

Distributed Decentralized Microservice Development:
A Distributed Model for a Decentralized System

by
Zachary M. Dall

Submitted in partial fulfillment
of the requirements for the degree of
Doctor of Professional Studies
in Computing

at

Seidenberg School of Computer Science and Information Systems

Pace University

October 2019

ProQuest Number:28026975

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent on the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



ProQuest 28026975

Published by ProQuest LLC (2020). Copyright of the Dissertation is held by the Author.

All Rights Reserved.

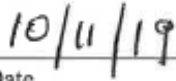
This work is protected against unauthorized copying under Title 17, United States Code
Microform Edition © ProQuest LLC.

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

We hereby certify that this dissertation, submitted by Zachary M. Dall, satisfies the dissertation requirements for the degree of *Doctor of Professional Studies in Computing*



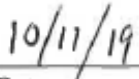
Dr. Tilak Agerwala
Chairperson of Dissertation Committee



Date




Dr. Charles Tappert
Dissertation Committee Member



Date



Dr. Robert Nardelli
Dissertation Committee Member



Date

Seidenberg School of Computer Science and Information Systems
Pace University 2019

Abstract

Distributed Decentralized Microservice Development:

A Distributed Model for a Decentralized System

by

Zachary M. Dall

Submitted in partial fulfillment
of the requirements for the degree of
Doctor of Professional Studies
in Computing

October 2019

Microservice is a distinct architecture that exhibits a high degree of independence regarding the decentralization of development, operation, and teamwork. Decentralization is where different teams, processes, and procedures that are fragmented to promote quick, agile development. Many companies and development groups are moving away from building sizeable monolithic software projects and using the agility of microservices to decouple services and deliver fast solutions. Even with the advantages of fast, agile deployment and the decoupling extensive monolithic services, microservices architecture encompasses more than the decomposition of applications. If the organization and cultural impacts are not addressed initially and continually, the output of a microservices build may not match the desired goal. This dissertation analyzes the need for sharing or distribution of team structures, standardizations, and governance within a microservice development project. The study shows, with the addition of tools and or models utilized anywhere within a development System Development Lifecycle (SDLC) process, decentralized teams are more prepared to mitigate risks and distribute findings and governance between the teams. Last, the study shows, with the additions of models and tools, used by the microservice decentralized development teams, that they are more comfortable gathering data to help streamline their development process, making for a stronger cohesive build.

Acknowledgments

I would like first to thank my doctoral advisor Dr. Tilak Agerwala. Dr. Agerwala took time out of his busy life and always brought his passion and his belief in higher learning. Also, Dr. Agerwala's drive and knowledge helped push and motivate me to think outside the box as well as pursue my passions outside school.

Second, I would like to thank Dr. Tapper for all the help, push, and motivation that he provided over the last 3 ½ years.

I would also like to personally thank one classmate who provided me the biggest lesson of all: "To believe in myself and not overthink." Thank you to a very talented Dr. Robert Nardelli (2018 Pace DPS graduate) who has been a great classmate, an excellent conference co-author, and a partner in crime for four years. I will never forget his effect on me, and it is incredible to call him now a life-long friend.

Third, is to my family and friends. I believe I drove everyone crazy during this process, and I have no words to explain my thanks and gratitude. It was tough to juggle valuable time away from family and friends to continue to work and complete something I have always wanted to accomplish.

Last, thank you to Kristen Stolle for being my rock and my sounding board, thought out the years.

In conclusion, I will NOT forget what this journey and what the Pace University staff have provided me. I'm very honored to call Pace University my alma mater.

Abstract.....	iii
List of Tables.....	x
List of Figures.....	xi
Chapter 1 Introductory to the Study.....	1
1.1 Introduction.....	1
1.2 Research Problem.....	3
1.3 Objective.....	4
1.4 Research Question(s) / Hypothesis:	5
1.5 Scope	6
1.6 Limitations.....	7
Chapter 2 Background on Service-Oriented Architecture and Microservices.....	8
2.1 Introduction.....	8
2.2 Service-Oriented Architecture	9
2.3 Microservices	10
2.4 Development.....	12
2.4.1 Why SOA development can cause problems.....	13
2.4.2 Reason to utilize Microservices.....	14
2.5 Decentralization and Distributed.....	16
2.6 Decentralization Organization and Use of Tools.....	20
2.7 Conclusion	22
Chapter 3 Initial Survey.....	23
3.1 Introduction.....	23
3.1.1 Reason for Initial Survey	23
3.1.2 Companies utilized within the survey	24
3.1.3 Survey Short description.....	24
3.1.4 Questions listed from Survey	25
3.1.5 Outcome of Survey.....	30

3.2 Outcome of Survey	48
3.2.1 Governance Standards	49
3.2.2 API Ownership.....	50
3.2.3 Security Rigger:	51
Chapter 4 Research.....	52
4.1 Chapter Introduction.....	52
4.1.1 Research and Setup.....	52
4.1.2 Experimental procedures.....	53
4.1.3 Learning Tools and Model overview	54
4.2 Tools / Models built.....	55
4.2.1 Overview of Tools built.....	55
4.2.2 Team Readiness Distributed Security Standard Matrix Tool.....	55
4.2.3 Distributed Security Standard Matrix Tool.....	56
4.2.4 CDAD (Characterize, Diagnose, Anticipate, Distribute) Learning Model	56
4.2.5 Dev-CDAD-Prod Model.....	56
4.3 Sub-models.....	57
4.3.1 Mind map and Standards	57
4.3.2 Scatter Charts.....	57
4.3.3 Heat Maps.....	58
4.4 Data Analytics.....	58
4.4.1 Microsoft Excel.....	59
4.4.2 Tableau Software	59
Chapter 5 Metrix and Learning Models.....	60
5.1 Introduction.....	60
5.2 Agile Readiness Risk Grid	60
5.2.1 Purpose.....	60
5.2.2 Overview.....	60

5.2.3 Instructions	61
5.2.4 Agile Readiness Risk Grid Instructions for Microservice	62
5.2.5 Team Agile Risk Grid Worksheet	62
5.3 Data Analytics – Radar Map.....	72
5.3.1 Radar Map – Agile Readiness Risk Grid	72
5.3.2 Radar Map How to read.....	72
5.3.4 Radar Map - Output.....	73
5.3.5 Radar Map – End Goal.....	74
5.3.6 Agile Readiness Scatter Charts	75
5.4 Tableau – Decentralized Team Overview.....	76
5.4.1 Introduction	76
5.4.2 Dimensionality.....	76
5.4.3 Filters/Marks.....	77
5.4.4 Distributed Team Agility Readiness Map	78
5.4.5 Agile Readiness Risk Grid – Conclusion	80
5.5 Distributed Security Standard Matrix Tool.....	80
5.5.1 Purpose.....	80
5.5.2 Overview	80
5.5.3 Instructions	81
5.5.4 Security Standard Matrix Tool.....	82
5.5.4.1 Security Standard Matrix Tool Worksheet	82
5.6 Data Analytics – Radar Map.....	91
5.6.1 Radar Map – Distributed Security Standard Matrix Tool.....	91
5.6.2 Radar Map How to read.....	91
5.6.3 Radar Map - Output.....	93
5.6.4 Radar Map – End Goal.....	93
5.7 Data Analytics – Risk Equivalency Metrix.....	94

5.8 Tableau – Decentralized Team Overview.....	96
5.8.1 Introduction	96
5.8.2 Agile Usage	96
5.8.3 Distributed Security Standard Matrix Reports - Tableau Reports/Dashboards.....	96
5.8.4 Dimensionality.....	99
5.8.5 Rating and Description of Risk Dashboard	101
5.8.6 Conclusion - Rating and Description Dashboard.....	103
5.9 Average Security Risk per Sprint Report.....	103
5.9.1 Introduction	103
5.9.2 Dimensionality.....	103
5.10 Team – Overall Output Risk Assessment Reports.....	107
5.10.1 Introduction.....	107
Chapter 6 Conceptual Models.....	111
6.1 Chapter Introductions	111
6.2 CDAD (<i>Characterize, Diagnose, Anticipate, Distribute</i>) Learning Model	111
6.2.1 Introduction	111
6.2.2 Conclusion.....	113
6.3 Dev-CDAD-Prod Model	114
6.3.1 Introduction	114
6.3.2 CDAC Overview.....	115
Chapter 7 Final Survey	117
7.1 Introduction.....	117
7.2 Exit Survey Questions and Answers	117
7.2.1 Exit Survey	117
7.2.2 Exit Survey Question #2 - Agile Readiness Risk Grid Survey Exit.....	119
7.2.3 Exit Interview – Feedback	122
Chapter 8 Conclusion and Future work.....	126

8.1 Conclusion	126
8.2 Future Work	128
8.2.1 Manual Utilization.....	128
8.2.2 Digitalize Utilization	128
Appendix A Glossary	129
References	133

List of Tables

Table 1 Differences between SOA and Microservices.....	12
Table 2: (Company A) Survey Question #4.....	35
Table 3: (Company B) Survey Question #4.....	36
Table 4 (Company A) Survey Question #8.....	42
Table 5 (Company B) Survey Question #8.....	43
Table 6: (Company A) Survey Question #9.....	44
Table 7: (Company B) Survey Question #9.....	44
Table 8: (Company A) Survey Question #10.....	45
Table 9: (Company B) Survey Question #10.....	46
Table 10: Illustration: Agile Groups and Agile Areas.....	63
Table 11: Agile Readiness Levels.....	64
Table 12: Security Group and Standards.....	83
Table 13: Security Standard Levels.....	85
Table 14: Final Survey Participants and Roles.....	117
Table 15: Final Survey: Likert Scale Question #1.....	118
Table 16: Distributed Security Standard Matrix Tool Survey Exit.....	118
Table 17: Final Survey: Likert Scale Question #2.....	119
Table 18: Agile Readiness Risk Grid Survey Exit.....	119
Table 19: Final Survey: Likert Scale Question #3.....	120
Table 20: CDAD Usage Survey Exit.....	120
Table 21: Final Survey: Likert Scale Question #4.....	121
Table 22: DEV-CDAD-Prod Usage Survey Exit.....	121
Table 23: Final Survey: Likert Scale Question #5.....	122
Table 24: Overall Research - Exit Survey.....	122

List of Figures

Figure 2-1: SOA build of a Monolith	13
Figure 2-2 Challenges of monolithic architecture.....	14
Figure 2-3 SOA vs. Microservice Architecture	15
Figure 2-4: Centralized / Decentralized and Distributed.....	17
Figure 3-1: (Company A) Survey Question #1.....	31
Figure 3-2: (Company B) Survey Question #1.....	32
Figure 3-3: (Company A) Survey Question #2.....	33
Figure 3-4: (Company B) Survey Question #2.....	33
Figure 3-5: (Company A) Survey Question #3.....	34
Figure 3-6: (Company B) Survey Question #3.....	34
Figure 3-7: (Company A) Survey Question #5.....	37
Figure 3-8: (Company B) Survey Question #5.....	38
Figure 3-9: (Company A) Survey Question #6.....	39
Figure 3-10: (Company B) Survey Question #6.....	39
Figure 3-11: (Company A) Survey Question #7.....	40
Figure 3-12 (Company B) Survey Question #7	41
Figure 3-13: (Company A) Survey Question #11.....	47
Figure 3-14: (Company B) Survey Question #11	48
Figure 5-1: Illustration: Agile Readiness Grid Instructions.....	62
Figure 5-2: Agile Readiness Risk Grid – Radar Map.....	74
Figure 5-3: Agile Readiness Risk Grid – Radar Map.....	75
Figure 5-4: Agile Readiness Risk Grid – Radar Map.....	76
Figure 5-5: Tableau Columns and Rows for Agile Readiness Risk Grid	77
Figure 5-6: Tableau Filters and Marks for Agile Readiness Risk Grid	78
Figure 5-7: Tableau Distributed Team Agility Readiness Map	79
Figure 5-8: Data Mining information within California.....	79

Figure 5-9: Security Standard Matrix Tool Instructions/Overview.....	82
Figure 5-10: Team Distributed Security Standard Radar Metrix	92
Figure 5-11: Performance Maturity	93
Figure 5-12: Risk Equivalency Metrix.....	95
Figure 5-13: Heat/Stacked Bar Chart.....	98
Figure 5-14: Mata Mining on Description - Rating and Description.....	99
Figure 5-15: Dimensions and data for Rating and Description Dashboard.....	100
Figure 5-16: Tableau Rows and Columns for Rating and Description Dashboard.....	100
Figure 5-17: Dimensions and Marks for Rating and Description Dashboard	101
Figure 5-18: Rating and Description Risk Dashboard.....	102
Figure 5-19: Data Mining / Deeper Drive into Rating and Description Risk Dashboard	102
Figure 5-20: Hierarchy of Time set in the ‘Average Security Risk Per Sprint Report’ .	104
Figure 5-21: Hierarchy of Program Increment of ‘Average Security Risk Per Sprint Report’	105
Figure 5-22: ‘Average Security Risk per Sprint Report’ – Dimensionality.....	105
Figure 5-23: Average Security Risk per Sprint Report	106
Figure 5-24: Data mining – Viewing data for Monitoring Security group’s risk per Sprint	107
Figure 5-25: Overall Output Risk Assessment Reports.....	108
Figure 5-26: Data mining - Overall Output Risk Assessment Reports.....	109
Figure 5-27: Deep-dive into analytics for ‘Logon’ Security Standard for all distributed teams.....	110
Figure 6-1: CDAD (Characterize, Diagnose, Anticipate, Distribute) Learning	112
Figure 6-2: Dev-CDAD-Prod Model.....	114
Figure 7-1: Exit Interview - Feedback #1	123
Figure 7-2: Exit Interview - Feedback #2	124
Figure 7-3: Exit Interview - Feedback #3	125

Chapter 1

Introductory to the Study

1.1 Introduction

In today's world of fast development, quick upkeep, and modular changes, to both internal and external applications/solutions, developers needed an approach to build and deliver elastic scalability solutions within a more agile way. Developers, when approached with an opportunity to build a given solution, look for the fastest way to develop with little or no downtime. This is important so they can meet their given SLAs but build with the quality and innovation that is needed.

Because of the shift in agility, and solutions' 'speed to market' timeframes, the days of SOA architecture is quickly becoming a thing of the past. In our next chapter, we will explain why developing via a Service Oriented Architecture (SOA) approach is not the way many companies are now creating software or how they are utilizing team (dev, business, etc.) to build solutions.

However, in the world now of quick-growing Digital and FinTech companies, the mindfulness of getting quick solutions to the market and clientele is now a critical "must. When demands are set on based on technology's growth, the end-users and consumers are now looking for fast solutions to known issues and more significant innovations. These

demands are not just about new technology but also the existing legacy technology that has been built to keep up with the latest technology and demands.

To perform the upkeep of technology, the mindset of building and delivery of development solutions as changed with the introduction of agile development. Along with the introduction of that allows smaller teams to work better in parallel, the usage of Microservices became a reality.

Microservices allows smaller independent teams to build and deploy a ‘break-out’ of a particular service, as independent solutions. Today’s modern deployment of Microservices are frequently customer-facing and typically highly integrated services used to create previously impossible combinations of application functionality. Forrester writes that Microservices have an “important role in the future of solution architecture,” mostly talking about faster solution deployments that focus on greater operational resilience and scalability [17]. Gartner also provided their overview that Microservices “enable unprecedented agility and scalability” [17]. Most of these solutions, leverage ‘API type of contracts’ to interface with a plethora of back-end systems to allow dynamic solutions to the end-user (client or clientele). These solutions allow for the fast adoption of fixes to integrate with solutions needed for full development.

However, these self-sufficient, autonomous solutions are normally ‘self-governing’ solutions that mostly follow self-guiding governance when the service was developed. As each service usually is independently built, the standards and ownership typically come from a ‘bottom-up’ development approach, where the developer building the service owns the overall SDLC model.

1.2 Research Problem

When starting or building a team and or solution for a new Microservice, the agility, the build-ownership, and standards become decentralized between multiple developmental teams.

By following the Microservices mantra, the developmental teams will be broken out to work on separate solutions and services, allowing them for agile development and builds to take place. Speed and safe scalability are important concepts [5]. However, splitting up teams can run into some issues with reliability and responsibilities. Decentralization may allow the team to select the right tool for the right job, but this also means that a developer may not follow a standardized development or technology patterns. Decentralization benefits the developers to have the freedom to choose the best useful tools but lacks the upfront governance that each team needs to follow, as many teams will be building a granular scaling/mixed technology stack. The many issues with the monolithic approach of SOA that it becomes so large and complex, even the developers can't understand their own system. However, even with Microservice, and the emphasis on small, self-governing teams, every fortuitous company need cross-functional teams and procedures to connect the dots [5].

Throughout this paper, we will should how agility is needed, but the needs for a preexisting distributed governance understanding between all teams, with a decentralized development team, are needed.

As governance holds many meanings and standards in development operations, in this paper, we will focus on the below factors:

- 1) Need for developers, within non-centralize (decentralization) teams, to follow and understand the importance of governance
- 2) Usage of models, at the inset and throughout a project, will help guild, or at least level set, the importance of governance between decentralized teams.
- 3) Consider the need to centralize problems and solutions around security needs when developers develop in separate decentralized teams
- 4) Agility needed for risk assessments and communications between decentralized teams for distribution to perform correctly within a Microservice environment/project/teams. Models/procedures should be utilized for adherence to much-needed standardizations within agile development.

1.3 Objective

The objective of this research is to help developers follow a model or methodology when gathering scope, designing build, and or thinking of the final output of a service. We hope to:

- Developing an *Agile Readiness Risk Grid Tool*, to build strong **team governance**, when selecting/building out decentralized agile teams; so best practices and behaviors can be followed throughout
- Constructing a *Distributed Security Standard Matrix Tool*, **risk model**, to educate all team members on a structured governance framework to manage microservices security development and risks that can happen during the breakdown and build-up of monolithic applications.

- Building a lightweight *CDAD* **governance model** to guide developers to Characterize, Diagnose, Anticipate, and Distribute critical standards when introducing and developing a solution within an Agile Microservice framework.
- I have developed a lightweight Dev-CDAD-Prod Model, based on a **DevOps model**. This model helps software development practices, combines software development and information technology operations, follows a development life cycle.

In this research, we will attempt to conclude that the need for a more distributed type of government rather than a decentralized type of governance when building a Microservice.

The initial gathering of information amassed is from survey data garnered from two companies. Initial pain-points and findings gathered will help build and create specific models, which were then presented to development/DevOps teams, in which they give feedback

Our attempt will be that giving first rules/guidelines/learning will be a must to help govern the outcome of a decentralized development solution, such as Microservices.

1.4 Research Question(s) / Hypothesis:

RQ1: What are some of the common problems that a distributed governance model can eliminate or alleviate when developers are ready to start developing their solutions?

H1: Distributed Governance model reduces setup time for Microservice build solutions

RQ2: What impacts are felt when applying distributed standardized models, within decentralized teams at/during the time of development (hurt or help)?

H2: Distributed Governance model reduces risks and garners a better resilience solution

RQ3: What impact administrating these models, provide to team and security standards (hurt or help)?

H3: Distributed Governance models, allows for a better resilient model that subsequently allows for better security standards and more reliable team risk mitigation

1.5 Scope

The research will be limited to two companies.

- Company A and B
 - Start: Gather data from interview/survey focusing on pain-points, what is needed
 - End: Introducing of new governance model to strengthen and “Phase-Into” their current process

The methodology that we will follow will be at an Empirical approach where we will collect our data via:

- Questionnaires/observations/Surveys (Before and After)
- Interviews (Before and After)

I will also utilize an Analytical Simulation approach as I will promote the simulation of data to enter within new Distributed Governance Metrics that we have constructed. The data will come from 2 new models built 1) Distributed Security Standard Risk Matrix Tool and 2) Agile Readiness Risk Grid.

These matrix learning tools are checklists and/or procedures to help guide the developers distributed between the team. It also provides procedures and guidelines to ensure developers are following some standardization of governance, be it security, ownership, and/or control as they are trying to develop services in a fully decentralized agile way. Last, I will analyze the simulated data, via a powerful analytical tool (Tableau) to represent data pulled from the matrix built. The end goal is to use these tools/models and proved they are designed to help the decentralized developmental teams, to analyze, and aid in any Risk mitigation by the distribution of standardization to each team, as well as allowing developers to continue to follow the full Agile Methodology

1.6 Limitations

Each place of business utilizes development methodologies different. Company A and B will utilize different development governance standards from the next. This paper will utilize feedback from 2 developmental teams, from separate companies, to help promote the need for standardization.

Chapter 2

Background on Service-Oriented Architecture and Microservices

2.1 Introduction

Within this chapter, we will discuss the questions that arise from different development methodologies that affect companies' different ways. The days of building an application, via an SOA architectural style, from the ground up with finalized completed requirements, are become a thing of the past. So, questions emerge [22]:

- What is the difference between Microservices and SOA?
- Is Microservices an evolution of SOA or something entirely different?

SOA and Microservices occupy different territories, but Microservices and SOA are similar in many respects.

Microservices have become an intricate part of the developmental landscape in recent years, as developer's gander at their achievements in decoupling their monolithic applications.

Companies like Uber and NetFlix may have brought the methodology to the mainstream; however, many companies are adopting microservices, within their enterprises to increase scalability, deployment speed, and release frequency.

It's crucial for companies to become accustomed to the advantages and disadvantages of microservices, as well as the disadvantages, as they seek to evaluate if this type of architecture is a good fit [2], [32].

2.2 Service-Oriented Architecture

In recent years, even up to today, developers are still utilizing and building extensive monolithic applications utilizing an SOA (Service-Oriented Architecture) approach. Developers utilize the Service Oriented Architecture as distinct components of the application that provide services to other components via a communication protocol over a network [3]. The communication can involve easy data passing, or it could involve two or more services, integrating and connecting services. These services carry out some small functions, such as validating payments, creating a user account, or providing social log-in [3].

However, when developing within a Service-oriented Architecture, developers are less concerned about modularization of an application, and more about how to fashion an application by the integration of distributed, separately-maintained and deployed software components.

When developing, there are two main roles in SOA, a service provider, and a service consumer. The Consumer Layer (human, other components of the app or third parties) interact with the SOA, and the Provider Layer consists of all the services within the SOA.[3], [6]

One big issue is that development, within SOA, becomes too complicated, due to its many procedures. The slowness becomes the downfall. Even if a change is made, the entire build needs to be validated, checked, and possibly re-engineered. While the teams and

functionality may be monolithic and possibly stable in the long term, SOA could not support the cooperation of IT [27].

Microservices is now the approximate next step in the evolution of Service-Oriented Architectures. This architecture is a specific way of developing software mobile and or web applications as suites of independent services — or ‘microservices.’

2.3 Microservices

When developing or breaking down large monolithic applications, services are created to serve only one specific business function. This function can be to utilize, User Management services, User Roles, Search Engine, Social Media Logins, etc. Furthermore, services are entirely independent of each other, in which the service could be written in different programming languages or even use different databases, to create a service. This is a significant change from the singular/monolithic development via SOA.

Also, unlike SOA, Microservices do not use centralized service management. Centralization is almost non-existent, as microservices use lightweight HTTP, REST, or Thrift APIs for communicating between themselves.

In layman terms, monoliths are similar to a big container wherein all the software components of an application are assembled and firmly packaged [10], [22]. The concepts of an SOA are present in modern architecture but have evolved in several ways. Integration tools, patterns, and standards have evolved so that functions and data can be more efficiently delivered and developed.

Service exposure has now evolved into APIs. Application Programming Interface (API) helps to simplify exposure, consumption, management, and, in some cases, monetizing business functions. Many software companies release their API to the public so that other software developers can design products that are powered by its service. The monolithic mantra cannot follow that agility. The building and deploying of APIs, help applications talk to each other without any user knowledge or intervention [27].

New application architectures, such as microservices architecture and API enablement, developers are now able to focus more closely on business logic, continuous development. The result of the build can also be structured to an environment or a particular solution where that build would need to be executed and or launched [10].

The combination of these developments enables solutions to be built in more agile styles and applications to benefit from new levels of elastic scalability and fault tolerance. [10].

The main differences between the two developmental methodologies are prominently characterized by Respodovski. [23]. Please see Table 1 below.

