www.ijecs.in

International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 6 Issue 11 November 2017, Page No. 22861-22864
Index Copernicus value (2015): 58.10 DOI: 10.18535/ijecs/v6i11.07

Microservices Maturity Model

Dr. Gopala Krishna Behara[1], Tirumala Khandrika[2]
[1]Wipro Technologies,
Strategy Consulting and Architecture Group,
[2]Wipro Technologies,
Strategy Consulting and Architecture Group,

Abstract:
Microservices Maturity model helps the enterprise customers to know where they stand and what it would take them to do in order to become a matured Microservices based enterprise. In addition, this Maturity Model provides the guidance to teams that are already using Microservices on what could be the next improvement.

Keywords: Maturity Model, Microservices, Maturity Levels, Maturity Scorecard

1. Introduction

A Maturity Model is a structured collection of parameters that describe certain aspects of Maturity in an enterprise.

As the Microservices Architecture matures, predictability, process controls and effectiveness also increase. Development of the Enterprise wide Microservices Architecture is critical because it provides the rules and definition necessary for the integration of information and services at the operation level across the Enterprise boundaries.

There exists many different architecture definition approaches for implementing Microservices across Industry. A 71% of companies will intensify their usage of Microservices in 2017 [1]. Many of them are unique and specific to the needs of individual development teams. Many enterprises, without understanding, have been using an approach towards leveraging APIs that classified as Microservices.

Many organizations know that they need to improve their Business and IT alignment in order to successfully manage change, but do not know how. Such organizations typically either spend very little on process improvement, because they are unsure how best to proceed; or spend a lot, on a number of parallel and unfocused efforts, to little or no avail. Many organizations continue to use old monoliths and only create new components as Microservices. The main reason companies have slowdown in adopting Microservices is because of missing knowledge and lack of qualified people [2-3].

An evaluation of the Enterprise practices against the model — called an ‘‘assessment’’ — determines the level at which the enterprise currently stands. It indicates the enterprise maturity in the area concerned, and the practices on which the enterprise needs to focus in order to see the greatest improvement and the highest return on investment. As the

Microservices Architecture Matures the predictability, process controls and effectiveness increases.

Microservices Architecture progresses in Maturity will benefit enterprise the following:

- Focus on Business Need
- Agility, enterprises can introduce new products or enhancements into the market quickly
- Enhancing the customer experience
- Can be developed & deployed independently of other services
- Reduced dependency on key resources
- Improved accuracy in scheduling software development / implementation
- More efficient deployment of technology solutions
- Increased traceability

Based on our interactions with various enterprise clients, the following are the high level queries raised by them,

- What would it take to become a Matured Microservices Enterprise
- What is the roadmap for the Enterprise
- What is the Microservices Maturity Model and where do they stand
- What are the architectural foundation pre-requisites for the Microservices adoption
- What are the infrastructural pre-requisites for the Microservices

Many enterprises, without understanding have been using various strategies like, “leveraging APIs that classified as Microservices”, “Domain driven design”, “SOA the right Way” and “adoption of DevOps Architecture”. This demands a need for designing Microservices that adopted across the Enterprises.

All the above had prompted the authors to create a Microservices Maturity model, which helps the enterprise customers to know where they stand and what it would take them to do in order to become a matured Microservices based enterprise. In addition, this Maturity Model is to provide guidance to teams that are already using Microservices on what could be the next improvement.

This Article describes the steps towards assessing the Maturity of an Enterprise in a Microservices Architecture Stream covering the aspects of the Enterprise adoption of domain driven design, infrastructure for provisioning and adoption of DevOps.

2. Maturity Assessment

Microservices adoption needs some careful thoughts. A quick Maturity assessment will be helpful to understand the

Maturity of the enterprise in Microservices adoption and some of the challenges the enterprise can expect.

The main drivers for the Enterprise Microservices Architecture adoption are,

- Enterprise primary business revolves around extremely high volume transactions
- Rapid development and enhanced operational support, reduce implementation and release cycle time and enables continuous delivery
- Enterprises migrating from Monolithic and Legacy Processes to Microservices
- Scalability and flexibility of Cloud service help in handling more customers simultaneously without interrupting service
- For the solutions that require the integration of business capability from disparate systems and programming models
- For aligning business and IT functionality
- For business and technology agility
- Better ROI
- For incremental development (Agile) and continuous deployment of business software

In addition, development of the Enterprise wide Microservices adoption is an ongoing process and cannot delivered overnight. The establishment of a Microservices across Enterprise is critical because it provides the rules and definition necessary for the integration of information and services at the operation level across enterprise boundaries.

