduction-to-safe/
- https://www.sei.cmu.edu/architecture/research/agile-architecting/
- http://www.slideshare.net/dannygreefhorst/agile-togaf-and-enterprise-architecture-will-they-blend
- http://pubs.opengroup.org/architecture/togaf9-doc/arch/
- https://www.zachman.com/resources/ea-articles-reference/327-the-framework-for-enterprise-architecture-background-description-and-utility-by-john-a-zachman

REFERENCES

[1] SAFe® 4.0 Introduction Overview of the Scaled Agile Framework® for Lean Software and Systems Engineering, 1st ed. A Scaled Agile, Inc., 2016..

[2] "Agile Manifesto for Software Development | Agile Alliance", Agile Alliance, 2001. [Online]. Available: https://www.agilealliance.org/agile101/the-agile-manifesto. [Accessed: 15- Mar- 2017].

[3] "What is Agile Software Development?", Agile Alliance, 2015. [Online]. Available: https://www.agilealliance.org/agile101/. [Accessed: 15- Mar- 2017].

[4] C. Bedell, "Where Agile and enterprise architecture collide", TechTarget, 2013. [Online]. Available: http://searchsoa.techtarget.com/feature/Where-Agile-and-enterprise-architecture-collide. [Accessed: 15- Mar- 2017].

[5] Systems and software engineering — Architecture description, ISO/IEC/IEEE 42010, 2011

[6] M. Cohn, "User Stories and User Story Examples by Mike Cohn", Mountain Goat Software, 2017. [Online]. Available: https://www.mountaingoatsoftware.com/agile/user-stories. [Accessed: 15- Mar- 2017].

[7] "Software Architecture | Research | Agile Architecting", Sei.cmu.edu, 2017. [Online]. Available: http://www.sei.cmu.edu/architecture/research/agile-architecting/. [Accessed: 15- Mar- 2017].

[8] "TOGAF® 9.1", The Open Group, 2011. [Online]. Available: http://pubs.opengroup.org/architecture/togaf9-doc/arch/index.html. [Accessed: 15- Mar- 2017].

[9] Zachman, J. A. (2001). The Zachman Framework for Enterprise Architecture: A Primer for Enterprise Engineering and Manufacturing, Zachman International. (Out of Print)

[10] Zachman, J. A. (2016). The Framework for Enterprise Architecture: Background, Description and Utility. [Online]. Available www.zachman.com: https://www.zachman.com/resources/ea-articles-reference/327-the-framework-for-enterprise-architecture-background-description-and-utility-by-john-a-zachman