Table 1 Differences between SOA and Microservices

	<i>SERVICE-ORIENTED ARCHITECTURE</i>	<i>MICROSERVICES ARCHITECTURE</i>		<i>SERVICE-ORIENTED ARCHITECTURE</i>	<i>MICROSERVICES ARCHITECTURE</i>
<i>Governance</i>	Common governance and standards. More rigid guidelines to follow.	Relaxed governance, with greater focus on independent teams, collaboration and freedom of choice	<i>Focus</i>	Focused on business functionality reuse	More importance on the concept of "bounded context"
<i>Data Storage</i>	SOA services share the data storage	Each microservice can have an independent data storage	<i>Communication</i>	For communication it uses Enterprise Service Bus (ESB)	For communication uses less elaborate and simple messaging systems
<i>Reusability</i>	Maximizes application service reusability	Focused on decoupling	<i>Message Protocols</i>	Supports multiple message protocols	Uses lightweight protocols such as HTTP, REST or Thrift APIs
<i>Systematic Change</i>	A systematic change requires modifying the monolith	A systematic change is to create a new service	<i>Platform</i>	Use of a common platform for all services deployed to it	Application Servers are not really used, it's common to use cloud platforms
<i>DevOps & Continuous Delivery</i>	DevOps and Continuous Delivery are becoming popular, but are not mainstream	Strong focus on DevOps and Continuous Delivery	<i>Containers</i>	Use of containers (such as Docker) is less popular	Containers work very well with microservices

2.4 Development

When it comes to developing an SOA, the solution is built by a group of the same team members acting on a solution that will entail all that is needing to build the solution from the requirement to a single interface need.

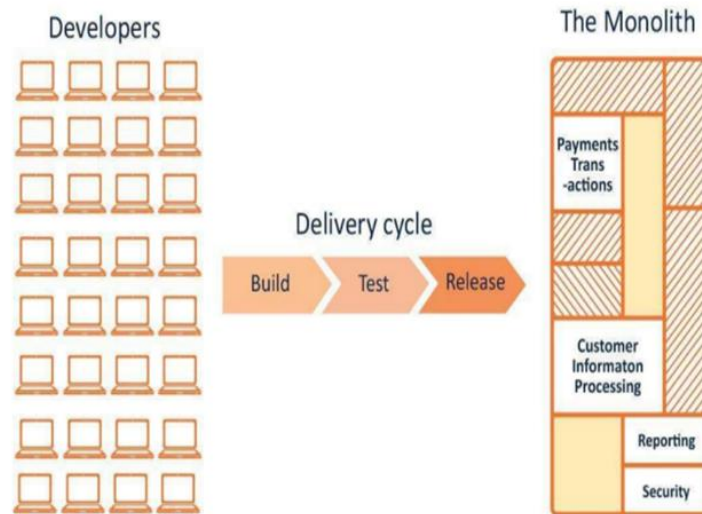


Figure 2-1: SOA build of a Monolith

2.4.1 Why SOA development can cause problems

With the motivation of the creation of building a mono-development build, each team member will be utilizing and working on the same platform and build at the same time. Each team member would work within the same framework, same requirements, and the same environment. The work and deployment of an SOA monolithic build would need to be built, tested, scheduled, and deployed in a gradual way.

This type of build would cause the development to meet below ‘Figure 2-2’ [22]:

Challenges of monolithic architecture	
Inflexible	Monolithic applications cannot be built using different technologies.
Unreliable	If even one feature of the system does not work, then the entire system does not work.
Unscalable	Applications cannot be scaled easily since each time the application needs to be updated, the complete system has to be rebuilt.
Continuous Development	Many features of an application cannot be built and deployed at the same time.
Slow Development	Development in monolithic applications takes a lot of time to be built since each and every feature has to be built one after the other.
Not Fit for Complex Applications	Features of complex applications have tightly coupled dependencies.

Figure 2-2 Challenges of monolithic architecture

Many operational and development dependencies are needed to be thought through, tracked, delivered, and govern.

2.4.2 Reason to utilize Microservices

The above challenges, in Figure 2-2, were many of the main reasons that led to the evolution of microservices. Microservices, aka microservice architecture, is an architectural/development style that designs an application as an assemblage of small autonomous services that are modeled around a business domain [22].

As stated before, within a microservice architecture, each service or API, is self-contained and is applied to a single business capability.

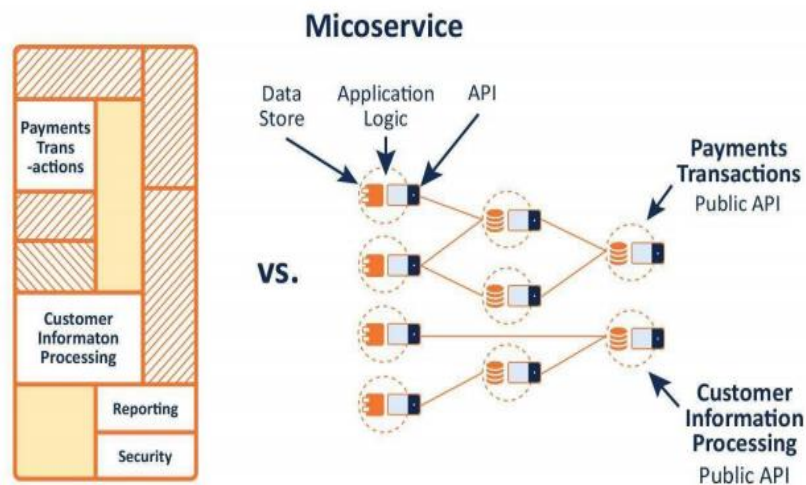


Figure 2-3 SOA vs. Microservice Architecture

The above diagram, Figure 2-3, allows us to visually show the layout and development of a ‘Payment Solution’ as a use case to understand the difference between SOA and MSA. In a monolithic development, the development lifecycle is following based on defined building blocks, predetermined development, but as we have seen can cause inflexibility of development, speed, and most crucial sharing and reusing what has been built.

The difference between the two development methods leads to a more flexible, sharing of resources and services, and this allows developers to deliver faster to consumers as well as allow for:

- Decoupling - Services within a system are primarily decoupled, so the application as a whole can be quickly built, altered, and scaled.
- Independent Development - All microservices can be quickly developed based on their proper functionality.

- Independent Deployment - Based on their services, they can be individually deployed in any application.
- Componentization - Microservices are treated as independent components that can be easily replaced and upgraded.
- Business Capabilities - Microservices are very simple and focus on a single capability.
- Autonomy - Developers and teams can work independently of each other, thus increasing speed.
- Continuous Delivery - Allows frequent releases of software through systematic automation of software creation, testing, and approval.
- Agility - Microservices support agile development. Any new feature can be quickly developed and discarded again.
- Fault Isolation - Even if one service of the application does not work, the system continues to function [17], [22]

2.5 Decentralization and Distributed

Within this chapter, we have gone over the pros and cons and SOA and microservices architecture. The main benefits of each allow developers to build specific technical components; however, microservices allows for quick development, fast deployment, and team agility.

Decentralized team, always need to think, and act as, separate/non-centralized teams.

Distribution of resources, foundations, and standards help to link teams together.

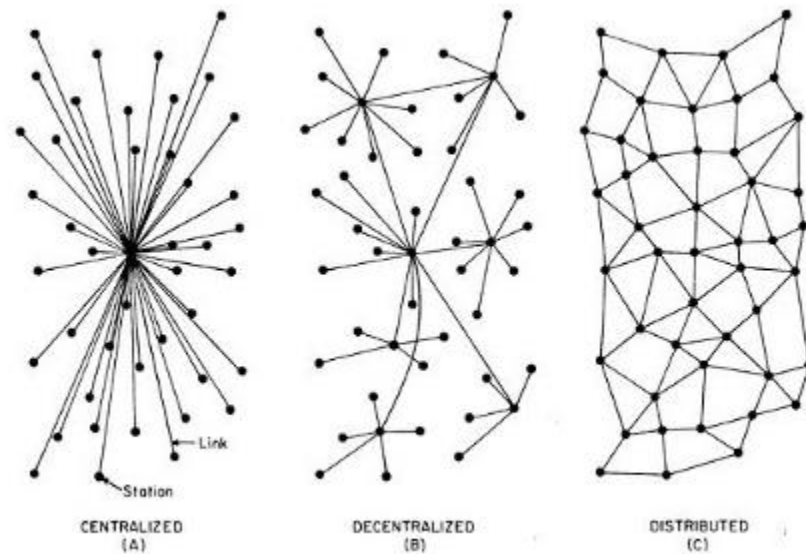


Figure 2-4: Centralized/Decentralized and Distributed

However, according to microservices, agile w/ the use of decentralized teams are the norm.

According to Martin Fowler, a pioneer of Microservice development, the use of decentralization is a significant affirmation of agility in microservices development.

According to Mr. Fowler [15]:

“Perhaps the apogee of decentralized governance is the build it / run it ethos popularized by Amazon. Teams are responsible for all aspects of the software they build, including operating the software 24/7.”

Mr. Fowler also goes forward to state that the decentralization of teams and its governance is not the norm, but companies are still pushing for this [15].

“Of course, just because you can do something, doesn't mean you should - but partitioning your system in this way means you have the option.

The devolution of this level of responsibility is definitely not the norm, but we do see more and more companies pushing responsibility to the development teams.

These ideas are about as far away from the traditional centralized governance model as it is possible to be.”

In retrospect, with the decentralization of teams and standards, there are many pros. A few pros of decentralization were mentioned by Mr. Fowler above; however, there are still a few more. The central abstraction behind decentralization is the delegation of responsibility of teams and the and the allocation of making decisions across various levels in an organization.

Decentralization is the opposite of centralization, as where a central command structure uses a more ‘absolute’ style of management. Decentralization may allow for flexibility and innovation, and encourage teams to take ownership, but it also can breed duplication of work as uniformity of organization policies becomes challenging. The work could become unorganized and possibly create duplication or even holes in standards [4].

To make sure that teams do not have duplication of work, teams, no matter what, must communicate. Communication, via any decentralized team, is key via any agile microservice deployment [32].

“Agile methods are less about software construction and more about humans working together and communicating. No matter what field you’re in, there’s something to learn here.” - Chad Fowler

Last, if your team and or company are new to the concepts of microservices, or if teams have not been introducing the governance and standards of an agile structure, decentralization can provide the lack direction or even spawn security concerns [15].

Even though the decentralization of the team is how the development will be handled via microservice development/developments, the need for a distributed standard, governance, and learning methods are still needed to combat any ambiguity that we list above.

However, even distributed models can still have challenges if the right people, processes, and tools are not in place. Additionally, when governance is distributed, there are still struggles to produce results if teams are not operating from the same set of principles or have a shared understanding of project goals.

The need for a hybrid approach and an adherence to best practices and development standards will help when the distribution is utilized. Also, the need for real-time communication between distributed team members is critical to getting work done and building relationships. Last, the addition of proper training and development standards, as well as best practices, need to be consistently adopted across the teams [1].

Also, with the addition of tool usages, the communication, and impact of optimization are much more significant. The ambiguity or fragmentation between standards and design will be abundantly less between groups of decentralized development groups.

2.6 Decentralization Organization and Use of Tools

The experience of any good decision stems from the team's ability to want to cultivate and grow with change. Within a microservice development structure, the decentralized teams have the ability to create their own group changes, but all teams need to harness the adjustments to realize a change is needed.

According to Gareth Morgan, author of Images of Organization [28]:

“...the challenge is to find small changes that can unfold in a way that creates large effects...”

Decentralization and the autonomy of an individual team are balanced by minimal transformations, that are imposed to keep interdependencies established. Teams are encouraged to devise, and problem solves issues locally but should also be required to establish if these problems or patters are seen anywhere else within the teams [30].

However, when creating a team, you need to make sure that you are not duplicating the existing pain points that have been observed initially, prior to any build. According to Conway's law [12]:

“Any organization that designs a system (defined more broadly here than just information systems) will inevitably produce a design whose structure is a copy of the organization's communication structure.” – Melvin Conway

The beauty of decentralized microservices team allows for teams to create what best fits the need of the build or organization, and however, if a team designs the way they feel the solution to be, and not share or distribute the design, the output is inevitability to be the

copy of that team. The issue here is the solution will stay stagnant as the rest of the decentralized groups, will not see the solution, and most importantly, any pain points noticed.

System and group behavior are also a result of the process and tools that workers utilize to perform their jobs. The usage of a decentralized model can lead to skill challenges in the organization if every team is required to cultivate their own expertise [29].

As we spoke for microservices governance, decentralization is necessary. The decentralization provides the innovation and flexibility that developers crave while maintaining the security and reliability that IT operations folks demand. As developers look to govern applications and teams, to make them more reliable and secure, tools are needed to facilitate and drive, decentralized governance [31].

A governance model for collaboration could include a very rudimentary list of guidelines explaining what collaboration consists of, what teammates should or should not be openly collaborating on. Also, a mode, tool, or platform can be dynamically built to accomplish these goals. Many companies and groups feel this is unnecessary, the openness of the groups, and how the organization wants to determine their goals [11].

The most crucial consideration is adopting a model that resonates with the overall organizational structure and corporate culture. Organizations seeking to affect a change should ensure they are addressing their needs.

When selecting a model to be utilized, a value proposition needs to be determined to understand how any model will be beneficial for decentralized development teams.

Defining the expected outcomes can help clarify the preferred model. The results of any identified risk or procedures that may be missing during an SDLC build will also determine what and how models should be constructed and distributed [33].

2.7 Conclusion

Within this chapter, we went over the development usage of SOA and how microservices have now become the needed go-to development methodology if your company and your teams are prepared. The usage of decentralized teams and then the need for distributed standards are a must mix to propel any development. The pos of utilizing a decentralized team allows for quick, agile development and deployment; however, the distribution of standards, training, and communication is a must need to deploy any successful release successfully.

In the next chapters, I will present findings, via survey data, in which not all teams fully understand what all goes into an exact microservices development structure. We will show the need for standards to be distributed to make the decentralization development team successful.

Chapter 3

Initial Survey

3.1 Introduction

Within the above chapters, we have gone over the definition and overview of SOA and Microservices methodologies. It also described the positives and difficulties found within periodical and written reviews.

Included in this chapter is an initial survey taken, by two companies, who are utilizing the microservice framework and methodologies to help develop faster developmental techniques to help with 1) breaking down old monolithic applications/solutions and to 2) build/create/plan for new builds utilizing agile/API builds.

3.1.1 Reason for Initial Survey

This “survey” is to help understand how and what the development team looks for, in terms of governance, standards. Also, to see and understand what pain points are, if any, the developer would like to see changed, fixed, or modeled for better learning techniques.

Within the survey, we gather information to help further understand and retrieve:

- A ‘picture’ of the Microservice development team
- Initial understanding that development has within their current Microservice process
- What type of governance, if any, the development teams are following?
- What type of governance would the development team like to follow?
- Amass pain points developers are currently running into w/ the loss of governance and or structure

- Understand issues developers need to have addressed
- What possibly can be added to deliver a stronger Microservices: tools, processes, and or learning models

3.1.2 Companies utilized within the survey

Development participants, from 2 companies, helped with the survey questionnaire. The participants chosen were developers who are currently working on an agile/microservice solution within their own companies.

The two companies, within the survey comprised of:

1) Company A

- Industry: Banking solution
- Size of company: 10,000+
- Audience: Developers/PO

2) Company B

- Industry: Digital Publication Services
- Size of company: 5,000+
- Audience/Participants: Developers/DevOps

3.1.3 Survey Short description

The following survey description was provided to each participant who participated in the survey. Both companies were providing the below survey description in order to give an overview of the request/questions:

In today's fast-moving developmental needs, many companies, are starting to dedicate resources and personnel to help, start rethinking the development of large/older monolithic solutions, by starting to develop separate functional services (microservices).

Because of the speed and flexibility of development, companies are trying to ensure they have a tightly governed model to help with the creation and completion of these microservices.

This "survey" is to help understand how and what the development team looks for, in terms of governance, standards. Also, as a participant, please help list what some pain points are seen and what should be changed, fixed, and or modeled for better learning techniques.

Please take a few minutes to help us collect data that will help build an excellent underlying model/process.

3.1.4 Questions listed from Survey

Each of the 11 below questions has been provided to both companies for feedback. Many questions follow a multiple-choice request. However, there are a few open-ended questions, in which we called 'Free Formed' answers. The 'Fee-formed' survey questions, provide the participants' written space to answer, 'open-ended question,' to aid for a 'deeper-dive' answer and description.

Both of these types of questions allow us the ability to understand the overview and understanding of the participants.

1) Survey Question #1 – Current Developmental structure

Q1: Within your current developmental structure, is there a robust and well-defined governance model/framework that, you as a developer member, follow for every Microservice build?

Q1: Survey's answer choices:

- a. Yes
- b. No

2) Survey Question #2 – Governing Body

Q2: How important is having a 'governing body' define the structure and outcome of the Microservice?

Q2: Survey's answer choices:

- a. Very important
- b. Median importance
- c. Low in the priority of a build

3) Survey Question #3 – Owner of Microservice

Q3: Once a Service/API is built and completed, who should own the Microservice after the build (ex. Corrections, additions)?

Q3: Survey's answer choices:

- a. Project Manager

- b. Scrum Manager**
- c. Developer who built service**
- d. Client – Product Owner**
- e. Run Team**
- f. Agile Team**

4) Survey Question #4 – What standards does a developer need

Q4: As per discussion, your API/Microservice team, is currently working on API governance standards. Standards such as - Naming conventions, Versioning rules, and JSON validations, etc. What other standards would API/Microservices developer, like to see implemented?

Q4: Survey's answer choices:

Free Formed answer (open-ended question for a deeper-dive answer and description to answer)

5) Survey Question #5 – Advantages of Microservices

Q5: What are the advantages of Microservices?

Survey's answer choices:

- a. Independent Development**
- b. Independent Deployment**
- c. Fault Isolations**
- d. Mixed Technology Stack**

e. **Granular Scaling**

f. **All of the above**

6) Survey Question #6 – Current following Centralized or Decentralized

Q6: Does your developmental group follows a Centralized to Decentralized governance framework when it comes to building out Microservices?

Q6: Survey's answer choices:

- a. **Centralized:** decision-making authority lies with a group or individual at the top. The other members of the organization then work to carry out the decisions made by top-level leaders
- b. **Decentralized:** many decisions are made by mid-level or lower-level users

7) Survey Question #7 – What Governance Framework

Q7: What governance framework would you like to see utilized?

Q7: Survey's answer choices:

- a. **Centralized**
- b. **Decentralized**
- c. **Hybrid**

8) Survey Question #8 – What needs addressing

Q8: What do you feel, still needs to be addressed when it comes to API

development, especially when building microservices?

Q8: Survey's answer choices:

Free Formed answer (open-ended question for a deeper-dive answer and description to answer)

9) Survey Question #9 – Addressing Current Pitfalls

Q9: What pit-falls, as a developer, do you feel that you run into and that needs to be still addressed from an API/Microservices Point of View?

Q9: Survey's answer choices:

Free Formed answer (open-ended question for a deeper-dive answer and description to answer)

10) Survey Question #10 – Level of Governance

Q10: What level of governance would you, as a developer, working on Microservices, like to see?

Q10: Survey's answer choices:

Free Formed answer (open-ended question for a deeper-dive answer and description to answer)

Survey Question #11 – “Learning Model” Requested by Developer

Q11: If a "learning model" from a governance POV is to be built/constructed, what from the below feature selections, would you as a developer like to see expanded... that would better help you perform your job?

Q11: Survey's answer choices:

- a. Process**
- b. Tools**
- c. Versioning**
- d. Continuous Integration**
- e. Continuous Development**
- f. Team structure**
- g. None**

3.1.5 Outcome of Survey

Each survey question, either be a multiple-choice and or 'free Forme' questions, was quantified and present within a graphical chart. Charts help quickly decipher the output and garner usable data to start generating a conclusion and help push research.

Understanding the developmental needs, pitfalls, and current usage of microservices, this will again strengthen our research methodology and gather venerable data.

1) Survey Answer #1 – Current Developmental structure

Q#1: Question: Within the developmental structure, is there a robust and well-defined governance model/framework that, developer members, follow for every Microservice build?

Q#1: Observation: The below charts allow for a profound understanding that both companies do not align with a ‘true’ decentralized model that developers should follow if wanting to deliver in a Microservices developmental method.

Q#1: Output from multiple choice answer:

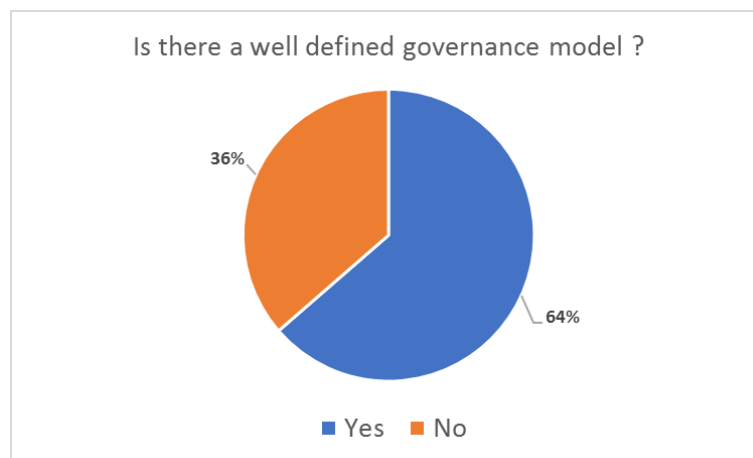


Figure 3-1: (Company A) Survey Question #1

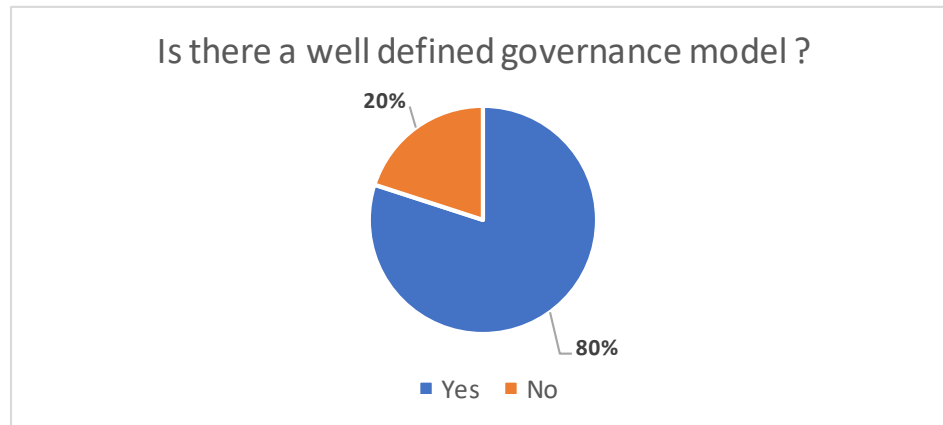


Figure 3-2: (Company B) Survey Question #1

2) Survey Answer #2 – Governing Body

Q#2: Question: How important is having a ‘governing body’ define the structure and outcome of the Microservice?

Q#2: Observation: Within these below data gather, both companies made it confident that a governing body is needed within their organization to be followed. The data is essential as the participates are stating, even if decentralization is favored, a type of governance is critical to follow.

Q#3: Output from multiple choice answer:

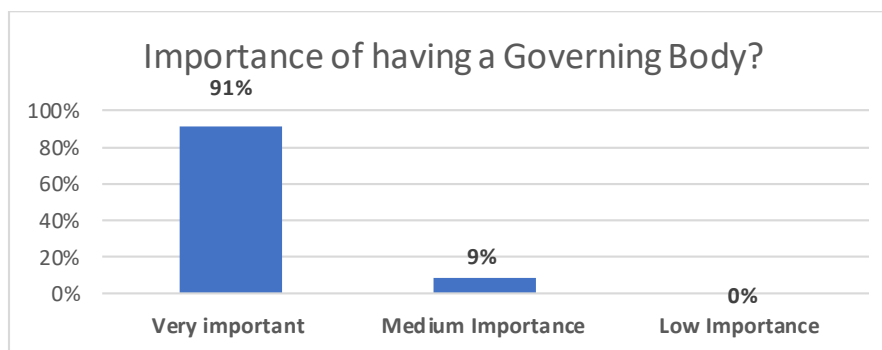


Figure 3-3: (Company A) Survey Question #2

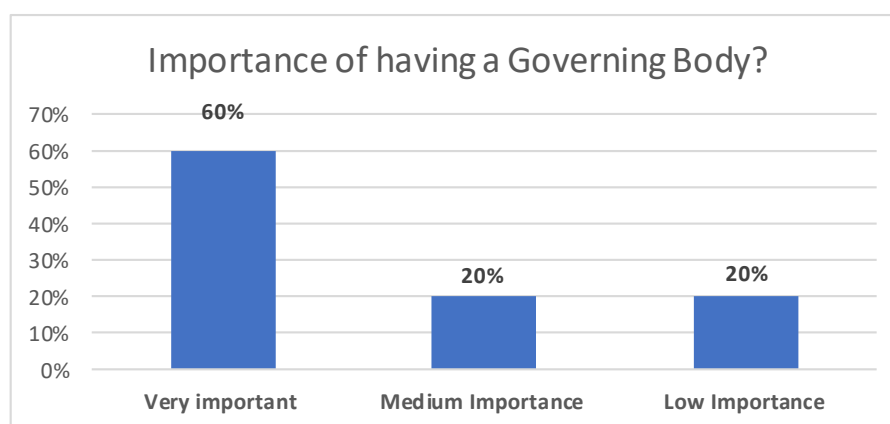


Figure 3-4: (Company B) Survey Question #2

3) Survey Answer #3 – Owner of Microservice

Q#3: Question: Once a Service/API is built and completed, who should own the Microservice after the build (ex. Corrections, additions)?