3. Microservices Maturity Assessment Framework

A Microservices Maturity Assessment framework comprises a maturity model with different maturity levels, a set of parameters that assessed methodology and a toolkit for assessment (questionnaires, tools, etc.). The outcome is a detailed assessment report, which describes the maturity of the enterprise as a whole, as well as the maturity against each of the Maturity assessment parameters.

The key components of this framework depicted below:

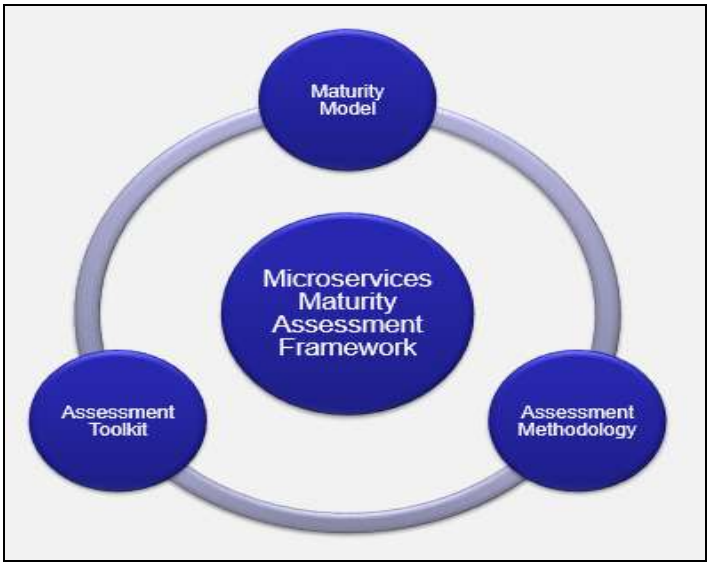


Figure 1 – Maturity Assessment Framework

Microservices Maturity Model (MSMM) helps to conduct the Maturity assessment across Enterprise. MSMM provides the framework that represents the key components of a productive Microservices architecture process. The goal is to enhance the overall success of enterprise Microservices adoption by identifying weak areas or examining the existing areas and suggesting changes/improvements to be fit for Microservices, providing a defined evolutionary path to improving the overall adoption process.

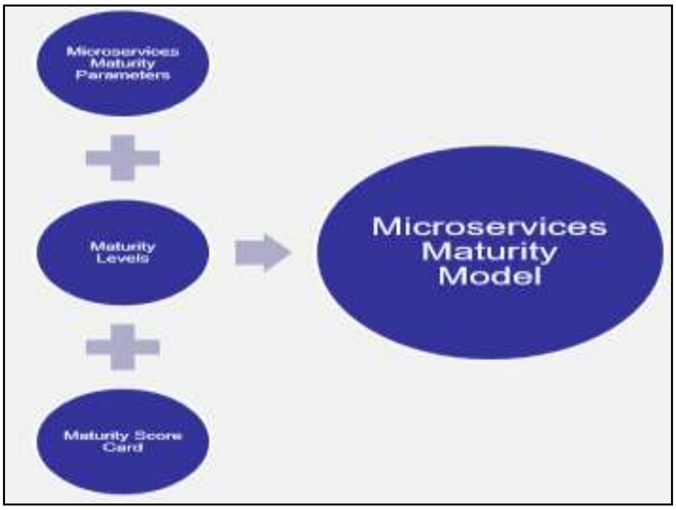

Microservices Maturity Parameters
Maturity Levels
Maturity Score Card
Microservices Maturity Model

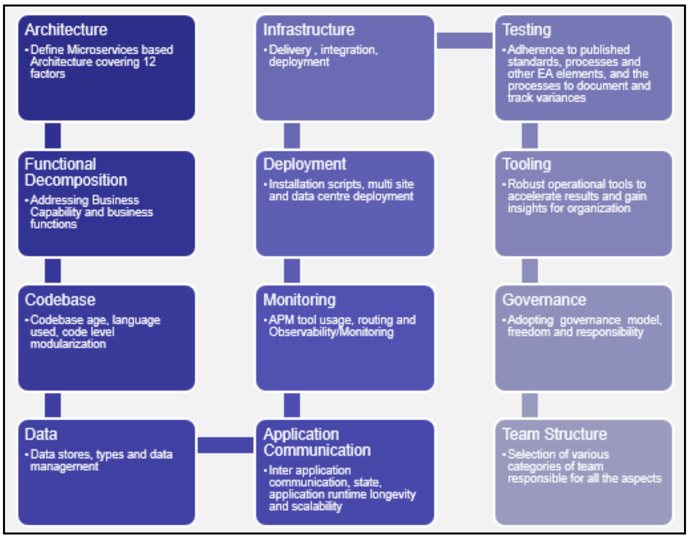
Figure 2 – Microservices Maturity Model

The MSMM comprises the following,

- MS Maturity Assessment Parameters
- Microservices Maturity Levels
- Maturity Score Card

4. Maturity Model Parameters

Based on our consulting experience in Microservices Maturity assessment and the industry standards, the following Maturity Parameters are used in assessment.