Q#3: Observation: Question three was to get a good understand of pure ownership of code. The survey data shows, for certainty, there is a need for governance around ownership, within decentralized teams. However, the findings made it very clear that both companies (A and B) are not in agreement with whom 1) owns Microservice service, 2) the API, 3), or even the code after the development is completed.

Both companies do list a percentage, that the developer should own the code, but may participants appear to say otherwise. Inconsistency garners there is a problem within the teams, and if the teams understand the standards, and also if the teams are aligned.

Q#3: Output from multiple choice answer:

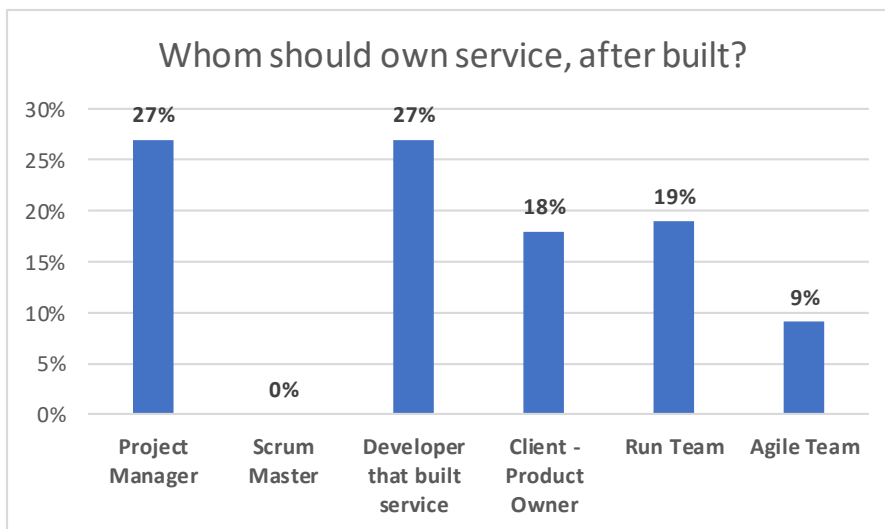


Figure 3-5: (Company A) Survey Question #3

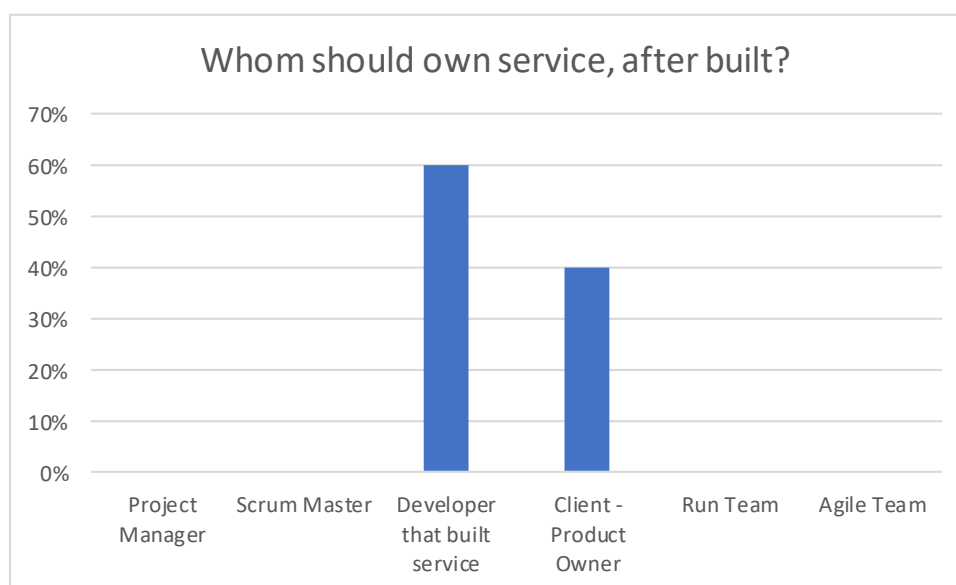


Figure 3-6: (Company B) Survey Question #3

4) Survey Answer #4 – What standards does a developer need

Q#4: Question: As per discussion, your API/Microservice team, is currently working on API governance standards. Standards such as Naming conventions, Versioning rules, and JSON validations, etc. What other standards would you, as an API/Microservices developer, like to see implemented?

Free Formed Answer Q#4: Please see below for output given by developers on what aspects they would like further discussion on as well as standardizations.

Q#4: Observation: Below updates, from both companies, describe the importance of standardizations, rules to follow, and a good picture of what is needed to strengthen a typical developmental release. The two companies state many items need to be talked about and vetted, the best they can, to provide a standard(s) for development.

Q#4: Output from the free-formed answer:

Table 2: (Company A) Survey Question #4

<ul style="list-style-type: none"> • Need Strong rules on usage
<ul style="list-style-type: none"> • If a microservice is developed, by multiple teams, and the developers are creating business services on top of that data service, there is no real picture of the usage of the service
<ul style="list-style-type: none"> • Should implement more robust governance around usage
<ul style="list-style-type: none"> • Security standards, around access, to the service (tokenization) is to be the most critical aspect of standardizing microservices
<ul style="list-style-type: none"> • Security Standards especially for Internal APIs
<ul style="list-style-type: none"> • Need to have control on release process for all API/microservices
<ul style="list-style-type: none"> • Release project broad in Agile JIRA program to be added

• Check/list procedures to make sure governance process being followed
• Standards on Database connections from architecture POV
• Domain-driven pattern implementation will be helpful
• Pivotal Cloud Foundry (PCF) Standards and guidelines
• Service Ownership
• Security Standards – like a set of rules to follow while developing services
• Maintaining a 3 rd party/open source libraries should be used when developing microservices
• Packaged Structure also needed to maintain standards during the build

Table 3: (Company B) Survey Question #4

• Define team for microservice ownership
• Once delivered what standard or whom will monitor the service
• Asynchronous messaging techniques to be used (define and list)
• Control on open source tools
• Overview of size/number of services
• Tokenization needs to be standardized
• Containerization list
• Logging standards to be laid out to help troubleshooting/usage
• Share pattern recognition if the team(s) find or runs in to
• Authentication standards

5) Survey Answer #5 – Advantages of Microservices

Q#5: Question: What are the advantages of Microservices?

Q#5: Observation: Each answer below, follows the true aspects and needs for utilizing the Microservice's developmental methodology. Developers and the agile team look for ease of development and deployment, granular scaling, and a way to choose their own technology to allow for fast deployment. These benefits purposefully follow a solid agile build but also bring the ownness to the team to provide and follow a string methodology.

Q#5: Output from multiple choice answer:

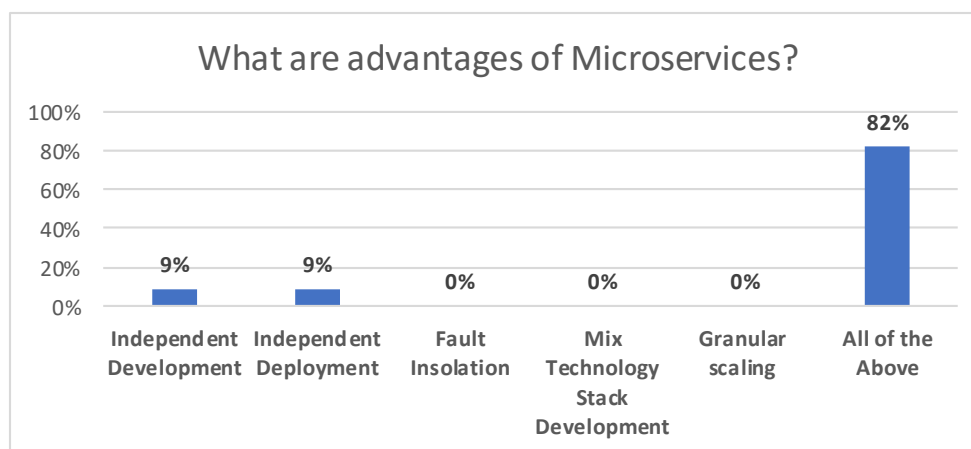


Figure 3-7 (Company A) Survey Question #5

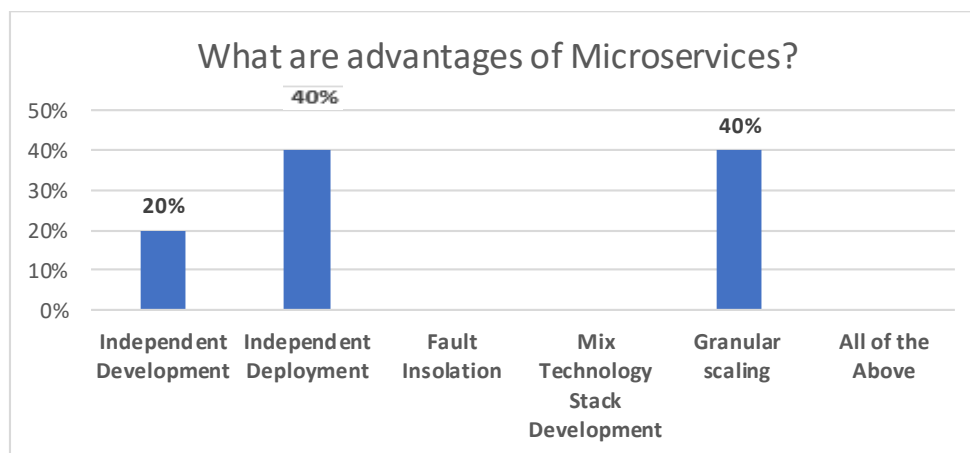


Figure 3-8 (Company B) Survey Question #5

Survey Answer #6 – Current following Centralized or Decentralized

Q#6: Question: Do you feel your developmental group follows a Centralized to Decentralized governance framework when it comes to building out Microservices?

Q#6: Observation: As described in Chapter 1, ensuring the use of a Microservice deployment, the ownership, and framework, will be led from a Bottom-Up approach. A bottom-up approach is the piecing together of systems to give rise to more complex systems, thus making the original systems sub-systems of the emergent system. The bottom-up approach (developer owns and builds) is vital for a Microservice build/team, as it allows for quicker development and deployment. [7].

From the answers garnered below (Figure 3-9 and Figure 3-10), both companies are not aligned, organized, and no real universal governance has is defined. No defined approach will cause communication and risk mitigations to suffer if teams are not aligned with which methodology/framework they will follow.

Q#6: Output from multiple choice answer:

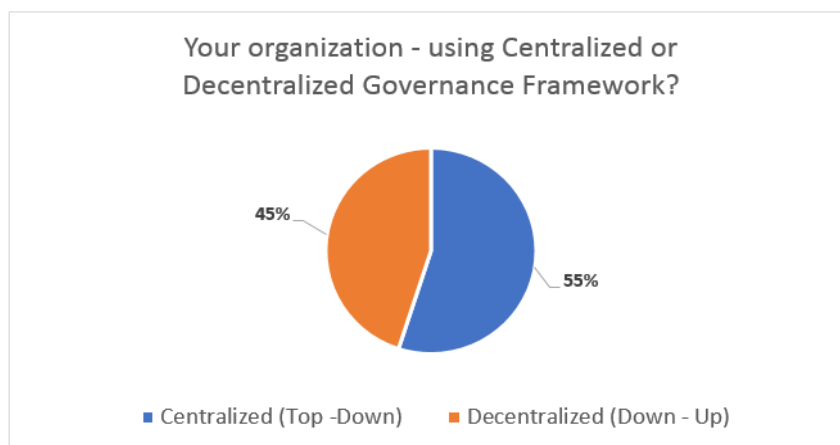


Figure 3-9 (Company A) Survey Question #6

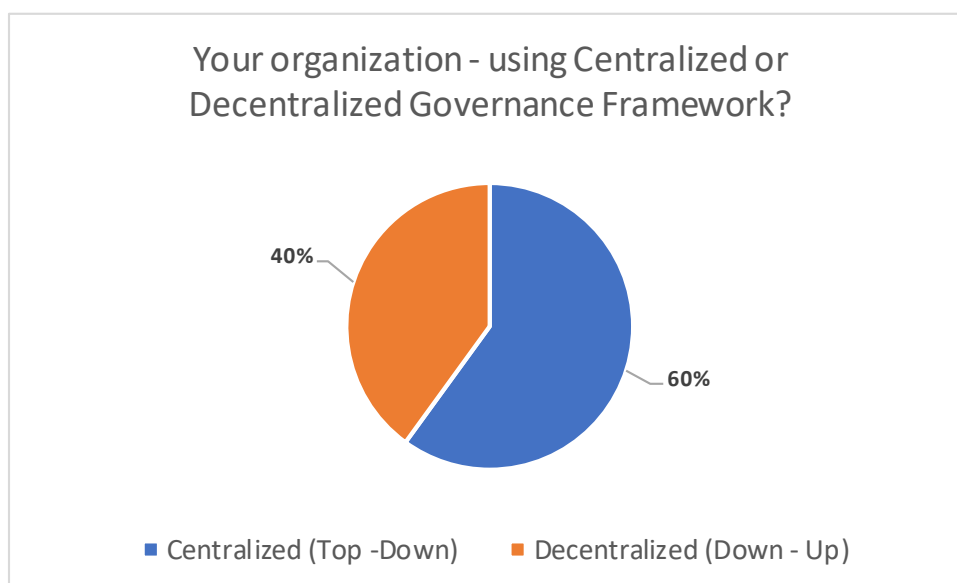


Figure 3-10 (Company B) Survey Question #6

6) Survey Question #7 – What Governance Framework

Q#7: Question: What governance framework would you like to see utilized?

Q#7: Observation: Established within Question 6, both companies were requested to state what type of governance framework, they are following, within their Microservices development process.

From the below data gathered (Figure 3-11 and Figure 3-12), an overwhelming observation can be seen. Less than 30% of both companies feel they are following a decentralized methodology, and both companies state they are following a ‘Hybrid’ approach to their Microservices development.

These observations bolster the question of why decentralized teams and governance are not working. It also brings to question, what more can be done to follow and strengthen standards for these companies.

Q#7: Output from multiple choice answer:

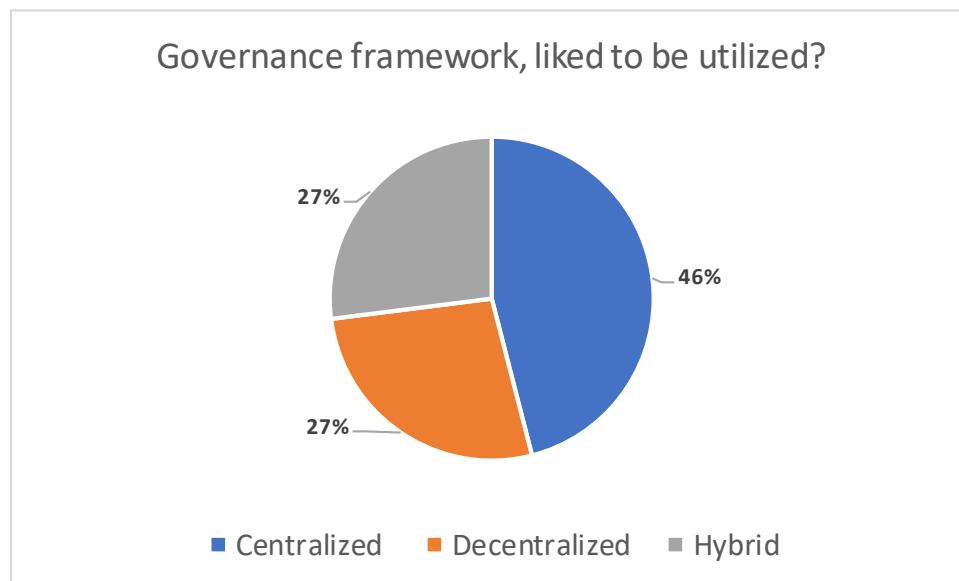


Figure 3-11 (Company A) Survey Question #7

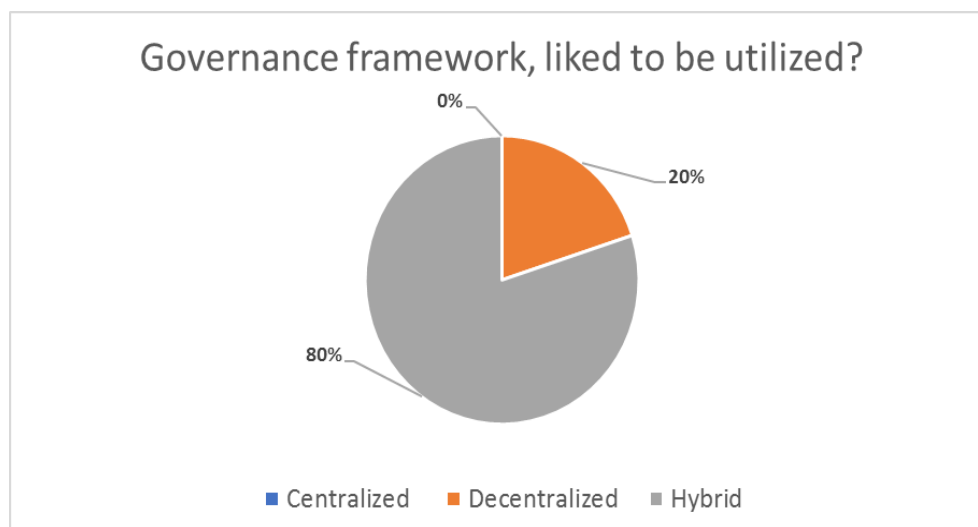


Figure 3-12 (Company B) Survey Question #7

7) Survey Answer #8 – What needs addressing

Q#8: Question: As a being part of the Microservice project, what still needs to be addressed when it comes to development, especially when building microservices?

Q#8: Observation: Below updates were provided by developers who are expressing what they feel needs addressing. The list of below requirements demonstrates that many standards need addressing. Observations should be shared and delivered to each decentralized team, to allow for maximum understanding between the teams.

As Microservice development, handles many moving parts, there is a strong need in which developers need strong team leadership, also requesting to robust user stories, help with developmental tooling, and other standards. The output of this question describes the enormity of a build and the need for standards in an agility build.

However, the participants embrace the quick decentralized agile approach, but the feedback also reveals that many standards need to be followed.

Q#8: Output from the free-formed answer:

Table 4 (Company A) Survey Question #8

Cataloging would cut down on redo's
Currently, most of the agile teams are not aware of a lot of API's what is already available
Needs to be a list of APIs, so PO's are aware
Dev portal infrastructure for Internal API's
Consistencies for 1) Tooling and 2) Contract design
Need Clarity on requirements w/ proper acceptance criteria and user stories
Need improvement on the flow diagrams and architectures
Generate a list and share test scripts with developers
PCF (Pivotal Cloud Foundry) services need to be more valuable and access to both QA/DEV environments
Automation of test scripts needed to reduce regression and QA efforts
Technical decisions/Functional discussion have to be taken at the initial stages of the development so that the flow and governmental standards go parallel between teams
Sharing Technical decisions/Functional discussions will make teams work more efficient if there are many changes in between that they need to be aware of
Need to have a STRONG BA (Business Admin) who has complete knowledge about the functionality of the build
Security Standards
Required some learning sessions on best practices for security output and standardizations between teams
<ul style="list-style-type: none"> • Knowledge sharing for new tools/open source - Coding and Best Practices
<ul style="list-style-type: none"> • Need to follow some form of template project with default functionality to make development time faster
<ul style="list-style-type: none"> • Need to have documentation on security, CI/CD before starting any new microservice build

Table 5 (Company B) Survey Question #8

<ul style="list-style-type: none"> • API-First driven design
<ul style="list-style-type: none"> • Challenge is freedom for mixed technology stack can create challenges in hiring/learning/collaboration/retention, to solve this we need help standardizing on our tech stacks and governance
<ul style="list-style-type: none"> • Microservices taking away operators' complexity and increases development complexity. In microservices, we need to be careful about how small service needs to because it is easier to split service apart but VERY difficult to combine.
<ul style="list-style-type: none"> • Domain-Driven Design helps during microservices design
<ul style="list-style-type: none"> • Another challenge is in order to deliver the particular business value it needs to touch multiple microservices resulting in multiple teams and to solve this we had to create a solution team covering multiple departments

8) Survey Answer #9 – Addressing Current Pitfalls

Q#9: Question: What pit-falls, as a developer, do you feel that you run into and what needs addressing from an API/Microservices Point of View?

Q#9: The data provided below (Table 6 and Table 7) were a tremendous help to understand the actual pitfalls that the development teams are currently facing. The below data points allow defining issues that need to be looked at, addressed, and turned into a learning tool/model.

Both companies A and B requested standards, precise requirements, and for the standards to be shared between teams.

The decentralized team mostly works in a siloed approach; however, the below observations are requesting the distribution of standards, vulnerabilities, and notification of changes, need to be communicated.

Q#9: Output from the free-formed answer:

Table 6: (Company A) Survey Question #9

<p>Few standards are missing: Including when we need them</p> <ul style="list-style-type: none"> • Security • Vulnerability issues • Logging standards
<p>How can we stop and learn as we grow? Are there models that can be followed as learning tools</p>
<p>Changes in requirements and how can we implement them, and keep track</p>
<p>We need a very strong PO (Product Owner) and Scrum Master, who has a clear idea of the objectives that need to be achieved.</p>
<p>Goals need to be laid out before a decentralization of teams</p>
<p>Need collaboration between both Product Owner and Scrum Master</p>
<p>Ownership of services (both tool and builder)</p>
<p>We have started working on the Microservices, but Development standards change frequently, once we are finalized with the standards, we can avoid rework on the same services.</p>

Table 7: (Company B) Survey Question #9

<p>Challenge is freedom for mixed technology stack can create challenges in hiring/learning/collaboration/retention, to solve this we need help standardizing on our tech stacks and governance</p>
<p>Microservices taking away operators' complexity and increases development complexity. In microservices, we need to be careful about how small service needs to because it is easier to split service apart but VERY difficult to combine. Domain-Driven Design helps during microservices design</p>
<p>Another challenge is in order to deliver the particular business value, and it needs to touch multiple microservices resulting in multiple teams, and to solve this, we had to create a solution team covering multiple departments.</p>

9) Survey Answer #10 – Level of Governance

Q#10: Question: What level of governance would you, as a developer, working on Microservices like to see?

Q#10: Observation: The below answers allow for a step now into what standards need following. The data gather, demonstrates a need and a wanting for the creation of standards, also ownership to take place, as well as sharing (distribution) of knowledge to occur throughout projects (Development, Deployment, and Monitoring)

Q#10: Output from the free-formed answer:

Table 8: (Company A) Survey Question #10

Standardization for: <ul style="list-style-type: none"> • Hosting • Tokenization • Access security • Input validation • Naming conventions
Control on: <ul style="list-style-type: none"> • Coding Standings • Code review process • Requirements • Design
Multi-level standards to share with teams should be fine
Share existing level of governance
Standardization and ownership
Functional and Architectural level decisions if we can be taken before starting the sprint development, will be helpful to complete the sprint on time and deliver more effective productivity
Decentralization of team to work together

Table 9: (Company B) Survey Question #10

REST API Design best practices
Initial design about microservices like how small microservice needs to be and specific criteria to follow while creating microservice. If domain driven design used, then more specific guidelines around bounded context and domains to decide.
Considering latest tooling for logging, monitoring and alerting, as there is scope to reduce traditional responsibilities like performance testing, end to end testing but we are still doing both as well as security within the MS
Initial design about microservices like how small microservice needs to be and specific criteria to follow while creating microservice. If the domain-driven designs are used, then more specific guidelines around bounded context and domains to decide.
Considering the latest tooling for logging, monitoring, alerting, There is scope to reduce traditional responsibilities like performance testing, end to end testing, but we are still doing both.
Practices for change of Authentication

10) Question #11 – “Learning Model” Requested by Developer

Q#11: Question: If a "learning model" from a governance POV was to build/construct, what from the below feature selections, would you as a developer like to see expanded... that would better help you perform your job?

Q#11: Observation: The culmination of the survey questions, the request for opinions on developmental needs, wants, and pitfalls, helped to garner information on what can be corrected or built. This information can help drive exceptional information to help inform a microservice developmental team.

The below output allows for the developmental teams to understand what developmental areas need better tooling, learning models, and techniques in order to succeed.

Q#11: Output from multiple choice answer:

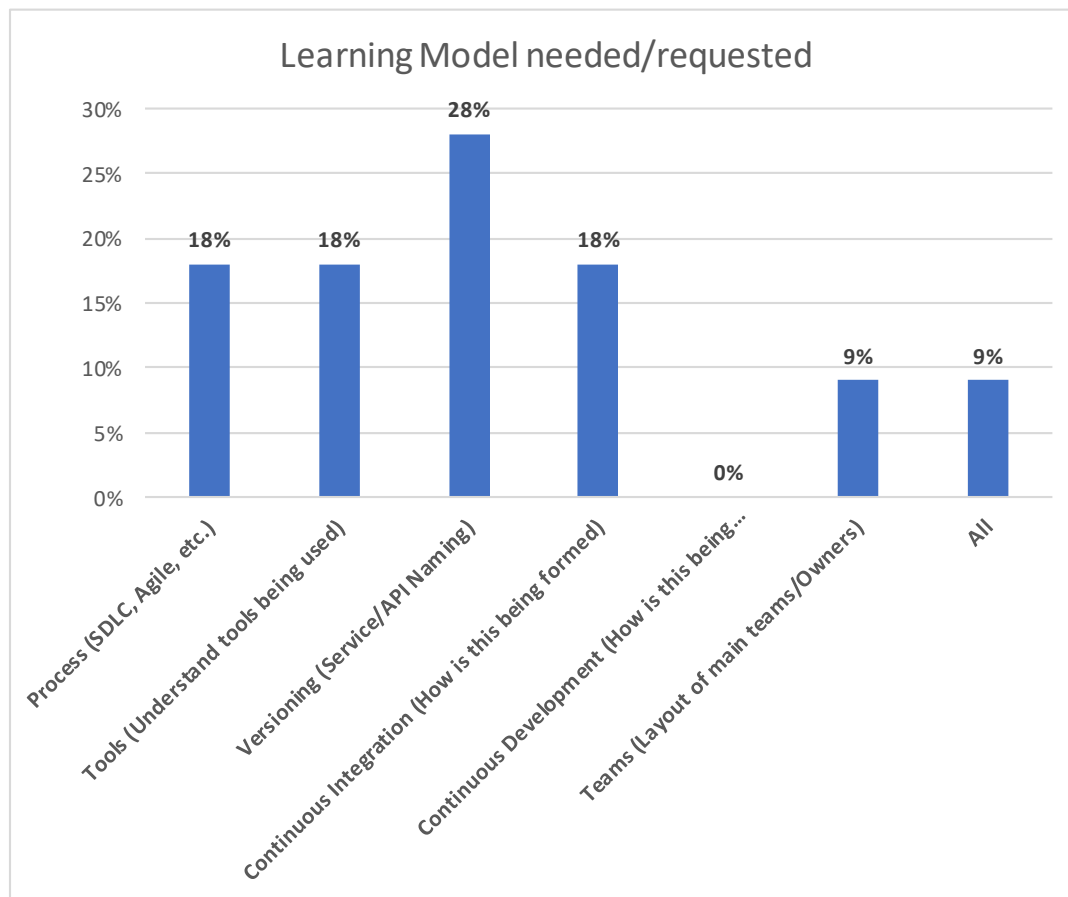


Figure 3-13: (Company A) Survey Question#11

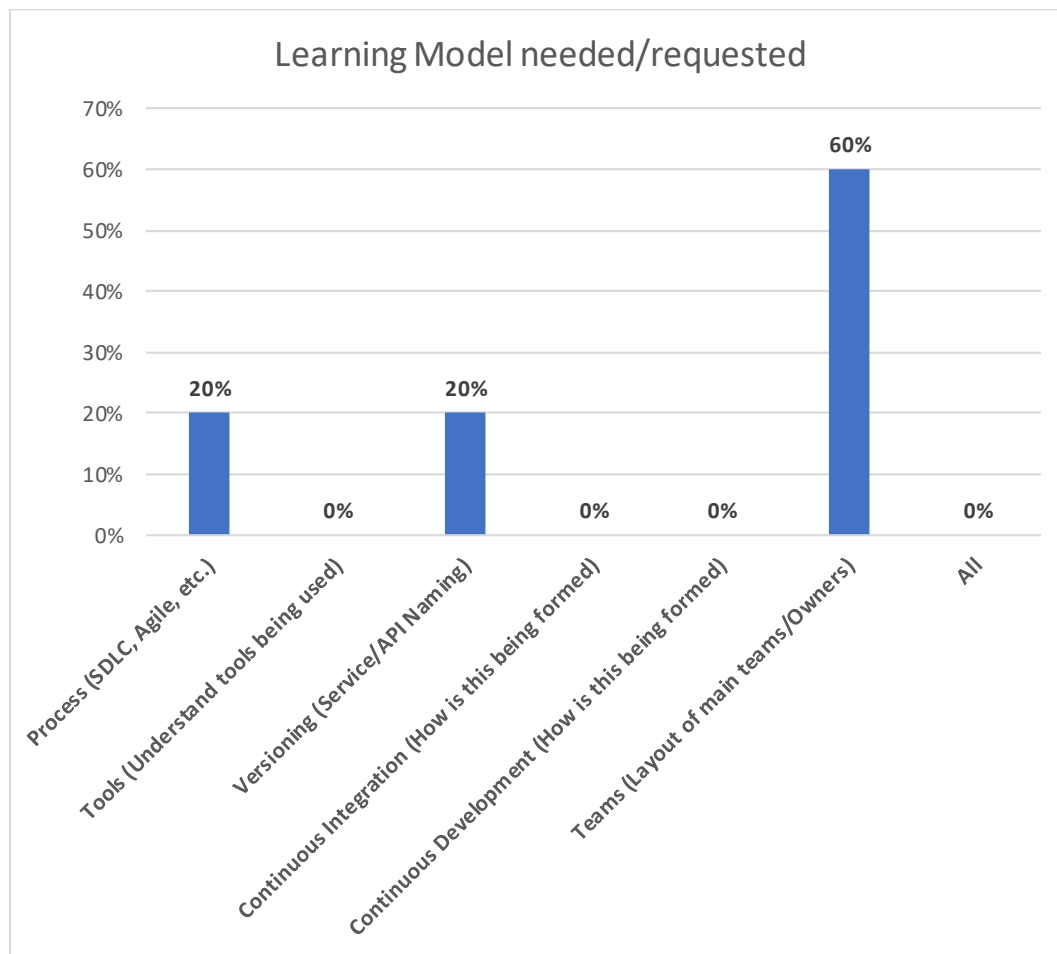


Figure 3-14: (Company B) Survey Question #11

3.2 Outcome of Survey

The outcome of the initial surveys demonstrated that the developers need a reliable governance standard, but a lack of understanding of where and how governance should be addressed.

Below are a few points outcomes/lessons which need addressing:

3.2.1 Governance Standards

a) Standards not finalized

- Per the survey, more than ½ of the developers that feel they are using centralized governance mythology and the other ½ feel they are following a decentralized methodology while developing Microservices
- However, the same developers are still requesting that they see many inconsistencies that need addressing
- Developers are requesting standards to be fully/partially vetted throughout the process (follow correct standards/governance)
- Developers are requesting a Hybrid approach (between centralized and decentralized) to understand how governance should be followed
- 60% of developers are not comfortable when governance issues need to be addressed or taken care of, within the development process
- Collaboration is either slow or non-existence between the decentralization of a development team

b) Lessons to be learned:

- A fluid-hybrid approach is required to keep all agile developers, updated with ever-changing standards, in a project
- Microservices development standards and team-leadership need deciding, and before the start of development, this will help with sharing ideas and will support significant application development to build modular services
- Deployment Instructions needed

- Technical and functional requirement changes are needed before the start of a sprint/project.
- Collaboration drives the fluidity of a development team
- Standardization, between the team, helps with rework

3.2.2 API Ownership

a) Lack of Ownership

- There is no definitive answer on whom should “OWNS” the fully developed service/API
- Lack of ownership can cause the services to become stale or also can cause issues with Versioning, naming, and standardization
- Lack of ownership will slow the process
- Coding standards and checks will suffer if no centralized owner
- If developers do not understand the standardizations or team structure of a true Microservice Methodology, then then the correct Microservices framework will not follow the ever-needed agility, reuse, building efforts.

b) Lessons to be learned:

- Agility between development teams happens from a “Bottom-up” development approach (developers build and owe)
- Within a true Microservice agility decentralized team approach, the developer is required to create, own, maintain, and govern the services that they build.
- Model/Lists need to be built before deployment, for processes/governance

3.2.3 Security Rigger: *Hard to follow w/in a decentralized development team*

a) Lack of Standards

- Developers, within the survey, stated security standards are most likely to suffer if developers do not follow a process within teams that are not centralized (decentralized), but still, need to be agile
- Standards are missing, no way to track changes
- No central contact or a way to share requirements (teams are siloed, no communication)
- If risk how to check if another team
- Over accountability

b) Lessons to be learned:

- API Standards / Best Practices / Naming conventions: setting up standardize criteria is crucial, (before, during and after) the build
 - Keeping the entire all decentralized development teams updated w/ latest changes, is a must to keep the developers/project members updated.
- CI/CD is crucial to developing an agile
- Standardization help w/ ease of troubleshooting
- Risks tracking and validation needed
- Clarity of security standards, between all phases of the build, is crucial
- Simplify documents and share between team
- Keep agile, but have communication

Chapter 4 Research

4.1 Chapter Introduction

Interesting findings demonstrate, standardizations in security, ownership, governance, and control are needs and requested on the behest of developers within the survey in chapter 3.

In order for Developers to make sure they are developing with similar standards, as well as utilizing the agility of microservice standards, their needs for a plan to request learning tools, checklists. These procedures will help guide the developers and distributed standards between teams. These tools and models will help to ensure the decentralized teams are following some standardization of governance, be it security, ownership, or control.

4.1.1 *Research and Setup*

Through this research, we aim to establish a basic understanding of the definite need for agility in microservice development in order to breakdown large monolithic solutions to create and reuse critical services easily.

The research will allow for the rethinking of the dedication of standardized governance that needs to be dispersed and understood via the multiple decentralized teams that are so critical in the practical need for microservices development and success.

We will ensure that developers' education is of the utmost importance. The models and approach we will confirm developers are educated well prior to the build ever starting.

4.1.2 *Experimental procedures*

Thought-out this and the next chapter (Chapter 5), the research goes through the construction of a few tools to aid developers with an awareness that distributed communications, risks, and mitigations are essential for decentralized teams.

The basis of the research was to gather information, via survey outputs and literary reviews, to assemble evidence and build procedures that will aid in the abilities for developers to design successful builds, utilizing decentralized teams.

This research utilizes simulated data to garner information to produce real-world examples and presents the benefits of the designed tools. The paper uses simulated data within both, newly created Team Readiness Distributed Security Standard Matrix Tool and the Distributed Security Standard Metrix Tool to simulate the need and benefits of these tools with decentralized teams via development.

To appropriately take advantage of this research, one would first select a project where their development methodology would abide by employing a decentralized team within an agile build. Before development, each decentralized team would need to recognize non-communication or lack thereof would be a deterrent to a successful agile build.

For each project, each decentralized team would utilize the created 'Team Readiness Distributed Security Standard Matrix' (see 'Figure 5-1') to garner information to exhibit if and how prepared each decentralized agile team is and if there are ready to develop. Each team would select their maturity level and would need to distribute their findings to each decentralized team. The finding allows for each team to understand what maturity levels the other needs to be successful at a particular moment within a build.

Second, the decentralized teams would also execute the procedures within the ‘Distributed Security Standard Metrix Tool’ (see ‘Figure 5-9’). Just as with the ‘Team Readiness Distributed Security Standard Matrix,’ each team select their security maturity level and would need to distribute their finding to each decentralized team. The purpose of this is to allow all teams to be aware of each team’s maturity level and if all teams are on the ‘same page.’

Research has shown that the lack of maturity and risk communication can cause issues and delays between each decentralized team, which can cause many different teams to not be in sync with each other.

Within this paper, models have been built to signify the need for a formation of standards. Thought-out this and the next chapter (Chapter 5), the overview of each tool and model has will be over in detail.

4.1.3 Learning Tools and Model overview

Based on the procedure and output from survey answers as well as literary research, this project will focus on two Risk and Readiness tools as well as two Learning models.

Each tool will be presented with:

- Overview/instruction of the tool
- Why the tool is useful
- Data analytics to better understand the benefits of the outcome for the developer.

In retrospect, each tool and or model will exclusively provide different models to quantify the importance of the data gathered.

4.2 Tools / Models built

Each of the below will represent data, and setup tools, to allow developers and DevOps to focus on specific standards that are needed to perform positive decentralized team management and governance.

Each tool and learning model will provide the needed governance that should be distributed between the decentralized dev teams.

From the output of the survey, in Chapter 3, the focus will be on:

- 1) Creation of Agile teams
- 2) Security readiness and possible risk
- 3) Creation of models that can be followed at every point of development

4.2.1 Overview of Tools built

Below are the two Risk and Readiness tools / Learning models built. Each model provides a separate overview of standards that need to be focused on, as well as supporting models to strength support

4.2.2 Team Readiness Distributed Security Standard Matrix Tool

The developmental reason for the creation of this matrix is to help create, form, display, and instruct on how an Agile team is behaving and maturing. This tool is to be used to make any adjustments in an agile way, and not to deter any production or solution that is has been developed.

The structure of a team, be it centralized and or decentralized, but still need a central readiness to ensure the teams and structure are mature and reliable.

4.2.3 Distributed Security Standard Matrix Tool

The purpose of the tool is to assess the security health for each decentralized microservice team of developers. This tool can be shared between teams or the DevOps Manager to provide an update of the health for the entire team.

4.2.4 CDAD (Characterize, Diagnose, Anticipate, Distribute) Learning Model

Based on each matrix, the learning models are to help combined the learnings 1) the survey, 2) the output from the primary and supporting models, and 3) gathering of all learnings.

The main goal is to have a repeatable description and steps involved in the acquisition of new skills and knowledge. Also, a way to engage all learned components to encourage and facilitate others.

4.2.5 Dev-CDAD-Prod Model

Based on the CDAC Model, the building of a Security DevOps structure was built dynamically to understand which main deliverables will help drive which standards. Also, what type of distributed governance is required during each SDLC development phase of a project.

4.3 Sub-models

Sub-models are importing and distinct part of explaining, supporting, and deep-diving into a more extensive or primary model.

Within our research, the four metric/learning models will allow teams to gather pertinent data. The sub-models, list below will allow diving deeper into how to use and harness the data rendered.

4.3.1 Mind map and Standards

Initially, standards are provided in a documentation format, and it is the terms of the developer to follow or not follow the requests.

While working with developers, the act of documentation standards, the developers wanted two things: 1) Clear concise outlines, and 2) visual analysis.

Radar Charts

As developers are searching for a comprehensive method to understand if they had planned for or met their governmental standards, we have built and introduced a Radar Chart to graphically display multivariate data by compiling data in the form of a two-dimensional chart.

The chart allows for quantitative variables represented on axes starting from the same point.

4.3.2 Scatter Charts

Utilizing the same data for the Radar Map, a DevOps Manager or IT Program Manager can utilize the chart data (XY chart) to show find out if there is a strong/weak relationship. The

chat will show data security levels that each team believes that they matched and can visually see if the team hit the mark/target set at the beginning of the build.

Each team can easily see and plot their data for each security area and understand if they hit or missed the mark.

4.3.3 Heat Maps

A heat map is a graphical representation of data where the individual values contained in a matrix, which are represented as colors. The usage of Heat Maps in our research will allow for a quick quantifiable view of data to focus quickly and accurately on a subsection of data. Analytics will allow us to deep dive into understanding the underlying data.

4.4 Data Analytics

Just as models and supporting sub-models are essential, the usage of data analytics help to inspect cleanse, and transform modeling data.

Amidst the data rendered, the goal of discovering useful information, and coming to conclusions, help support and defend decision-making.

Throughout our research, we have utilized two data modeling tools to help build and gather data.

The two tools that were instrumental in architecting, rendering, and data minding, all finding, are Microsoft Excel as well as Tableau Software.

4.4.1 Microsoft Excel

Excel was utilized to create presentable tables, allowing research data to be arranged via actionable tables (rows and columns). These data points helped yield mathematical data points, as well as utilize relevant graphs to view research data in other formats.

4.4.2 Tableau Software

Tableau is a data analysis/analytic tool that guides the transformation of data into actionable insights. The software grants a researcher to explore analytics, via workbooks and dashboards, to allow for ad hoc analyses.

Tableau will allow a decentralized team, easily share work with anyone, as well as easily make an impact on the company's business. From global enterprises to early-stage startups and small businesses, Tableau is a data analytics tool to help view and understand their data [14].

Chapter 5 Metrix and Learning Models

5.1 Introduction

Within this chapter, we deep dive into each Metrix, and the Model will be represented, explained, and visually shown. The overview will start with the purpose of the model, and a dive deeper into the understanding, as well as for instructions on how to utilize. Last will be actual screenshots of the tool, and the data analytics utilized within Excel and Tableau.

5.2 Agile Readiness Risk Grid

5.2.1 Purpose

The purpose of the Agile Readiness Risk Grid is to assess the agility health of an organization's team.

5.2.2 Overview

The Agile Readiness Risk Grid is to be utilized to help create, form, display, and instruct on how a Group Agile team is behaving and maturing. This tool is to be used to make any adjustments in an agile way and not to deter any production or solutions that may have developed.

If a team is a centralized team, this makes things more manageable, and all standards, goals, and deliverables are to follow one core process.

Microservice: As teams are requested to be decentralized, this tool is to help distribute standards between teams. The need to keep all team(s) (centralized or decentralized) in constant updates on team maturity.

5.2.3 Instructions

The Agile Readiness Risk Grid is designed to team dynamics and readiness and to are following and how mature and if any risks seen.

The instructions are as follows:

1. In the 'Team' sheet, please assign a rating in the 'Current Level' field based on the health levels that you as a developer feel you are currently at

- a. Hamper (0)
- b. In Transition (1)
- c. Supportable (2)
- d. Strong (3)
- e. Optimal (4)

2. Place the desired health level in the 'Target Level' field. The level will monitor and identify areas of improvement.

3. Place notes in the 'Comment' field to show what is needed and how to reach desired goals.

4. Once completed, review the 'Radar Chart'; the teams can check all security standards.

NOTE: (1) The **RED** line represents your maturity level. For optimal maturity, the RED line should reach the outer rim of the chart.

(2) The **BLUE** line represents the CURRENT level that the single distributed team believes they are nearest to. The BLUE line represents the current maturity snapshot where the team currently resides

5. Additional analytics are utilized from the data created from the tool.

5.2.4 Agile Readiness Risk Grid Instructions for Microservice

Agile Readiness Risk Grid Instructions for Microservice	
Overview:	The Agile Readiness Risk Grid is to be utilized to help create, form, display and instruct on how a Group Agile team is behaving and maturing. This tool is to be used to make any adjustment in an agile way, and not to deter and product or solution that is being developed. If team is a centralized team, this makes thing easier and all standards, goals, and deliverables are to follow one core process. Microservice: As team are requested to be decentralized, this tool is to help distribute standards between teams. The need to keep all team(s) (centralized or decentralized) in constant updates on team maturity.
Purpose:	The purpose of the Agile Readiness Risk Grid is to assess the agility health for an organizations team.
Instructions:	The Agile Readiness Risk Grid is designed to team dynamics and readiness and to are following and how mature and if any risk are being see . The instructions are as follows: <ol style="list-style-type: none"> 1. In the 'Team' sheet, assign a rating in the 'Current Level' field based on the health levels: <ol style="list-style-type: none"> a. Hamper (0) b. In Transition (1) c. Supportable (2) d. Strong (3) e. Optimal (4) 2. Place the desired health level in the 'Target Level' field. This will monitor and identify areas of improvement. 3. Place notes in the 'Comment' field to show what is needed and how to reach desired goals. 4. Once completed, review the 'Radar Chart' the teams can check all security standards. <p>NOTE: (1) The RED line represents your maturity level. For optimal maturity, the RED line should reach the outer rim of the chart. (2) The BLUE line represents the CURRENT level that the single distributed team believes they are at. The BLUE line represents the current maturity snapshot where the team currently resides</p>
Note:	Teams should evaluate their maturity level on end of each via each Sprint deliverable (be it Project end or Sprints).

Figure 5-1: Illustration: Agile Readiness Grid Instructions

5.2.5 Team Agile Risk Grid Worksheet

Within the Agile Readiness Risk Grid tool, the purpose is to allow decentralized teams to prepare and collaborate to ensure a proper team is developing strategies that are needed.

1) Section 1: Agile Groups and Areas

The worksheet is broken down into 6 'Agile Groups,' and each group will have a particular 'Agile Area' that coincides with the Group. This area will require a current level of preparation to understand the readiness of each. Data can be rolled up to view data at each group. The data is to show a picture of an agile team/activity readiness.

Please see below within 'Table 10', is a hierarchy of the break-out of these the Agile Groups, with associated Agile Area.

Table 10: Illustration: Agile Groups and Agile Areas

#	Agile Group	Agile Area
1	Team	
		Agile Methodology
		Teamwork
		Co-location
2	Project Initiation	
		Team size
		Dedicated team
		Colocation
3	Definition of Scope	
		Key Roles for Script
		SME and PO Non - Development team
		Definition of done
		Story size
		Backlog grooming
4	Agile Standards	
		Sprint Meetings (Grooming, Stand-up, Triage)
		Team/Project Retrospectives
		Requirements understood
		Risk Mitigation
5	QA Agility	
		Scope of testing
		Unit testing / Code Reviews (software)
		Automation (software)
6	CI/CD	
		Continuous Integration
		Continue Development

2) Section 2: Readiness Levels

Every Agile Area, within an Agile Group, will be assigned a rating in the 'Current Level' field based on the health levels or the below readiness levels.

Each readiness level (0-4), demonstrates, how prepared an Agile Area is. The end game is to share the updates and overview of each group, to allow for the team's view status.

In the next section, Section 3, each team will use the below to Readiness Levels to fill in and update, what current level and targeted maturity levels they want to ascertain.

Table 11: Agile Readiness Levels

Agile Area	Hamper (0)	In Transition (1)	Supportable (2)	Strong (3)	Optimal (4)
Agile Methodology	Not yet doing or being Agile.	Agile Methodology selected: - Scrum - Kanban - SAFe - XP	The team is comfortable with agile methodology. Any improvements needed to address, the team can speak up	The methodology can be utilized	Actively utilizing methodology, and able to decipher next moves
Teamwork	Non-existent	Improvements happening	Team able to navigate teammates and their expertise	Team >80% comfortable	Teams, be it centralized or decentralized, there is no coercion. Team is devoted

Colocation	Team members have very little proximity to each other.	Plans are in place to move team members as close to each other, as is currently feasible.	Most team members are accessible to any other team members	No issue with co-location as communication solutions are in place	No need to worry about the team, if internal or not.
Team size	The team selected but no dedicated team has been finalized or selected	Understand the need for more manageable teams are needed	The smaller team are started to get finalization, now that SME and technologies are finalized	Teams, be it centralized or decentralized are now are a manageable size <10	The ideal size of the team (developers, PO, Scrum Masters) are a finalized unit 8-10 teammates
Dedicated team	A team may be on multiple teams at this moment	The team are now ramping off older project but and more then >60% dedicated to this agile project	Enough team members are on a team to support process (but risks do occur)	>80% dedicated to this agile project	Most people are 100% allocated to the team. 'Bench' created or any possible resource movement/loss (PO, QA, Scrum)
Colocation	Team members have very little proximity to each other.	Plans are in place to move team members as close to each other, as is currently feasible.	Most team members are accessible to any other team members	Most team members sit within hearing distance of each other	Most team members are sitting in a team area together.