Architecture
- Define Microservices based Architecture covering 12 factors

Infrastructure
- Delivery , integration, deployment

Testing
- Adherence to published standards, processes and other EA elements, and the processes to document and track variances

Functional Decomposition
- Addressing Business Capability and business functions

Deployment
- Installation scripts, multi site and data centre deployment

Tooling
- Robust operational tools to accelerate results and gain insights for organization

Codebase
- Codebase age, language used, code level modularization

Monitoring
- APM tool usage, routing and Observability/Monitoring

Governance
- Adopting governance model, freedom and responsibility

Data
- Data stores, types and data management

Application Communication
- Inter application communication, state, application runtime longevity and scalability

Team Structure
- Selection of various categories of team responsible for all the aspects

Figure 3 – Microservices Maturity Parameters

A brief description of each parameter follows.

Microservices Maturity	Description

Parameters	
Architecture	This parameter covers Microservices based Architecture covering, 1) architectures and architecture disciplines, 2) architecture perspectives, 3) architecture requirements, 4) architecture means, 5) organizations and individuals and 6) architecture methods
Functional Decomposition	Addressing Business Capability and business functions, which helps to establish the use cases.
Codebase	This Parameter indicates Codebase age, language used, code level modularization. It also provides the typical code coupling and cohesion characteristics
Data	This parameter helps to assess Data stores, data store types and data management.
Infrastructure	It covers Infrastructure related to delivery , integration, deployment and governance parameters
Deployment	This parameter addresses deployment platforms, deployment fabric, deployment artifacts and deployment orchestration. Artifacts are typically, Installation scripts, multi-site and data centre deployment. Infrastructure mutability and infrastructure provisioning details verified.
Monitoring	This covers the APM tool usage, routing and Observability/Monitoring. Strong emphasis on detailed, end-to-end monitoring of production systems. Ability to detect and alert on system issues. Sufficient monitoring to be able to do remote runtime diagnosis
Application Communication	Addresses Inter application communication, state, application runtime longevity and scalability. Helps to verify application configuration, inter platform service discovery.
Testing	Verifies the level of testing, automated testing for all the environments, component based testing.
Tooling	Robust operational tools to accelerate results and gain insights for enterprise.
Governance	Architecture governance is the practice and orientation by which enterprise architectures and other architectures managed, controlled at an enterprise-level.
Team Structure	Type of teams necessary to adopt Micorservices at various levels of the enterprise. Team has freedom to select technology choices, through governed process and then responsible for its consequences.

5. Microservices Maturity Levels

The Microservices Maturity Model and its levels are depicted below. The model follows the path of an enterprise as their Micorservices adoption matures, and sets benchmarks to measure the performance and path that is a natural progression in the development of enterprise architecture. Each level contains statements that are indicative of a Microservices adoption Program at that level.