Key Roles for Script	No dedicated team	No dedicated team, but the team finalized.	The team does have a dedicated proxy to converse with, but the final team had not finalized	The team has full knowledge of project deliverable	Scrum Master, Product Owner, Developers and Quality Assurance, Stakeholders, are well defined
SME and PO Non-Development team	No one at this time	PO shared between multiple projects	The team is now intimately regulated with the team's product management needs.	Dedicated SME and PO now a part of the team and contributing to open questions, triage issues, and possible scope creep mitigation issues.	Quality rapport and alignment between the Product Owner and team are now moving well and can answer concerns, which are now answered by a dedicated PO.
Definition of done	Not definition	Each Scrum Team has its definitions, and there is no standardization between teams	Collaboration is happening between teams to understand the definition of DONE	Definition of Done has been shared between team, and acceptance if the definition has shared between Sprint Team	The assessment of Done has is final. All User Story's follow the same acceptance critical to make a story "DONE."

Story size	Not definition	T-Shirt / Sprint Poker is in progress, but no store sizing finalized	Developers are now able to break down Size and update User Stories	Epics and User Stories finalized and developers can break down requirements enough to complete deliverables w/in the allotted Script window	Stories now are finalized with the timing needed to progress w. Script. No questions.
Backlog grooming	No stories have groomed and ready, this means no deliverables can start within the next Sprint	Backlog grooming started but impeding within the current Sprint	>50% of stories groomed and ready for development	>80% of stories groomed and ready for development	Developers have ample stories groomed and ready for development.
Sprint Meetings (Grooming, Stand-up, Triage)	Not being held	Meetings held, timing or regularity not set yet	The team now ready for permeate meetings. All Scrum and project meetings have been finalized and defined. However, timing is not being followed as meeting are not mature	Team >80% comfortable with adoption	Adoption of meeting at regular intervals and timing kept

Team/Project Retrospectives	Not being held	Meeting are held, only, if there is enough time	Actions are being written down. However, actions/follow-up have not been prioritized and slated to be addressed.	The team understand the value of a retrospective and understand mitigation plan are needed to move FW to deliver a valuable solution.	A typical meeting, a part of any Sprint /Project deliverable
Requirements understood	Business Requirements are still needed finalizing by Bossiness SME/Owners	Scenarios and Business User cases designed off of original documentation. Developers can move FW, but scope creep may happen	Use Cases finalized. Developers can start to wring Agile Epics and User Stories and Sub Stories. Developers can start to enter into Agile Tracker/Agile SDL tool (example: JIRA)	Developers understand full Business requirements	Epics, User Stories, and Sub-Stories are now all mature and able to be fully worked on
Risk Mitigation	No tool/process currently used	All actual risks consideration within the group, no universal process created	Risk mitigation but no standards are finalized	Agile tools, such as the Distributed Security Standard Matrix, are being utilized and implemented	Tools and constant communication mitigation of risks stared between Scrum/Agile teams

Scope of testing	Timelines not defined	Testing not completed within allotted Sprint window (caused concern and pushes timelines)	Testing is on-going, but not all validation stories finalized	Manual testing is covered, and validation of stories are followed	Assessed tool (Appium, Selenium, UFT) has been used to help secure the timing of validation. To help close out a story to "DONE."
Unit testing / Code Reviews (software)	Not being used	Developers manually test coding	Definition and component testing are happening	All stories tested, either manual or by Tool (Junit, Spock, Cuppa)	All approaches to facilitate Unit testing finalized. Unit Testing has also been added as a User Story, within Sprint, for finalization
Automation (software)	Not being used / no Automated Tools have been dedicated to project	Automation does not keep in the allotted time. No standard allotted time.	50% Automation 50% Manual Testing	>70% of scripts fully Automated by a chosen/ Dedicated Tool	100% Utilization and coverage via the use of a dedicated Automation tool (Appium, Selenium, UFT)
Continuous Integration	Not implemented or standards finalized	No automation. All Integration completed by hand. Issues are hard to mitigate	Automation of CI implemented, either a homegrown solution or Product Tool No failure notifications. No continues to run!	Integration of code is now more frequent, and verification of issues handled, and notification will go out	Integration now a part of the system.

Continue Development	Not implemented	Set up, but manually run. Failures not fixed right away.	Automation of CD implemented, either a homegrown solution or Product Tool No failure notifications. No continues to run!	Release on demand is not feasible. Incremental changes (security, code, versioning) can now happen	Automated Business Automation software delivers speed and efficiency increase while errors drop.
-----------------------------	-----------------	--	--	---	--

3) Section 3: Team Readiness

As per each Agile area, that rolls up to each Agile Group, the below are the three driving factors, an analyst must fill in.

Within the Team Agile Risk Grid, the team will be required to fill in the following:

- Team's Current Level (0-4)
- Target Maturity Level
- Max Level

Incorporated in the above 3 data points, the analyst can select a Maturity Level that spans from Level 0 to Level 4.

- Zero is the lowest level, stating an Agile Area, within an Agile Group is not prepared (Hampering the process).
- Four is the highest level an Analyst can also choose, stating the process is ready and understandable (Optimal).

These maturity levels will allow the analysts to:

- 1) Select the Target Maturity in which the Group want to achieve (up to the level of 4)

2) Select the Team's Current Level, that the team is currently situated.

These levels are to help understand where each team's Agile readiness is. Is the team's Zero levels, hampering the development status, or are they are Optimal Level, of Four, in which the team is running on all cylinders.

The output is required now to be shared by each decentralized agile microservices team, to decipher the overall readiness. If there are significant inconsistencies, the team will need to mitigate and plan how to resolve it.

5.3 Data Analytics – Radar Map

5.3.1 Radar Map – Agile Readiness Risk Grid

Now that data has been entered, within the Agile Readiness Grid, the below radar report can now easily decipher, where the team is currently struggling and is at risk of not meeting the group's "Target Maturity Level."

The purpose of the Agile Readiness Risk Grid is to assess the agility health for an organization's team and process. To ask the question, "Is there Risk is seen within the Sprint team?"

5.3.2 Radar Map How to read

The map should allow for an effortless view where gaps are within standards as well what was the 1) initially requested maturity and 2) what is the maturity the group is currently seeing.

- (1) The **RED** line, on the Radar Chart, represents the client's maturity level. For optimal maturity, the RED line should reach the outer rim of the chart. This level is the level set before the building of the services and where the developer needs to deliver the security standards, in each distributed team.
- (2) The **BLUE** line represents the CURRENT level that the single distributed team believes they are currently situated. The blue line represents the current maturity snapshot where the team currently resides.

NOTE: Should frequently run to determine risk, especially when there are decentralized teams.

5.3.4 Radar Map - Output

Within the Map, each team will select the maturity they would like to get a good overview of. Below is a visual outcome of such a map.

The map will have two sections, the Maturity, the client, is Targeting, and the Maturity, the team, would like to see.

As an example, please view Continuous Development, the team was requested to meet a maturity level of (3), and per map, they hit that target (3).

Another example is Teamwork. The maturity level of (4) signifies the current level of maturity is (3). Per the Readiness Levels within the Team Agile Risk Grid, the team was targeting an (4) Optimal rate that stated, “Teams, be it centralized or decentralized, there is no coercion. The team is devoted”. However, the current level analyses, states that the Readiness level is at a (3) Serviceable Level, in which the Teamwork is: Team >80% comfortable.

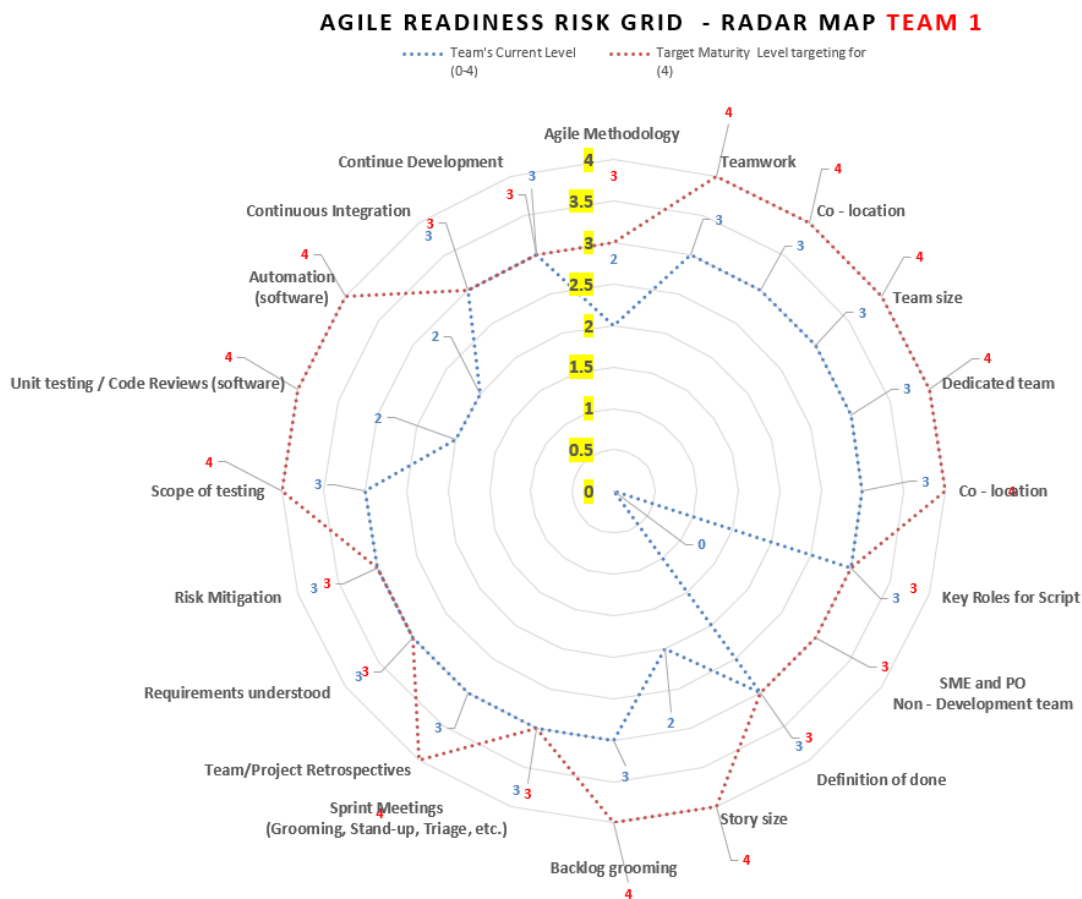


Figure 5-2: Agile Readiness Risk Grid – Radar Map

5.3.5 Radar Map – End Goal

Incongruent with agility, the need for risk mitigation and readiness needs to be a factor within the context of a project. The usage of the Radar Map is to all a quick look into the maturity the team wants to see and how it ‘measures’ up.

The end goal is to share analytics and risks with the rest of each decentralized team to help distribute knowledge and mitigate any risks.

5.3.6 Agile Readiness Scatter Charts

Another example to understand current Level chosen vs. Targeted Maturity, is represented with the below Scatter chart. Chart easily represents data leased, by the team, within the agile group, which was provided from the Agile Readiness Team analytics tool.

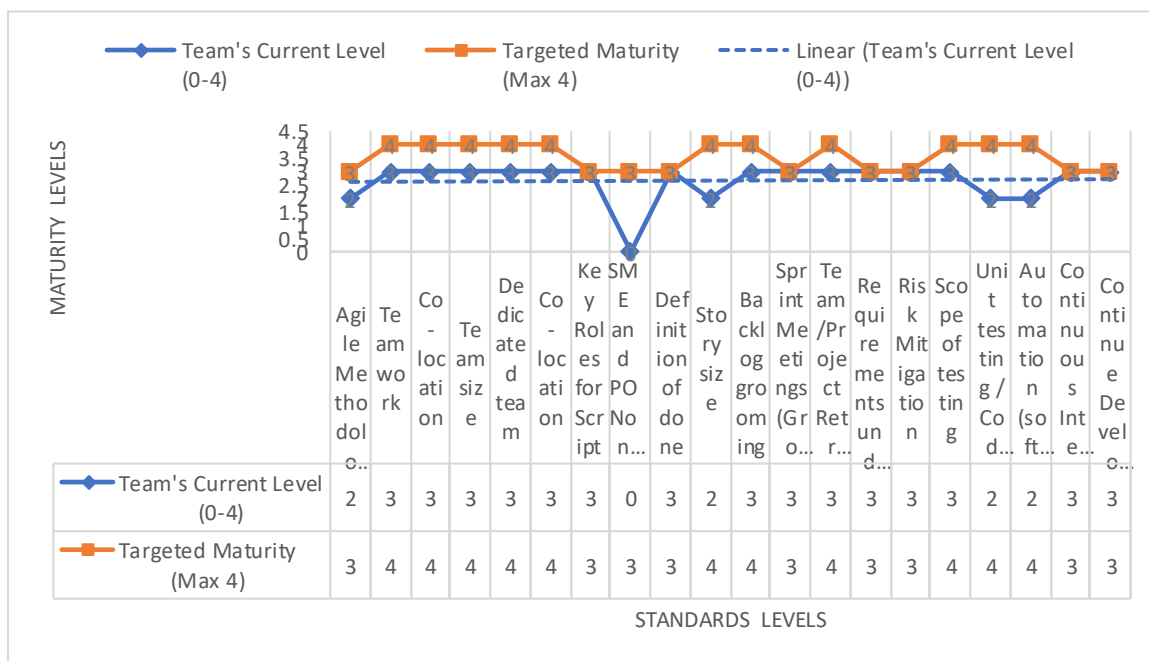


Figure 5-3: Agile Readiness Risk Grid – Radar Map

5.4 Tableau – Decentralized Team Overview

5.4.1 Introduction

Per the discussion in Chapter 4, the usage of analytics to view, gather, and mine data is the key to understanding the correct output of data. As per the Radar Map, created in Microsoft Excel, there are limitations to the analytics of data. Within this section, we will dive in to the usage of another analytical software called Tableau. This software will allow us to break down data further and slice the analytics to see the importance.

5.4.2 Dimensionality

In order to use such analytical tools, as Tableau, the developer will need to create and define data into dimensionality. Please see the below data, that was imported via an Excel data source, to form Dimensions and Measures within the Tableau Software.

Agile Group	Agile Area	Team's Current Le...	Team	State	City
Team	Agile Methodology	2	Team 1	Connecticut	Stamford
Team	Teamwork	3	Team 1	Connecticut	Stamford
Team	Co - location	3	Team 1	Connecticut	Stamford
Project Initiation	Team size	3	Team 1	Connecticut	Stamford
Project Initiation	Dedicated team	3	Team 1	Connecticut	Stamford
Project Initiation	Co - location	3	Team 1	Connecticut	Stamford
Definition of Scope	Key Roles for Script	3	Team 1	Connecticut	Stamford
Definition of Scope	SME and PO Non - Dev...	0	Team 1	Connecticut	Stamford

Figure 5-4: Agile Readiness Risk Grid – Radar Map

Each dimension, with Tableau, needs to be set to a Row/Column, in order to build the table with Tableau.

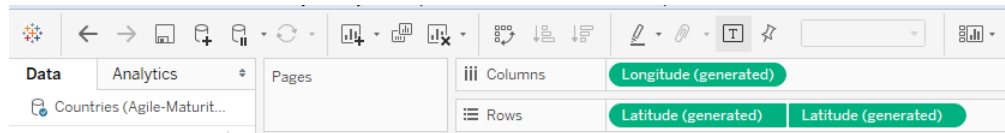


Figure 5-5: Tableau Columns and Rows for Agile Readiness Risk Grid

5.4.3 Filters/Marks

With the finalization of data-dimensionality, the additional usage of filters is now essential to help restrict the number of records the teams would like to have present in the data set. The filters are based on the conditions that the teams need to provide. Various types of filters used in Tableau are extract filters, data source filters, context filters, dimension filters, and measure filters [16].

Each dimension, with Tableau, can then be filtered or marked to view specific data represented from the directionality, below is a screenshot from Tableau, to filter teams and set the data marks for color and detailed info that needs to be visually displayed.

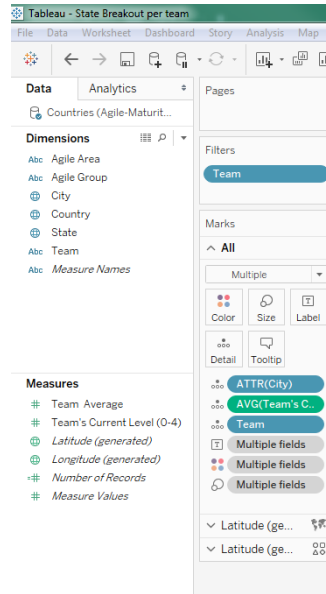


Figure 5-6: Tableau Filters and Marks for Agile Readiness Risk Grid

5.4.4 Distributed Team Agility Readiness Map

With dimensionality, filters, and data markers set, the outcome that's produced was a graphical view of the current Agility Readiness of 4 decentralized development teams.

The data simulates each team's current level of risk per the average of each team's 'Current Level of Readiness.' The map below is a very easy pictorial to view each decentralized team's latest risk.

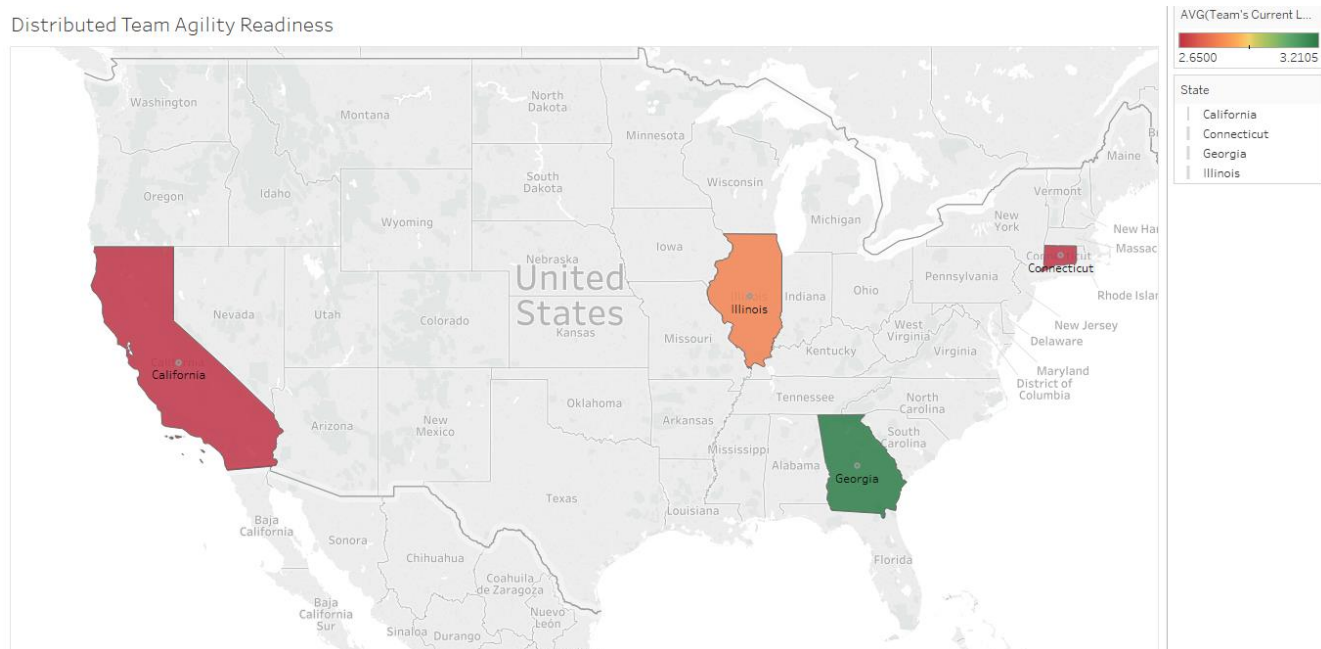


Figure 5-7: Tableau Distributed Team Agility Readiness Map

Tableau also allows the analytical user to deep-dive further into data. Please take the above Table 9 output. In order to view the overall updated, for example, in California, the analyst can ‘hover’ over the California state icon and will be able to view the following:



Figure 5-8: Data Mining information within California

The output given allows for each team to understand exclusive data about each region and distributed team. Example, Figure 5-8, presents Team3, via Mesa, California, which is at a 2.65% out of the maximum 4% a team can reach.

The analytics will allow for more in-depth conversation and analytics to mitigate the ‘why’ and what is causing these risks within the teams.

5.4.5 Agile Readiness Risk Grid – Conclusion

To make sure teams keep up with agility, and to truly follow a decentralized microservices development efforts, all team of developers needs to work within a loosely coupled team. However, to allow for the foundation of building strong, agile/decentralized teams, throughout builds, teams need to make user they are aligned with each other. If not aligned, have the tool provide a health check to analyze the issues seen, and to mitigate any risks quickly.

5.5 Distributed Security Standard Matrix Tool

5.5.1 Purpose

The purpose of the Distributed Security Standard Matrix Tool is to assess the security health for each decentralized Microservice team of developers. The Distributed Security Standard Matrix Tool is shared between teams or the DevOps Manager to provides updates of the health for the entire team.

5.5.2 Overview

Distributed Security Standard Matrix Tool Instructions can be used to set transformation goals, monitor progress, and get the team in cohesion regarding agile development. The instructions include Authentication Standards, Input and Output Standards, Logging

assessments. This tool is used in many other creative ways, such as to focus retrospectives and to help people at all levels do a self-assessment of their understanding of agile practices. The overview encourages self-paced learning and allows people the opportunity to learn from others that may have more agile experience

5.5.3 Instructions

The Distributed Security Standard Matrix Tool is designed to gauge security standards the DEV teams are following.

The instructions are as follows:

1. In the 'Team' sheet, please assign a rating in the 'Current Level' field based on the health levels that you as a developer feel you are currently at

- a. Hamper (0)
- b. In Transition (1)
- c. Supportable (2)
- d. Strong (3)
- e. Optimal (4)

2. Place the desired health level in the 'Target Level' field. 'Target Level' will monitor and identify areas of improvement.

3. Place notes in the 'Comment' field to show what is needed and how to reach desired goals.

4. Once completed, review the 'Radar Chart'; the teams can check all security standards.

NOTE: (1) The **RED** line represents your maturity level. For optimal maturity, the RED line should reach the outer rim of the chart.

(2) The **BLUE** line represents the CURRENT level that the single distributed team believes they are currently within. The BLUE line represents the current maturity snapshot where the team currently resides

5. Additional analytics utilize data created from the tool.

NOTE: Teams should evaluate their maturity level before and or after a new MS / API built.

5.5.4 Security Standard Matrix Tool

Distributed Security Standard Matrix Tool Instructions	
Overview:	Distributed Security Standard Matrix Tool Instructions can be used to set transformation goals, monitor progress, and get the team in cohesion regarding agile development. This includes: Authentication Standards, Input and Output Standards, Logging assessments . This tool can also be used many other creative ways, such as to focus retrospectives and to help people at all levels do a self-assessment of their own understanding of agile practices. This encourages self-paced learning and allows people the opportunity to learn from others that may have more agile experience.
Purpose:	The purpose of the Distributed Security Standard Matrix Tool is to assess the security health for each decentralized Microservice team, of developers. This tool can be shared between teams or the DevOps Manager to provide and update of the health for the entire team.
Instructions:	<p>The Distributed Security Standard Matrix Tool is designed to gauge security standards the DEV teams are following: The instructions are as follows:</p> <ol style="list-style-type: none"> 1. In the 'Team' sheet, assign a rating in the 'Current Level' field based on the health levels: <ol style="list-style-type: none"> a. Hamper (0) b. In Transition (1) c. Supportable (2) d. Strong (3) e. Optimal (4) 2. Place the desired health level in the 'Target Level' field. This will monitor and identify areas of improvement. 3. Place notes in the 'Comment' field to show what is needed and how to reach desired goals. 4. Once completed, review the 'Radar Chart' the teams can check all security standards. <p>NOTE: (1) The RED line represents your maturity level. For optimal maturity, the RED line should reach the outer rim of the chart. (2) The BLUE line represents the CURRENT level that the single distributed team believes they are at. The BLUE line represents the current maturity snapshot where the team currently resides.</p>
Note:	Teams should evaluate their maturity level before and/or after a new MS / API has been built

Figure 5-9: Security Standard Matrix Tool Instructions/Overview

5.5.4.1 Security Standard Matrix Tool Worksheet

Within the Security Standard Matrix Tool, the purpose is to allow decentralized teams to prepare and collaborate to ensure proper setup, understand any risks that are seen to help set up security standards.

4) Section 1: Security Group and Standards

The worksheet is broken down into 9 'Security Groups, and each group will have a particular 'Standard' that coincides with that group. This area will require a current level of preparation to understand the readiness of each. Data can be rolled up to view data at each group. Each group analysis is to show a picture of security readiness and standards via each team.

Please see below the hierarchy, in 'Table 12', of the break-out of these the Security Groups, with associated Standards.

Table 12: Security Group and Standards

#	Security Group	Standards
1	Authentication Standards	
		Standard Methods
		Logon
		Sensitive Data
2	Input	
		User Submitted Content
		Scrubbing user input
		Enforce HTTP Methods
3	Tokenization	
		Token Generation
		'TLL'/expire utilization
4	Output	
		Sensitive Data
		Headers
5	Encryption	
		Field Level
		Password Hash
6	Standard Return Errors	
		Bad Requests
		Unauthorized
7	Logging	

		Activities
		Threats
		Attempts
		Serialization
8	Tracing	
		Performance
		Bottlenecks
9	Monitoring	
		Database
		Users
		Data Collectors

5) Section 2: Security Standard Levels

All the nine Security Standard are assigned to a Security Group, which will be assigned a rating in the 'Current Level' field based on the health levels or the below readiness levels.

Each readiness level (0-4), demonstrates, how prepared a Security Group is. The end game is to share the updates and overview of each group, to allow for a team's view status.

In the next section, Section 3, each team will use the below Readiness Levels to fill in and update, what current level and targeted maturity levels they want to ascertain.

Table 13: Security Standard Levels

Security Standard	Hamper (0)	In Transition (1)	Supportable (2)	Serviceable (3)	Optimal (4)
Standard Methods	Standards have not set	Suitable Authentication discussion: OAuth2 JWT Password Storage	Quick validation (such as Postman testing) has occurred and ready to use for development	Validation tested - Authentication validity tested Security failure is phishing vulnerability tested	Setup and Receiving data Auth selected and developed upon
Logon	Understood requirement needed Not being used at this time	A standard method for authentication selected	Implemented Max Retry and Jail safety mechanisms to test vulnerabilities. Logon still need to be vetted	Validated and test	Encrypt Everything
Sensitive Data	Understood requirement needed Not being used at this time	Protect Sensitive Endpoints Make sure that all endpoints with access to sensitive data require authentication	universally unique identifiers (UUID) utilized to identify resources	Unintended operations are to test endpoints and backdoors	Fully implemented

User Submitted Content	Not setup	iframes utilized for partial development Need to check for external hosted JavaScript libraries, as these can cause issues	Content Security Policy is set up	Vulnerabilities such as Cross-side scripting and SQL injections have tested	Vulnerabilities can now be fully detected
Scrubbing user input	Not set up, requirements still being finalized	HTML tags, Java and or SQL statements recognized as possibly incorrect, incomplete, improperly formatted	Duplication of user input in progress	database setup and testing started	Data fully scrubbed
Enforce HTTP Methods	All simple CRUD methods finalized	All simple CRUD methods finalized	Method list finalized and listed, as well as HTTP status listed (200, 405, 201)	Vulnerabilities such as Cross-side scripting tested	Methods listed tested and finalized
Token Generation	Not understood and or not being done	Setup but not being used	Token and Passwords have been developed and ready for testing but not tested yet	Token and Passwords tested	Token and Passwords tested and implemented

'TLL'/expire utilization	Not understood and or not being done	Setup but not being used	Setup TTL but not active	Test for Fresh, Stale, and expiring content. Time to Live should now be set to govern the process. See if CDN (Content Delivery Networks) can be set up to optimization techniques to minimize page rendering time and improve user experience	Fully implemented
Sensitive Data	Not set up, requirements still being finalized	Data still open	Token and Passwords have been developed and ready for testing but not tested yet	Token and Passwords tested	Token and Passwords tested and implemented
Headers	Understood requirement needed Not being used at this time	REST headers and parameters finalized / size limitation as well	API testing to understand if developers uncover data for Authorization, Content type, and ad cache control	Can be used	Requests from Headers finalized

Field Level	DB not set up yes for and Field level encryption	Setup but not active	sensitive data (ex CC, SSN,) have been encrypted but still visible to developers	Sensitive data tested	All Field Level Data tested encrypted and stored
Password Hash	Not implemented	Password setup as plan text as of now for development purpose	Hashing is set up and tested for transformation	Password is stored in the database and fully render and tested	Password stored as a hash
Bad Requests	Not implemented	Identify the Order of the request HTTP Status Codes	Requests finalized and listed, and HTTP status listed - 204 No Content -400 Bad Request -406 Not Acceptable -500 Internal Server Error	Validated and test	The server takes responsibility for these error status codes.
Unauthorized	Not implemented	Identify the Order of the request HTTP Status Codes	Requests finalized and listed, and HTTP status listed - 401 Unauthorized	Validated and test	The server takes responsibility for these error status codes.

Activities	Not understood and or not being done	Understood requirement need Not being used at this time	Setup but not active	Setup and tested	Setup and Receiving data
Threats	Not understood and or not being done	Understood requirement need Not being used at this time	Setup but not active	Setup and tested	Setup and Receiving data
Attempts	Not understood and or not being done	Not implemented	Setup but not active	Setup and tested	Setup and Receiving data
Serialization	Not understood and or not being done	Understood requirement need Not being used at this time	Setup but not active	Setup and tested	Setup and Receiving data
Performance	Not understood and or not being done	Not implemented	Setup but not active	Setup and tested	Setup and Receiving data
Bottlenecks	Not understood and or not being done	Understood requirement need Not being used at this time	Setup but not active	Setup and tested	Setup and Receiving data
Database	Not understood and or not being done	Setup but not being used	Setup but not active	Setup and tested	Setup and Receiving data
Users	Not understood and or not being done	Setup but not being used	Users added to the system but not active	Setup and tested	Setup and Receiving data

Data Collectors	Not understood and or not being done	Not needed at this time	Setup but not active	Setup and tested	Setup and Receiving data
------------------------	--------------------------------------	-------------------------	----------------------	------------------	--------------------------

6) Section 3: Security Standard

As per each Security Standard, that rolls up to each Security Group, the below are the three driving factors, an analyst must fill in.

Within the Distributed Security Standard Matrix Tool, the team will be required to fill in the following:

- Team's Current Level (0-4)
- Target Maturity Level
- Max Level

Incorporated in the above 3 data points, the analyst can select a Maturity Level that spans from Level 0 to Level 4.

- Zero (0) is the lowest level, stating an Agile Area, within an Agile Group is not prepared (Hampering the process).
- Four (4) is the highest level an Analyst can also choose, stating the process is ready and understandable (Optimal).

These maturity levels will allow the analysts to:

1) Select the Target Maturity in which the Group want to achieve (up to the level of 4)

2) Select the Team's Current Level, that the team is currently with

These levels are to help understand where each team's Security Standard readiness is.

Are standards at a Zero level, hampering the development status, or are they are Optimal Level, of Four, in which the standards are following?

The output is required now to be shared by each decentralized agile microservices team, to decipher the overall security readiness and if standards are met. If there are significant inconsistencies, the team will need to mitigate and plan how to resolve it.

5.6 Data Analytics – Radar Map

5.6.1 Radar Map – Distributed Security Standard Matrix Tool

Now that data has been entered, within the Distributed Security Standard Matrix Tool, the below radar report can now easily decipher, where the team is currently struggling and is at risk of not meeting the group's "Target Maturity Level."

\The purpose of the Distributed Security Standard Matrix Tool is to assess the security health for each decentralized Microservice team of developers or to see the health of the entire team. To ask the question, "Is there Risk seen within each sprint development team?"

5.6.2 Radar Map How to read

The map should allow for an effortless look to see where gaps are within standards and what was the 1) initially requested maturity and 2) the current maturity group.

(2) The **RED** line, on the Radar Chart, represents the team's maturity level. For optimal maturity, the RED line should reach the outer rim of the chart. This level is the level set before the building of the services and where the developer needs to deliver the security standards, in each distributed team.

(3) The BLUE line represents the CURRENT level that the single distributed team believes they are currently within. The blue line represents the current maturity snapshot where the team currently resides.

NOTE: Should frequently run to determine risk, especially when there are decentralized teams.

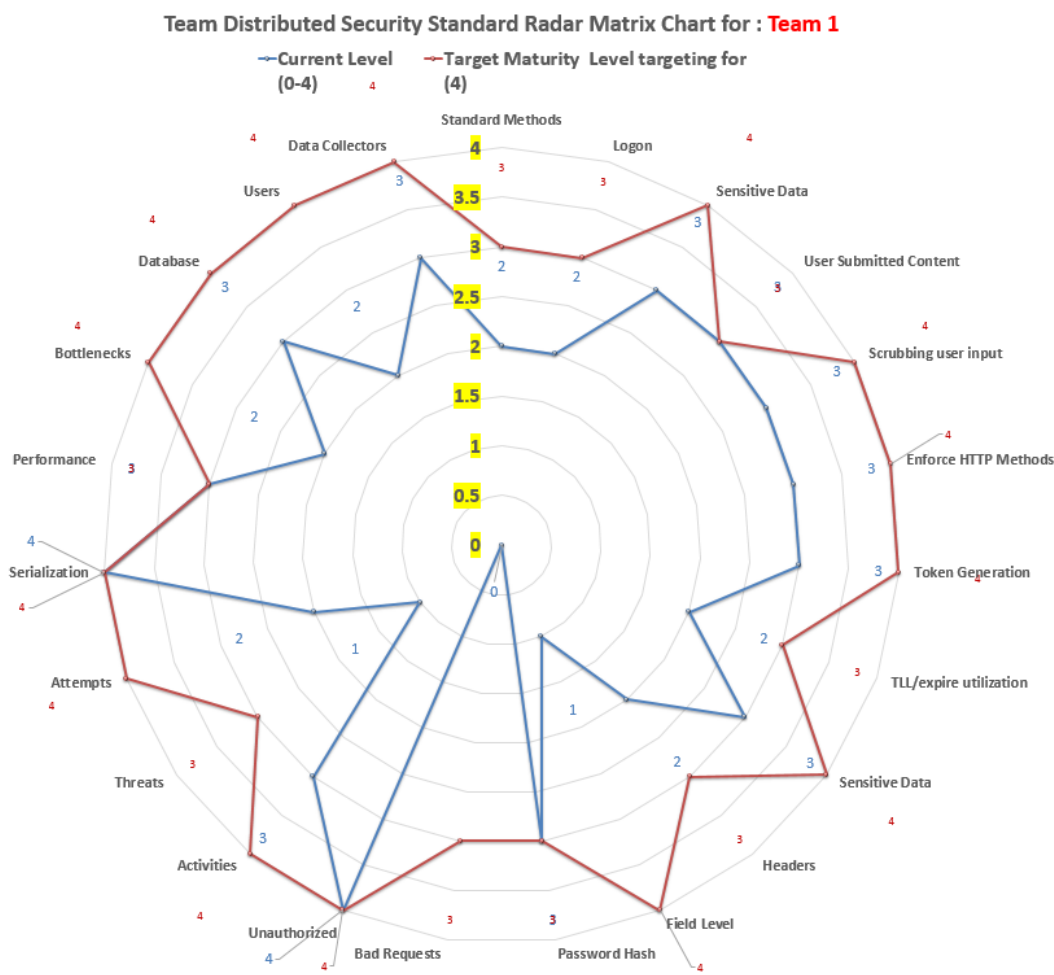


Figure 5-10: Team Distributed Security Standard Radar Metrix

5.6.3 Radar Map - Output

Within the Map, each team will select the maturity they would like to receive an overview within. The above, Figure 5-10' is a visual outcome of such a map.

NOTE: The map will have two sections, to understand the current security maturity standard the team is producing and the security maturity the team would like to see.

As an example, please view Performance Maturity per 'Figure 5-11'. The team has met the Performance maturity level of (3), and per map, the team hit that Performance target (3). Per the Security Standards Levels, the team is comfortable with current standards being "set up and tested." Below please see 'Figure 5-11' for the definition of Performance Security Standard (3).

Distributed Team Name: Team 1 of 4 Teams		Distributed Security Standard Matrix Tool Instructions				
Date of Report: 7/20/2019						
Group Security	Security Standard	Hamper (0) Informal	In Transition (1) Defined	Supportable (2) Used	Serviceable (3)	Optimal (4)
Loggi	Threats	Not understood and/or not being done	Understood requirement need Not being used at this time	Setup but not active	Setup and tested	Setup and Receiving data
	Attempts	Not understood and/or not being done	Not implemented	Setup but not active	Setup and tested	Setup and Receiving data
	Serialization	Not understood and/or not being done	Understood requirement need Not being used at this time	Setup but not active	Setup and tested	Setup and Receiving data
Tracing	Performance	Not understood and/or not being done	Not implemented	Setup but not active	Setup and tested	Setup and Receiving data
	Bottlenecks	Not understood and/or not being done	Understood requirement need Not being used at this time	Setup but not active	Setup and tested	Setup and Receiving data

Figure 5-11: Performance Maturity

5.6.4 Radar Map – End Goal

Incongruent with agility, the need for risk mitigation and readiness needs to be a factor within the context of a project. The usage of the Radar Map is to present all a quick look into the maturity the team wants to see and how it 'measures' up.

The end goal is to share analytics and risks with the rest of each decentralized team to help distribute knowledge and mitigate any risks.

5.7 Data Analytics – Risk Equivalency Metrix

The beauty of data is that it can be rendered and can easily be manipulated to render data to work for the developer/team. As an example, please see a quick slice of analytics measuring Risk Equivalency from the Distributed Security Standard Matrix Tool data set. The table below, via 'Figure 5-13: Risk Equivalency Metrix', displays output to easily view if a security standard, via a single team, is meeting their Targeted Assessment or not.

Group Security	AREA	Current Assessment (0-4)	Target Assessment	Equal to Goal Assessment Not Equal to Goal Assessment
Authentication Standards	Standard Methods	⇒ 2	⇒ 3	Not Equal to Assessment
	Logon	⇒ 2	⇒ 3	Not Equal to Assessment
	Sensitive Data	⇒ 3	↑ 4	Not Equal to Assessment
Input	User Submitted Content	⇒ 3	⇒ 3	Equal to Assessment
	Scrubbing user input	⇒ 3	↑ 4	Not Equal to Assessment
	Enforce HTTP Methods	⇒ 3	↑ 4	Not Equal to Assessment
Tokenization	Token Generation	⇒ 3	↑ 4	Not Equal to Assessment
	TLL/expire utilization	⇒ 2	⇒ 3	Not Equal to Assessment
Output	Sensitive Data	⇒ 3	↑ 4	Not Equal to Assessment
	Headers	⇒ 2	⇒ 3	Not Equal to Assessment
Encryption	Field Level	⇒ 1	↑ 4	Not Equal to Assessment
	Password Hash	⇒ 3	⇒ 3	Equal to Assessment
Standard Return Errors	Bad Requests	↓ 0	⇒ 3	Not Equal to Assessment
	Unauthorized	↑ 4	↑ 4	Equal to Assessment
	Activities	⇒ -	↑ -	Not Equal to Assessment

Figure 5-12: Risk Equivalency Metrix

This Metrix allows for

- Accessible output, to view if Team has met their selected Target Assessment, selected before starting the phase of development

Data deciphered via formatting listed below

- If **Red** – Not Met (Risk needs mitigation)

- If **White** – Assessment has been made, in which the teams have met their Security goals.

End Goal: Findings distributed via decentralized teams

5.8 Tableau – Decentralized Team Overview

5.8.1 Introduction

Per the discussion in Chapter 4, the usage of analytics to view, gather, and mine data is the key to understanding the exact output of data. As per the Radar Map and Risk Equivalency Matrix, created in Microsoft Excel, there are limitations to the analytics of data and its visualization.

Within this section, as was performed in Chapter 5 / Sub-chapter 5.4, we will again utilize Tableau analytic prowess to setup further analytics and outputs.

5.8.2 Agile Usage

The benefit of utilizing Tableau is to allow for more in-depth dive into data rendered, that has been entered within the Distributed Security Standard Matrix tool. Tableau will allow for data to utilize and to quickly gather data, as well as understand any inconsistencies. Last, the data shared with all the teams, allowing for agile mitigations.

5.8.3 Distributed Security Standard Matrix Reports - Tableau Reports/Dashboards

To perform ‘quick’ analytics and share the output with the decentralized teams, below Tableau reports are a reliable indicator that data can be used to present data to quickly analyzes needed data points.

The below reports developed for security governance reporting:

7) Stacked Bar and Heat Map Dashboard

Per description within section 4.3.4, a heat map is a graphical representation of data where the individual values contained in a matrix are characterized as colors. The usage of Heat Maps within Tableau will allow for a quick quantifiable view of data to focus quickly and accurately on a subsection of data. Analytics will allow the teams to deep dive into understanding the underlying data.

The end goal of the Heat Map analytics tool provides additional filtering and data manipulation for the greater definition of data and possible Risk Assessment and mitigation. Also, within the dashboard, the stacked bar graph is a chart that uses bars to show comparisons between categories of data, but with the ability to break down and compare parts of a whole. Each bar in the chart represents a whole, and segments in the bar represent different parts or categories of that whole [28].

The below example, within 'Figure 5-14, allows for a dashboard representation to all developers a view all Security Standards, within the below heat map, and allows for further data mining into within Security Group the standard below to and what is the risk. Note that filters have been added to quickly restrict the number of records present in the data set based on the given condition.



Figure 5-13: Heat/Stacked Bar Chart

8) Rating and Description Dashboard

The Rating and Description Dashboard had been built to elaborate and visually described the Information about each Security Standard, rolled up to its Security Group, and explained what Security Standard Levels, each has met. Please view Table 13: Illustration Security Standard Levels, for more info. For example, an analyst will be able to view

- What Security Standard Level and Description met; (0-4) Hamper-Optimal
- What Security Group and Standard reviewed

- Moreover, most importantly what is the full description of the Standard has been met

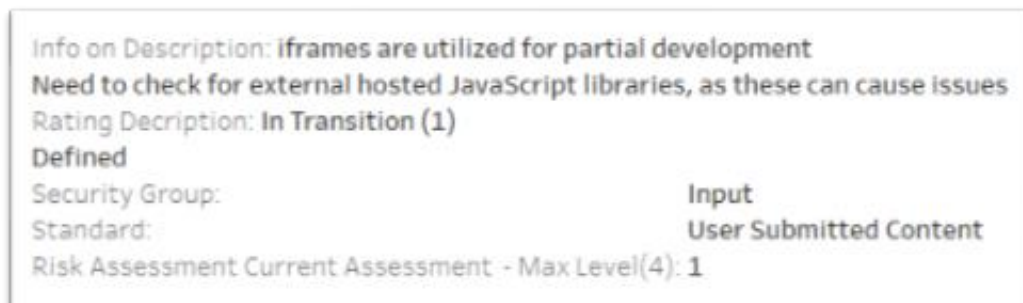


Figure 5-14: Mata Mining on Description - Rating and Description

5.8.4 Dimensionality

In order to use such data, shown in the above ‘Figure 5-15’, the below dimensionality within ‘Figure 5-16’ has been set up. The users create and define data into dimensionality. Dimensions are fields that allow an analyst to slice and further describe data records (e.g., names, dates, IDs, geographical info). The dimensionality will also measure which value fields allow the aggregation data (summed, averaged, and more).

Please see data within ‘Figure 5-16’, which was imported via a Microsoft Excel data source, to form Dimensions and Measures within the Tableau Software.

Security Group	Standard	Risk Assessm...	Rating	Description	Info on Description
Authentication Stand...	Standard Methods		2	Supportable (2)Used	Quick validation (such as Postman testing) has occurred and ...
Authentication Stand...	Logon		4	Optimal (4)	Encrypt Everything
Authentication Stand...	Sensitive Data		1	In Transition (1)Defined	Understood requirement needed Not being used at this time
Input	User Submitted Conte...		1	In Transition (1)Defined	Iframes are utilized for partial development Need to check fo...
Input	Scrubbing user input		3	Serviceable (3)	database setup and testing started
Input	Enforce HTTP Methods		1	In Transition (1)Defined	All simple CRUD methods have to been finalized
Tokenization	Token Generation		1	In Transition (1)Defined	Setup but not being used
Tokenization	TLL/expire utilization		2	Supportable (2)Used	Setup TTL but not active
Output	Sensitive Data		1	In Transition (1)Defined	Data still open
Output	Headers		2	Supportable (2)Used	API testing to understand if developers uncover data for: Aut...
Encryption	Field Level		3	Serviceable (3)	Sensitive data has been tested

Figure 5-15: Dimensions and data for Rating and Description Dashboard

Each dimension, with Tableau, can then be filtered or Marked to view specific data represented from the directionality, below is a screenshot from Tableau, to filter teams and set the data marks for color and detailed info that needs to be visually displayed.

The dimensionality of the row and columns make up the structure of the Rating and Description Dashboard. Please see Figure 5-16:

Columns	Security Group	Standard
Rows	AVG(Risk Assessmen..)	

Figure 5-16: Tableau Rows and Columns for Rating and Description Dashboard

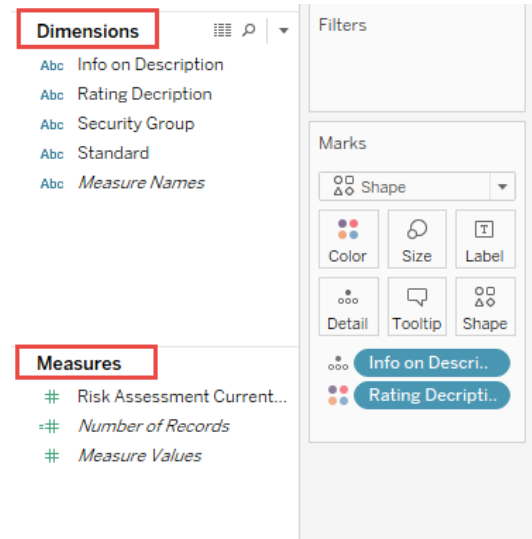


Figure 5-17: Dimensions and Marks for Rating and Description Dashboard

5.8.5 Rating and Description of Risk Dashboard

With dimensionality, filters, and data markers set, the outcome produces a graphical view of the current Ratings and Description of Risks for each current security standard. Each circle, below within 'Figure 5-19', is a pictorial representation of each Risk Assessment (Hamper -Optimal), for each Security and its associated Group

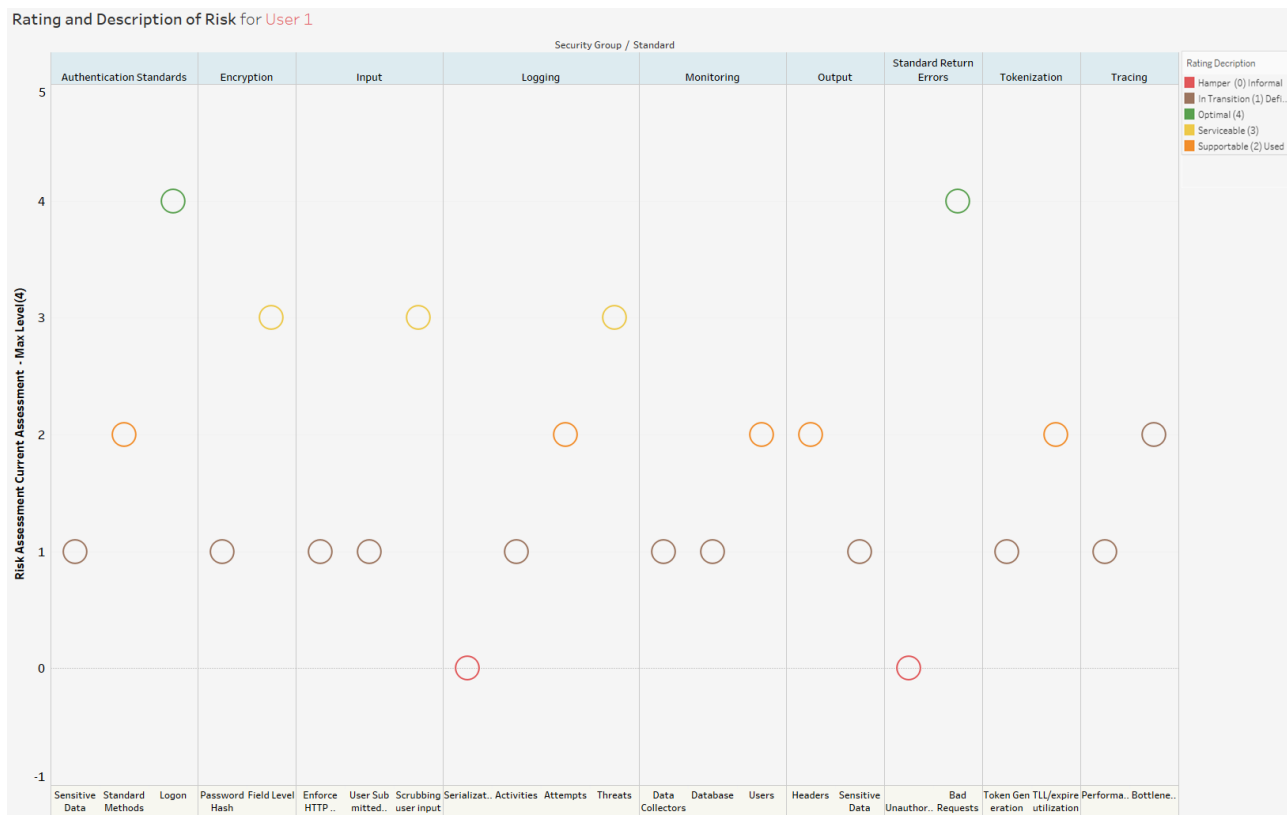


Figure 5-18: Rating and Description Risk Dashboard

As described in the below ‘Figure 5-20’, we see data mining feed from the data gathered from the above Rating and Description Risk Dashboard (in ‘Figure 5-19’). Below is a deeper dive and quick description of the current stage of each standard. 23 below.

Info on Description:	Encrypt Everything
Rating Description:	Optimal (4)
Security Group:	Authentication Standards
Standard:	Logon
Risk Assessment Current Assessment - Max Level(4):	4

Figure 5-19: Data Mining / Deeper Drive into Rating and Description Risk Dashboard

5.8.6 Conclusion - Rating and Description Dashboard

Ensuring teams have a clear description of each standard, the Rating and Description dashboard developed. Each team, within the decentralized team model, will have their dashboard to view and analyze all security standards and possible risks. The Rating and Description of Risk dashboard were created to ensure clarity and analysis.

5.9 Average Security Risk per Sprint Report

5.9.1 Introduction

Development efforts and risks allow for developers to identify and understand what standards they are following or what is the current status needs addressing.

Analysts can utilize this data to dive deeper into how or if these risks are hampering their development build, in terms of build time. As well as to better understand the Average Team Risk, via all developmental Sprint windows, the below dimensionality created.

5.9.2 Dimensionality

In order to view analyze data granularity, the data definitions of each dimensionality needed to be clarified.

As this report specifically measures development efforts, two units of development times need explaining.

- A Plan or PI signifies the length of time a plan or program will set. Each Program Increment is typically set to 6 two weeks sprints, allowing for a total of 3 months in which all developmental teams, will be assigned to build, test and deliver core functionality agreed upon at the beginning of the PI.

- A sprint (or iteration) is the basic unit of developmental times and effort, that is, restricted to a specific duration. The typical duration, within an agile microservice development, is that of a 2-week sprint window.

In this example, the time hierarchy has been broken down into from Year, Quarter, Months, and week. (NOTE: 2 weeks make up a Sprint)

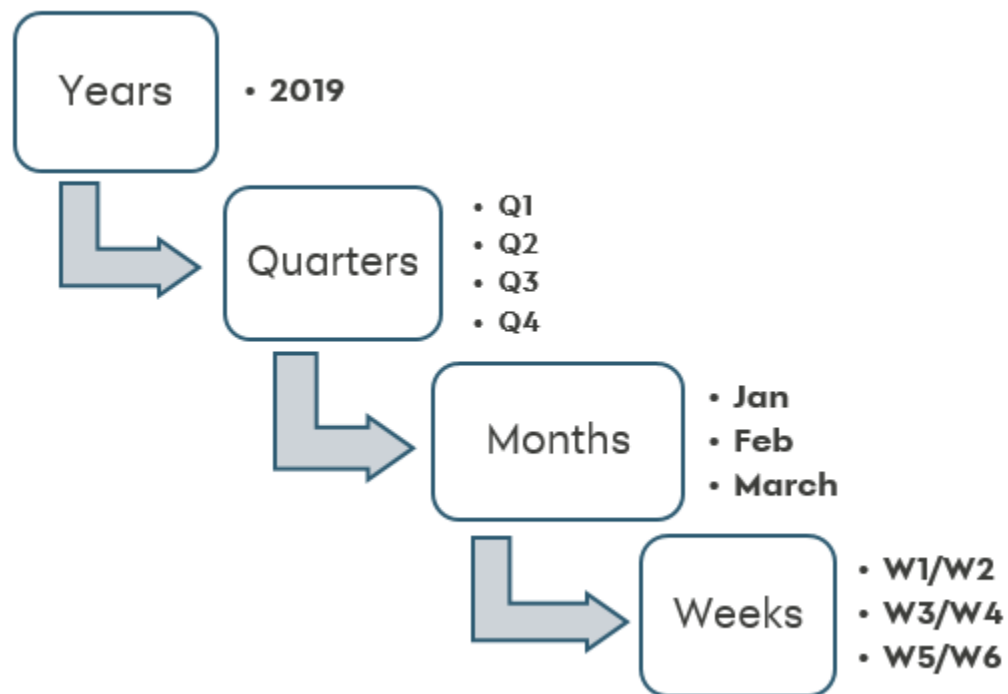


Figure 5-20: Hierarchy of Time set in the ‘Average Security Risk Per Sprint Report.’

‘Figure 5-22’, follows the same hierarchy as time; however, each sprint window coincides with a Program Increment.

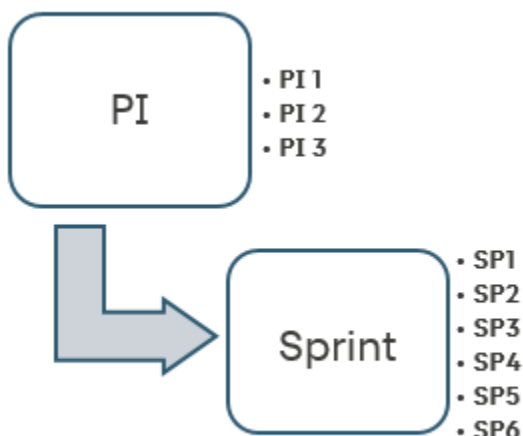


Figure 5-21: Hierarchy of Program Increment of ‘Average Security Risk Per Sprint Report’

Last, as per below ‘Figure 5-23’, each team will be represented within each Security Group, and the average risk per Quarter, per Sprint, per week, and will display the Rating Description for each,

Abc Predictive Security Groups	Abc Predictive Standards	# Predictive Team 1	Abc Predictive Rating Desc	Abc Predictive Sprints	Abc Predictive Plan	# Predictive Year	Abc Predictive Quarter	Abc Predictive Months	Abc Predictive Weeks
Authentication Stand...	Standard Methods	2	Good Standards	SP1	P1	2019	Q1	Jan	W1/W2
Authentication Stand...	Logon	3	Bad Standards	SP1	P1	2019	Q1	Jan	W1/W2
Authentication Stand...	Sensitive Data	1	Excelent Standads	SP1	P1	2019	Q1	Jan	W1/W2
Input	User Submitted Content	3	Good Input	SP1	P1	2019	Q1	Jan	W1/W2
Input	Scrubbing user input	4	Bad Input	SP1	P1	2019	Q1	Jan	W1/W2
Input	Enforce HTTP Methods	2	Excelent Input	SP1	P1	2019	Q1	Jan	W1/W2
Tokenization	Token Generation	3	Medium Tok	SP1	P1	2019	Q1	Jan	W1/W2
Tokenization	TLL/expire utilization	2	Low Tok	SP1	P1	2019	Q1	Jan	W1/W2
Output	Sensitive Data	3	Med Output	SP1	P1	2019	Q1	Jan	W1/W2
Output	Headers	0	Low Output	SP1	P1	2019	Q1	Jan	W1/W2
Encryption	Field Level	2	muedium Enc	SP1	P1	2019	Q1	Jan	W1/W2
Encryption	Password Hash	3	high emc	SP1	P1	2019	Q1	Jan	W1/W2

Figure 5-22: ‘Average Security Risk per Sprint Report’ – Dimensionality

Each circle within ‘Figure 5-24’, will provide the average security risk per Security Group.

For example, the Security Group called Monitoring (circled in Red), displayed in 'Figure 5-24', contains 3 Security Standards (Database, Users, Data Collectors). The 'Average Security Risk per Sprint Dashboard,' allows Team 1, to easily display all Security Group Averages for each Sprint.



Figure 5-23: Average Security Risk per Sprint Report

Analytics build to help team quantify, how much risk, is shown via a 1) a Security Group, 2) within a specific sprint-build window, and 3) what is the criticality.

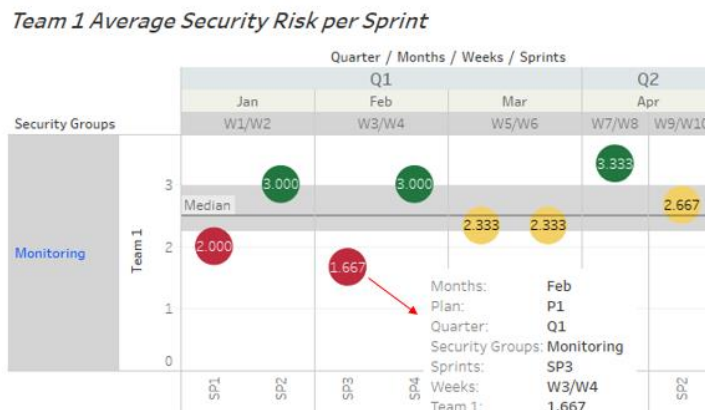


Figure 5-24: Data mining – Viewing data for Monitoring Security group’s risk per Sprint

5.10 Team – Overall Output Risk Assessment Reports

5.10.1 Introduction

All research and analytics have been to provide to represent a deep understanding of the decentralized teams and the knowledge in which each team needs to 1) understand coloration is importing and 2) and the need to analyze data for consistencies.

The team Overall Output Risk Assessment Reports are the final representation that will be built to provide a centralized team overview of all (4 Teams) and their assessment of each Standard.

Per the below ‘Figure 5-26’, the output of the report will again show visual annotations, that would be able to have data deeper diver into the description and output of the original report (please see ‘Figure 5-27’).

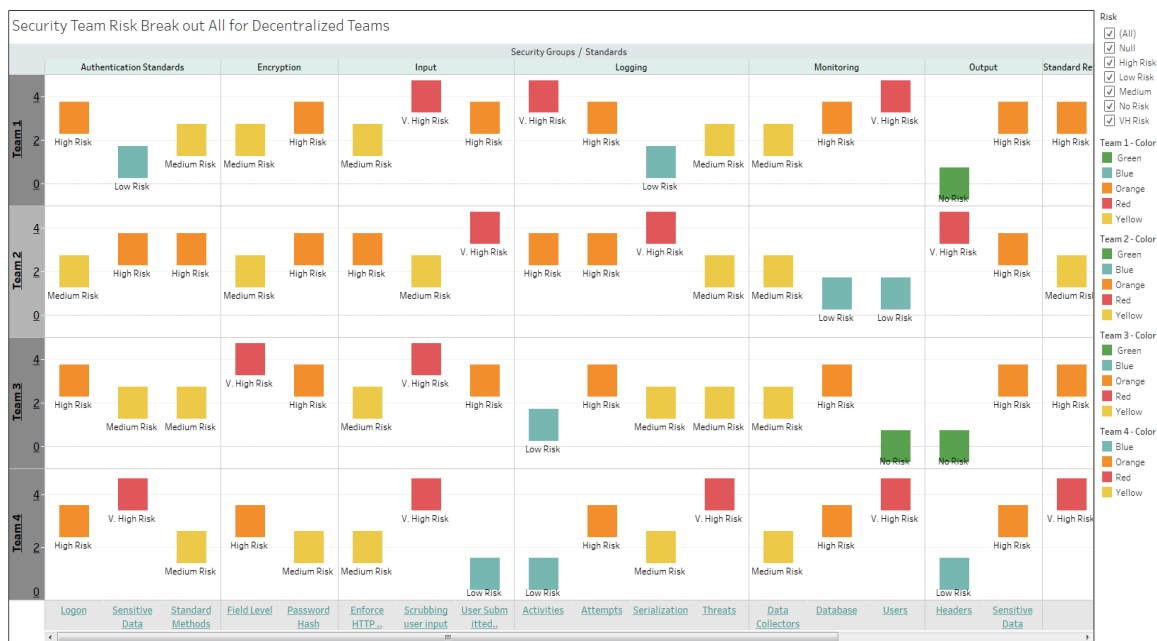


Figure 5-25: Overall Output Risk Assessment Reports

Data analytics and data mining will provide data, in which each team has currently found. These analytics can quickly glance at what each team, each Security Standard, and how it compares to other teams. The distribution of knowledge can be shared between each team to help mitigate setup and risks.

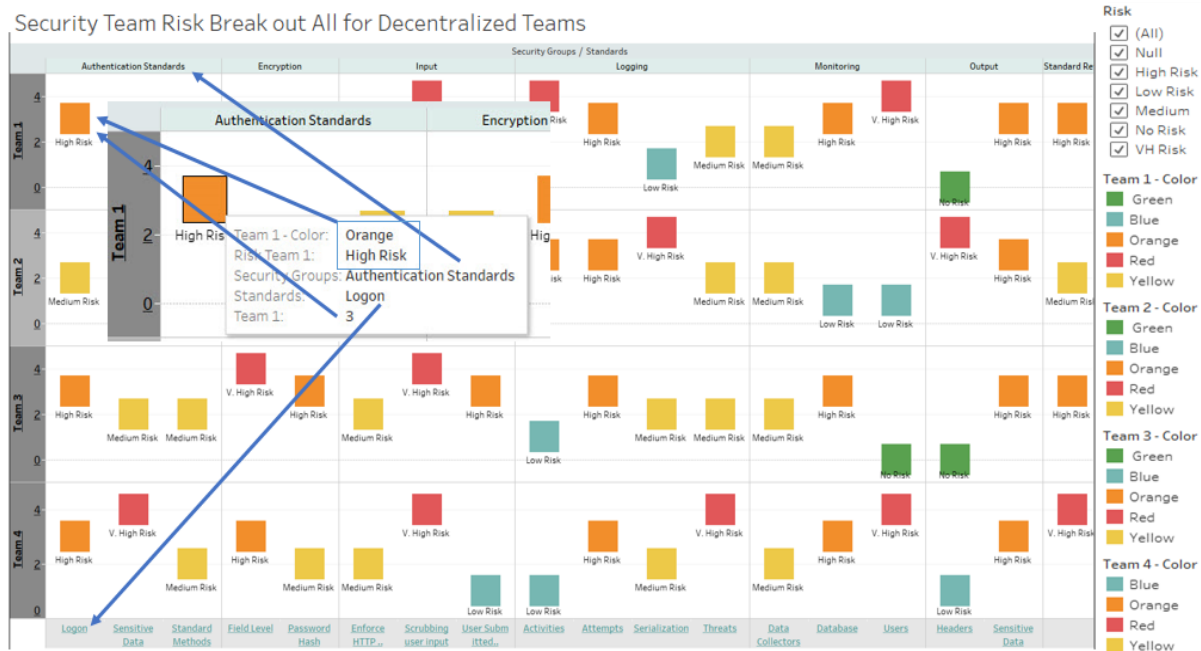


Figure 5-26: Data mining - Overall Output Risk Assessment Reports

The last example is the below analytics, ‘via Figure 5-28’, to show the segmentation of a specific Security Standard called ‘Logon.’ Within the Logon group, the visualization displays that all teams show a high-risk determination, but not Team 2.

The data will allow each distributed team, to share communication to understand 1) why they are behind 3) how Team 2 mitigated the risks that all other distributed teams are noticing, and or 3) what possible issues Team 1,3,4 have done performed incorrectly

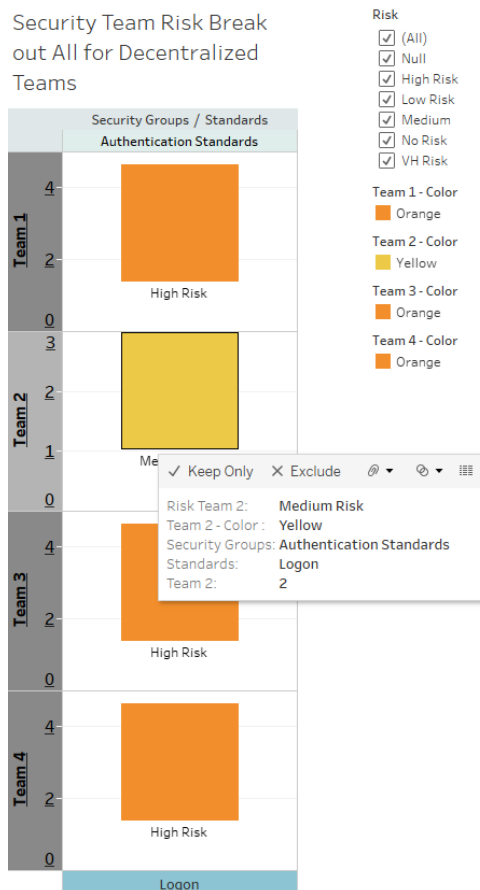


Figure 5-27: Deep-dive into analytics for ‘Logon’ Security Standard for all distributed teams

Chapter 6 Conceptual Models

6.1 Chapter Introductions

Data that has been gathered, via Reports, Metrics, and Dashboards, has culminated in the demand for communications between decentralized agile teams.

Via data gathered from initial surveys, in Chapter 4, as well as Reports, Metrics, and Dashboards gathered via Chapter 5, a mold/plan emerges.

The definition of a model is a reflection of the research questions, a framework of inquiry, including variables, and research designs developed as part of the grant activities.

Within this chapter, we will illustrate two conceptual models and how they tie together the data to explain the events of the research.

- 1) CDAD (Characterize, Diagnose, Anticipate, Distribute) Learning Model
- 2) Dev-CDAD-Prod Model

6.2 CDAD (*Characterize, Diagnose, Anticipate, Distribute*) Learning Model

6.2.1 Introduction

A composition of a model is to conceptually construct steps and standards to ensure events are positively understood and repeatable for a definitive outcome. The CDAD has been created to convey standards, and governance was followed and delivered. The models allow for each decentralized agile teams, to ensure standards and repeatable risks mitigated, and lessons learned.

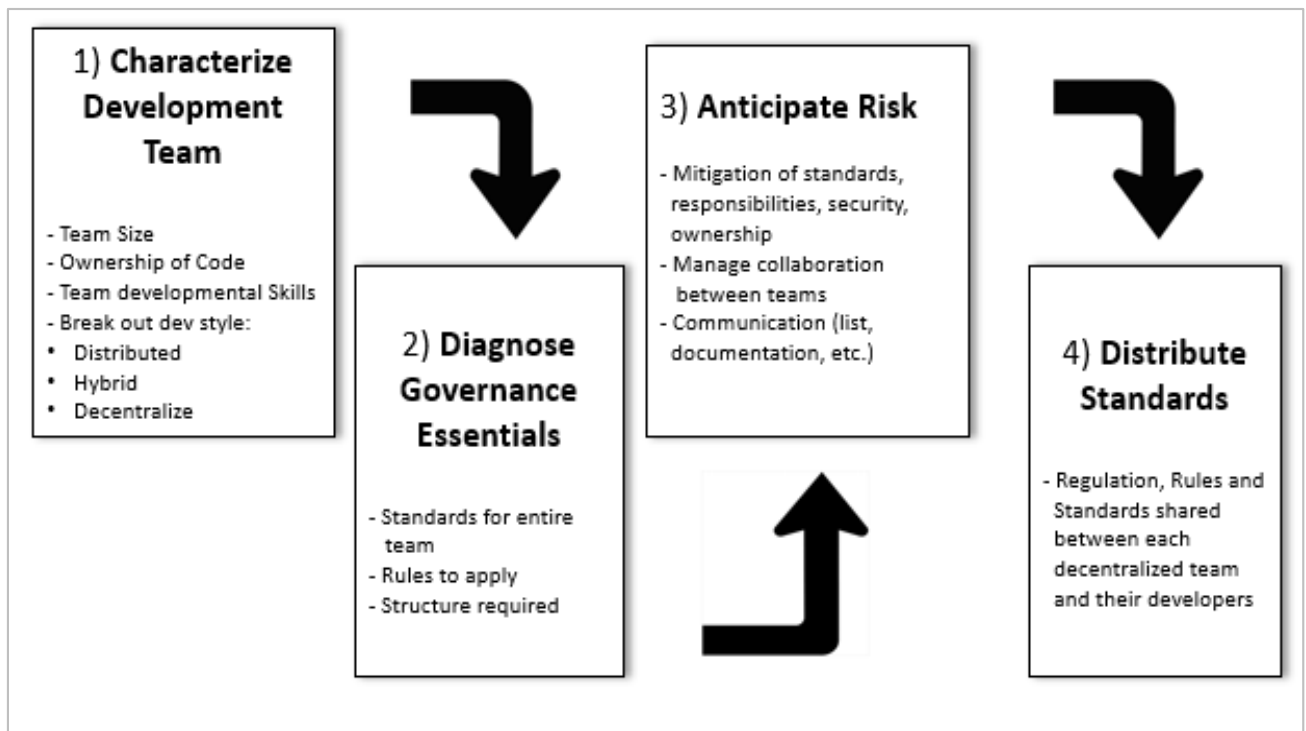


Figure 6-1: CDAD (Characterize, Diagnose, Anticipate, Distribute) Learning

The above CDAD Model, shown in ‘Figure 6-1’, is compiled to be utilized at the beginning, middle, and end of a project. The model is composed to deliver and recognize the need for standards but will not hurt the agility of a project.

Below is the breakout of the CDAD model:

1. Characterize the Development team to develop:

- a. Team Size
- b. Ownership of Code
- c. Team developmental Skills
- d. Break out dev style:

e. Distributed, Hybrid, Decentralize

2. **Diagnose Governance Essentials and create/finalize:**

a. Standards for entire team

b. Rules to apply

c. Structure required

3. **Anticipate Risk by preparing for:**

a. Mitigation of responsibilities, standards, security, ownership

b. Manage collaboration between teams

c. Communication (list, documentation)

4. Be ready to **Distribute Standards** that are created and to share with other teams for full distributed governance and overview:

a. Regulation, Rules, and Standards shared between each decentralized team and their developers

6.2.2 Conclusion

In a combination of each stage of the CDAC model, the characteristics allow developers and teams to follow an informative/repeatable representation, which ensures teams can characterize, diagnose, anticipate, distribute standards and risk within an agile process.

6.3 Dev-CDAD-Prod Model

6.3.1 Introduction

Based on the CDAC Model, the building of a Security DevOps structure, see Figure 30 below, was built dynamically to understand which main deliverables will help drive which standards. Also, what type of distributed governance is required during each SDLC development phase of a project.

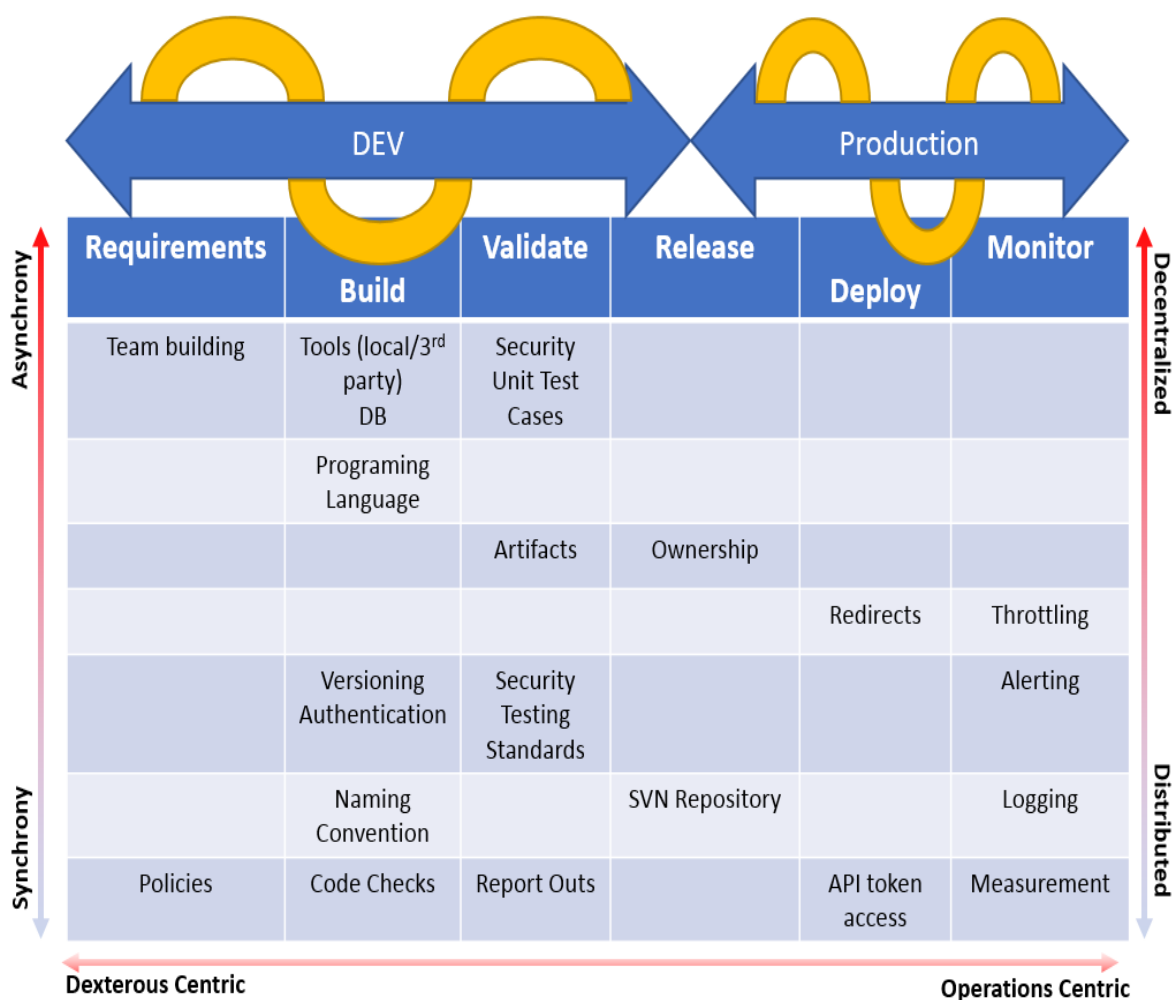


Figure 6-2: Dev-CDAD-Prod Model

6.3.2 CDAC Overview

Per the above Model (in Figure 6-2), each System Development Life Cycle (SDLC) development phase has been laid out in the following: Requirements, Build, Validate, Release, Deploy, and Monitor.

Per the research, each SDLC phase needs to be accountable for building specific standards and deliverables. The above DEV-CDAD-Prod model displays how both team building and security deliverables, fall into a distinct SDLC phase.

According to the Dev-CDAD-Prod model, the higher and left within the Model, the more decentralized a standard can be executed or ran asynchronously (nonparallel/serial). For example, building a team or selecting a programming language, per each decentralized team can be done with not a lot of standards/risks.

However, if the developers require to build policies, logging, and measurement standards, the model shows that the policies/access is vital for distribution and shared between the team synchronously (in parallel) with the same importance. These standards cannot be shifted or cannot be broken down.

6.3.4 Conclusion (Dev-CDAD-Prod)

The Dev-CDAD-Prod Model, based on the DevOps model, to ensure software development practices, combine software development and information technology operations to follow a development life cycle while delivering features, fixes, and updates frequently in close alignment with business objectives.

Within the Dev-CDAD-Prod Model, the standards follow from the initiation of projects (requirements) down to the Production standards (such as logging and monitoring). All

standards are importing to the life of an application/solution. Following these efforts allows for full transparency to follow an accurate distributed/decentralized model.

Chapter 7 Final Survey

7.1 Introduction

Within the final survey, the intent is gathered feedback from a range of participants, thought the organizations, to provide evidence that the matrix, tools, and models aided in the understanding of risks and the distribution of governance for all decentralized teams.

Table 14 displays the participants that a part of the final survey.

Table 14: Final Survey Participants and Roles

#	Participants	Role in Organization
1	Participant # 1	Technical Product Owner - API
2	Participant # 2	System Engineer
3	Participant # 3	Senior Security Solutions Architect
4	Participant # 4	Microservices Developers
5	Participant # 5	Microservices Developers
6	Participant # 6	Microservices Developers
7	Participant # 7	Microservices Developers
8	Participant # 8	Agile Coach / Educator
9	Participant # 9	Agile Coach / Educator

7.2 Exit Survey Questions and Answers

7.2.1 Exit Survey

The survey utilized the Likert scale approach as well as a few interviews, in which we gathered feedback. The Likert-type scale was used as it a widely used approach to scaling responses in survey research.

Below, please notice each matrix and tool has its Likert scale question. Below each set of questions is a report-out. The survey actively demonstrates, the usage of tools, helps aid in

the distribution of standards, teams' structure, and understand and governance needed.

The participants confirmed that the tools should be utilized via any point of the sprint development window, as the data gathered would be very useful for all teams.

Also, the usage of a data analytics tool, such as Tableau, helps to interpret further any data or risks that each decentralized team should know.

Table 15: Final Survey: Likert Scale Question #1

Distributed Security Standard Matrix Tool Survey Exit					
Q#1	Questions	Agree Strongly (1)	Agree (2)	Disagree (3)	Disagree Strongly (4)
1	Instructions for Metrix Tool easy to follow?				
2	This tool would help mitigate security risks for your teams?				
3	Does this tool help with agility?				
4	You would use this tool to help set up security standards? (before development starts)				
5	Would you use this tool to "check" status of security standards (during a Sprint Window)				
6	Would you use this tool to "check" status of security standards (at end of development Sprint Window)				
7	All Sprint Teams (decentralized) should utilize this tool?				
8	Is this tool easy to gather data to create analytics on standards?				
9	The usage of Analytical software (i.e. Tableau) assisted data rendered from matrix?				

Table 16: Distributed Security Standard Matrix Tool Survey Exit

Distributed Security Standard Matrix Tool Survey Exit									
Participants	Question 1	Question 2	Question 3	Question 4	Question 5	Question 6	Question 7	Question 8	Question 9
Technical Product Owner - API	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
System Engineer	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Senior Security Solutions Architect	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree	Agree Strongly	Agree Strongly	Agree	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Agile Coach / Educator	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree	Agree Strongly	Agree Strongly
Agile Coach / Educator	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly

7.2.2 Exit Survey Question #2 - Agile Readiness Risk Grid Survey Exit

Table 17: Final Survey: Likert Scale Question #2

Agile Readiness Risk Grid Survey Exit					
Q#2	Questions	Agree Strongly (1)	Agree (2)	Disagree (3)	Disagree Strongly (4)
1	Instructions for Metrix Tool easy to follow?				
2	This tool would help mitigate security risks for your teams?				
3	Does this tool help with team risk agility?				
4	You would utilize this tool to setup up teams and standards ? (before development starts)				
5	You would utilize this tool to setup up teams and standards ? (during a Sprint Window)				
6	You would utilize this tool to setup up teams and standards ? (at end of development Sprint Window)				
7	All Sprint Teams (decentralized) should utilize this tool?				
8	Is this tool easy to gather data to create analytics on standards?				
9	The usage of Analytical software (i.e. Tableau) assisted data rendered from matrix?				

Table 18: Agile Readiness Risk Grid Survey Exit

Agile Readiness Risk Grid Survey Exit									
Participants	Question 1	Question 2	Question 3	Question 4	Question 5	Question 6	Question 7	Question 8	Question 9
Technical Product Owner - API	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
System Engineer	Agree	Agree Strongly	Agree Strongly	Agree	Agree	Agree Strongly	Agree Strongly	Agree Strongly	Agree
Senior Security Solutions Architect	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Agile Coach / Educator	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Agile Coach / Educator	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly

Table 19: Final Survey: Likert Scale Question #3

CDAD Usage Survey Exit					
Q#3	Questions	Agree Strongly (1)	Agree (2)	Disagree (3)	Disagree Strongly (4)
1	Is this CDAD Model helpful ?				
2	Would you implement this model into you development methodology?				
3	Does the CDAC Model help bring to mind, team awareness?				
4	Does the CDAD Model help bring to mind, standardization essential within an agile methodology?				
5	Will the CDAD model be shared within your team's decentralized project?				

Table 20: CDAD Usage Survey Exit

CDAD Usage Survey Exit					
Participants	Question 1	Question 2	Question 3	Question 4	Question 5
Technical Product Owner - API	Agree Strongly	Agree	Agree Strongly	Agree Strongly	Agree Strongly
System Engineer	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Senior Security Solutions Architect	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Agile Coach / Educator	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Agile Coach / Educator	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly

Table 21: Final Survey: Likert Scale Question #4

DEV-CDAD-Prod Model					
Q#4	Questions	Agree Strongly (1)	Agree (2)	Disagree (3)	Disagree Strongly (4)
1	Is this CDAD Model helpful ?	X			
2	Would you implement this model into you development methodology?	X			
3	Does the CDAC Model help bring to mind, team awareness?	X			
4	Does the CDAD Model help bring to mind, standardization essential within an agile methodology?	X			
5	Would you add this model to your DevOps department?	X			

Table 22: DEV-CDAD-Prod Usage Survey Exit

DEV-CDAD-Prod Usage Survey Exit					
Participants	Question 1	Question 2	Question 3	Question 4	Question 5
Technical Product Owner - API	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
System Engineer	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Senior Security Solutions Architect	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Agile Coach / Educator	Agree Strongly	Agree	Agree Strongly	Agree Strongly	Agree
Agile Coach / Educator	Agree Strongly	Agree	Agree Strongly	Agree Strongly	Agree

Table 23: Final Survey: Likert Scale Question #5

Overall Research - Exit Survey					
Q#5	Questions	Agree Strongly (1)	Agree (2)	Disagree (3)	Disagree Strongly (4)
1	Microservice developmental teams require better communication and risk mitigation to successfully launch a project?				
2	A hybrid approach of distributing standard governance to agile decentral teams is a must need?				
3	Utilizing analytics helps garner strength for data gathered?				
4	Having Standard Risk Check Points between developmental Sprints and Pis(Program Increment), are valuable to align with agile principals ?				
5	Will models and uniform evaluations help streamline distributed teams?				

Table 24: Overall Research - Exit Survey

Overall Research - Exit Survey					
Participants	Question 1	Question 2	Question 3	Question 4	Question 5
Technical Product Owner - API	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
System Engineer	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Senior Security Solutions Architect	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Microservices Developers	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Agile Coach / Educator	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly
Agile Coach / Educator	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly	Agree Strongly

7.2.3 Exit Interview – Feedback

When reviewing the responses, from the initial survey and exit survey, an interesting finding becomes identified. All respondents had a strong opinion on the benefits that tools and models do help to guide and aid in the communication of standards via decentralized microservice teams.

To round up the research and the survey data, interviews conducted between three participants, help to garner more insight. These participants were not originally a part of any survey or conversions. The data gathered from the below three participants help further garner the importance of distributions as well as models. All respondents agreed that the ‘Agile Team Readiness Grid’ and ‘Distributed Risk tool’ can be used within other projects, within their current companies.

Please see all feedback below within Figure 7-1 to 7-2:

The first form of feedback was from a VP, IT Financial Reporting – Development:

From: [REDACTED]
Sent: Wednesday, August 14, 2019, 8:01 AM
To: Dall, Zachary
Subject: RE: Feedback from Dissertation Conversation:

In my opinion, the models exhibit strong potential for usage in scaling enterprise DevOps at the company, particularly in the readiness assessment phase of evaluating our current state so we have a better idea of how far the journey will be to target state. One of the biggest hurdles to beginning that journey has a meaningful first step and the processes you have outlined could deliver that ‘Minimum Viable Product’ that creates the early win that overcomes the inertia and begins the necessary momentum.

I hope this helps good luck with the delivery!

[REDACTED]

VP, IT Financial Reporting – Development

Figure 7-1: Exit Interview - Feedback #1

The second form of feedback was from an SVP, Technology Leader – Treasury:

From: [REDACTED]
Sent: Thursday, August 15, 2019, 3:19 PM
To: Dall, Zachary
Subject: RE: Feedback from Dissertation Conversation:

Feedback:

- I think the model could be utilized in my experience. In the past, at past companies, we struggled with the wing to wing process.
- This model would allow us to manage the solution wing to wing and mitigate risk along the way in a systemic fashion.
- The model will allow teams to think about the entire lifecycle with an agile eye. When you think about today and the future using this model, it will allow you to see what you need to do today but build for tomorrow. It is important to have the solution/product roadmap in mind in its entirety.
- This model forces you to think about not only creation but also growth for today and tomorrow.

[REDACTED]

SVP, Technology Leader – Treasury

Figure 7-2: Exit Interview - Feedback #2

The third form of feedback was from a Development Subject Matter Expert:

From: [REDACTED]
Sent: Tuesday, August 13, 2019, 4:16 PM
To: Dall, Zachary
Subject: Some points and recommendation Distributed Governance

Distributed Governance ensures:

- Authentication and integrity of data are the most important cybersecurity requirements and were assessed to be critical for all types of interactions, including monitoring and control commands, to ensure that the data exchanged comes from known sources and has not been modified in transit.
- Authorization and non-repudiation are important to ensure that commands are authorized, executed as specified, and reported back accurately.
- Availability is critical since Microservice usually operate autonomously and can be preset to perform the functions that are intended to be achieved

So, considering the above critical matrices that were done within the study, it is recommended to be used and adopted for better design and implementation within Microservices agile projects.

Thanks & Regards,



SME Development - Deposits

Figure 7-3: Exit Interview - Feedback #3

Chapter 8 Conclusion and Future work

8.1 Conclusion

Throughout the paper and research, the focus was to provide a well-versed understanding of the critical need, via decentralized development teams, to maintain open communication, distribute standard, mitigate risks, all while following agile standards.

The research conducted was between two companies to gather pain-points noticed and how the use of tools was needed to strengthen governance and risks within their microservices decentralized development teams. From the research, the goal of decentralized development, allows the software engineering teams to solve development problems more efficiently, and teams can build; however, they seem fit.

The decentralization of teams is not harmful unless they do have a way to distribute needed standards prior, during, and after a microservices has been constructed. The usage of models and tools helped to strengthen the communication, risks, and any possible agile mitigation, throughout a microservice build.

Decentralization of teams can strive, via any development build, only if the distribution of governance and procedures are followed, within each decentralized team.

Additionally, it is essential to develop via an SDLC mindset, so that developers follow standards needed to be governed, and the decentralized components are built with the same governed structure.

Last, below are some pros and cons for Decentralized and Distribution via the research and data gathered:

Decentralized Pitfalls:

- No single-entry point where decisions can be discussed or finalized
- Strong possibilities communicated standards are lacking between other decentralized teams
- Decisions influence the decisions of all other decentralized teams
- No single team will have complete standards
- Decision (risks, mitigations, communication) between teams lacking

Distributed Windfall:

- Standardizations created to govern
- Standards provided need to be distributed to the ALL decentralized teams, to follow as guidelines
- No ambiguity means better standard, less question, quicker/secure deployment
- Learning is essential between decentralized teams

8.2 Future Work

The research, within this paper, gathered a multitude of data via surveys, to understand what development teams are not focusing on, what pain-points developers see as well as models and learning procedures that they felt were needed to help conduct better development between the decentralized team.

Because all data, within the paper, has been simulated data, future work can be to:

8.2.1 Manual Utilization

Manually utilize all matrix and tools, presented in this paper, to gather data from an ongoing/existing microservices development project.

Each decentralized 'team lead' will need to:

- 1) Complete and execute the Agile Team and Security Risk assessments
- 2) Utilize an analytic software tool, like Tableau, to compile and data mind all data gathered from each team via the tools
- 3) Distribute the output of each assessment with each decentralized team
- 4) Mitigate any observed risks
- 5) Utilize the CDAD and the Dev-CDAD-Prod Models to follow structure thought-out all SDLC phases and project iterations of the microservice project

8.2.2 Digitalize Utilization

Digitalize the Agile Team Matrix and Security Risk Grid assessment and follow the CDAD and the Dev-CDAD-Prod Models. Possibly build a front-end GUI, in which each decentralized team can add data, analyze data, and distribute to all microservice development team to analyze risks and governance procedures.

Appendix A Glossary

Team	Definition
Asynchrony	Asynchrony is the occurrence of events independent of the main program flow and ways to deal with such events.
Agile	Agile is an iterative approach to software development.
Application Programming Interface (i.e., API)	<ol style="list-style-type: none"> 1) It provides a way to connect computer software components. 2) Specifies how these different software components can interact with each other and enable content and data to share between components. 3) Software-to-software interface, not a user interface.
Bottom-up development	<ol style="list-style-type: none"> 1) Planning provided to the team members, i.e., the people who are actively working on the project, have a say in the project planning and decisions made collaboratively. It will improve team communication and team building, and also empowers the team members. 2) Progress is made by the composition of available elements, beginning with the primitive elements provided by the implementation language and ending when the desired program reached
CI/CD	Continuous Integration/ Continue Development
Dexterous Centric	Operations can perform within multiple approaches.

Decentralized Team	<p>1) In Agile, this follows the progression from development, testing, and IT teams morphing into smaller DevOps teams.</p> <p>2) Rather than forcing a uniform monoculture, decentralizing allows for different teams to pick their specifics within the scope of what the standards allow.</p> <p>3) The goal is to free a software engineer team to solve development problems independently more efficiently and with higher velocity.</p>
Developer	A team member who programs computers or designs the system to match the requirements of a systems analyst
DevOps	(Development and operations) is an enterprise software development team used manage the agile relationship between development and IT operations
Dimension	Are usually those fields that cannot aggregate; measures, as its name suggests, are those fields that can be measured, aggregated, or used for mathematical operations.
Distributed Governance	<p>1) Specification of principles and methods which enable scalable coordination for forming consensus and to legitimate decisions.</p> <p>2) The approach must be, a principal's dispersed governance /rules, within a workforce of various user needs, must be distributed to be effective and efficient.</p>
Hierarchy	Consists of different levels, each corresponding to a dimension attribute.
Filter	Filters help restricts the number of records present in the data set based on the given condition.
Learning Model	A learning model is a description of the mental and physical mechanisms that are involved in the acquisition of new skills and knowledge and how to engage those mechanisms to encourage and facilitate learning.
Likert Scale	Is a psychometric scale commonly involved in research that employs questionnaires

Mark	Marks provides Tableau users with control over how the data displayed in the view.
Microservice Architecture	MA is the style of architecture that defines and creates systems through the use of small independent and self-contained services. They are also aligned closely with business activities.
Microservice	<ol style="list-style-type: none"> 1) A style that structures an application as a collection of loosely coupled services, which implement business capabilities. 2) Enables the continuous delivery/deployment of large, complex applications. 3) Take more of a bottom-up development/ownership approach.
Model	It is the reflection of a research question, framework of inquiry, including variables, and research designs developed as part of the grant activities. A model is considered theoretical since social and learning theories inform the development.
Monolithic	<ol style="list-style-type: none"> 1) Typically a single-tiered software application in which the user interface and data access code combined into a single program from a single platform. 2) Self-contained, and independent from other computing applications. 3) A top-down mindset of requirements, testing, development, deployment.
Operation Centric	Workable viable solution with no need for change
PO (Program Manager)	Articulates a program's strategy and objectives and assesses how it will impact a business. They must define and oversee a list of dependent projects needed to reach the program's overall goals
Program Increment (PI)	Delivers incremental value in the form of working, tested software, and systems. NOTE: PI's are typically 8 – 12 weeks long.
SDLC (System Development Life Cycle)	Used in systems engineering, information systems, and software engineering to describe a process for planning, creating, testing, and deploying an information system.

(SOA) Service-oriented architecture	A style of software design where services are provided to the other components by application components, through a communication protocol over a network
(SME) Subject Matter Expert	An SME is a team member who has special skills or knowledge on a particular job or topic. SMEs are highly accessed by instructional designers to extract intelligence when developing courseware and learning programs.
Sprint	Time boxed iteration of a continuous development cycle Routine Sprints are 1-2 weeks long.
Survey	Market research surveys make it easy to get opinions from potential and existing customers, test concepts, measure brand awareness, and more.
Synchrony	Simultaneous action, development, or occurrence.
Tableau	Data Analytics visualization tool that allows for fast analytics, smart dashboards, and data mining and manipulation.
Top-down development	Progress made by defining the required elements in terms of more basic elements, beginning with the required program and ending when the implementation language reached.

References

- [1] AIM Consulting, “12 Best Practices for Distributed Development Teams Using Agile and Scrum Methodologies.” AIM Consulting. 17-July-2015
- [2] Avidan, Z & Otharsson, H, "Accelerating the Digital Journey from Legacy Systems to Modern Microservices." Edition 1-B, 2018
- [3] Arsov, K., “Microservices vs. SOA — Is There Any Difference at All?”. A Medium Corporation, 10-Nov-2017
- [4] Ayers, C., “8 Pros and Cons of Decentralization”. 23-Sept-2016
- [5] Baker, C., “Managing microservices: How to get big benefits from a small service architecture.” World Canada. 22-Sept-2016
- [6] Barry, D. & Dick, D., “Web Services, Service-Oriented Architectures, and Cloud Computing: The Savvy Manager's Guide (Second Edition).” Waltham: Morgan Kaufmann. 2013.
- [7] Bachar, H., “Top-down vs. bottom-up approaches: Which is right for you?.” Clarizen. 20-Dec-2016
- [8] Bass, Len; Weber, Ingo; Zhu, Liming, “DevOps: A Software Architect's Perspective.”, Addison-Wesley. 2015
- [9] Brockway, E., “What You Should Know About Tableau.” New Horizons. 6-Sept-2017
- [10] Cark, K., “Microservices, SOA, and APIs: Friends or enemies? A comparison of key integration and application architecture concepts for an evolving enterprise.” IBM. 21-Jan-2016
- [11] Cawthorne, J. “How to Apply Governance to Your Collaboration Tools” CMS LiveWire. 22, Oct 2018
- [12] Conway, M., “How Do Committees Invent?” Thompson Publications, Inc., April 1968
- [13] Faulkner, B., “The Pros & Cons of Decentralized Leadership.” WordPress. 10-Feb-2019
- [14] Fields, E & Rueter M., “Tableau for the Enterprise: An IT Overview.” Tableau.com. 2019
- [15] Fowler, M “Microservices - a definition of this new architectural term.” Marin Fowler.com. 25-March 2014

- [16] Harvey, C., “7 Technologies that Enable a Microservices Architecture.” InformationWeek. 15-June, 2017
- [17] Heffner, R. “Microservices Have An Important Role In The Future Of Solution Architecture.” Forrester. 16-July-2105
- [18] Guru, “Types of Filters in Tableau: Condition by Formula, Extract, Context.” Guru99. 27-March-2019
- [19] Joosten, T., “A National Research Model for Online Learning.” The University of Milwaukee. 27-Sept-2015
- [20] Kappagantula, S., “What Is Microservices – Introduction to Microservice Architecture.” Microservices Zone. 22-May, 2019
- [21] Könönen, H., “Microservices: Considerations before implementation.” Aalto University. 2018
- [22] Mone`, L., “Microservices vs. SOA – Know the Difference.” LeanIX. 2019
- [23] Mone`, L., “Microservices-What an Enterprise Architect needs to know,” Document Version 1.1 Lean IX. 2019
- [24] Morgan, G., “Images of an organization” Berrett-Koehler Publisher, San Francisco, CA. 1998
- [25] Nadareishvili I., Mitra R., McLarty M. & Amundsen, “Microservices Architecture. Aligning principals, practices and culture” O’Reilly Media, Sebastopol, C., June 2016
- [26] Learn Model. (n.d.). In *Alleydog.com's online glossary*. Retrieved
- [27] Roos, D., “How to Leverage an API for Conferencing,” HowStuffWorks. 2019
- [28] Shrivastava, S. & Date, H., “Distributed Agile Software Development: A Review” Journal of Computer science and Engineering, Volume 1, Issue 1, May 2010
- [29] Smolic, H., “Stacked Bar chart: Definition and Examples.” Businessq. 12-Feb 2017
- [30] Stine, M. “Migrating to Cloud-Native Application Architecture” O’Reilly Media, Sebastopol, CA, 2015
- [31] Taylor, T., “How microservices governance has evolved from SOA” TechTarget, 24 Oct 2017
- [32] Wittmer, P., “SOA vs. Microservices – Learn the Difference & Decide.” Tiempo Development. 21-Nov-2018
- [33] Zuker, A., “Project Management: Centralized vs. Decentralized Delivery,” “Project Connection. 2017