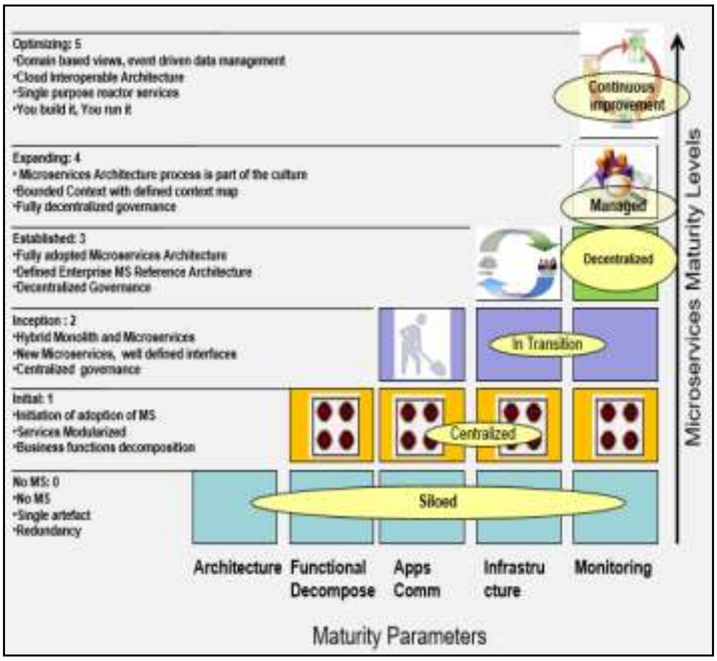

Optimizing: 5
•Domain based views, event driven data management
•Cloud Interoperable Architecture
•Single purpose reactor services
•You build it, You run it
Continuous improvement
Expanding: 4
• Microservices Architecture process is part of the culture
•Bounded Context with defined context map
•Fully decentralized governance
Managed
Established: 3
•Fully adopted Microservices Architecture
•Defined Enterprise MS Reference Architecture
•Decentralized Governance
Decentralized
Inception : 2
•Hybrid Monolith and Microservices
•New Microservices, well defined interfaces
•Centralized governance
In Transition
Initial: 1
•Initiation of adoption of MS
•Services Modularized
•Business functions decomposition
Centralized
No MS: 0
•No MS
•Single artefact
•Redundancy
Siloed
Architecture
Functional Decompose
Apps Comm
Infrastru cture
Monitoring
Maturity Parameters
Microservices Maturity Levels

Figure 4 – Microservices adoption Maturity Levels

Level	Description
Level 0: No Microservices	Enterprise still develop applications in a monolithic approach. There is no a documented Microservices in place at this level of maturity. While solutions are developed and implemented, done with no recognized standards or base practices.
Level 1: Initial	The base Microservices architecture framework and standards defined, typically performed informally.
Level 2: Inception	Define hybrid monolithic and Microservices. New Microservices were developed. Manage Governance Centrally
Level 3: Established	Define enterprise Microservices Reference Architecture using approved standard and/or customized versions of the templates. Initiate decentralized governance
Level 4: Expanding	Collect Microservices performance metrics, analyze and acted up on. Metrics used to predict the performance and provide better understanding of the Microservices across the enterprise. Fully decentralized governance.
Level 5: Optimizing	The Microservices architecture processes are mature. Domain based views, event driven data management are established. There are ongoing refinements and improvements based on the understanding of the impact changes have to these processes.

6. Maturity Assessment Methodology

The following diagram depicts the Microservices Maturity assessment approach used at Enterprise:

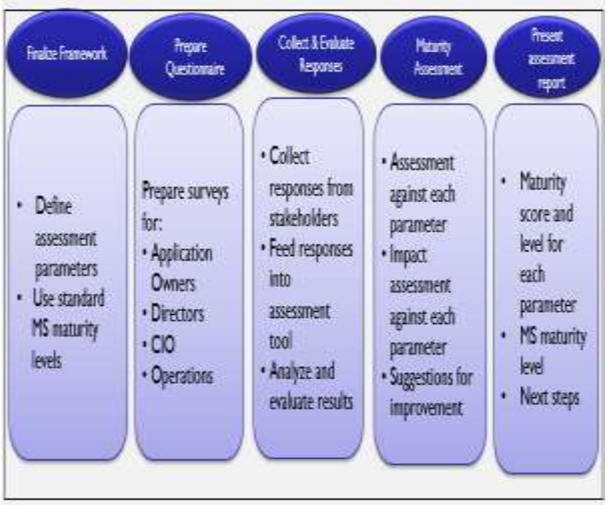
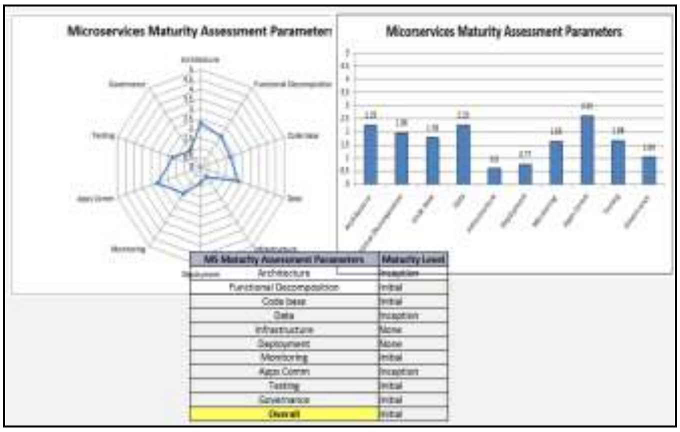

Finalize Framework
Prepare Questionnaire
Collect & Evaluate Responses
Maturity Assessment
Present assessment report
• Define assessment parameters
• Use standard MS maturity levels
Prepare surveys for:
• Application Owners
• Directors
• CIO
• Operations
• Collect responses from stakeholders
• Feed responses into assessment tool
• Analyze and evaluate results
• Assessment against each parameter
• Impact assessment against each parameter
• Suggestions for improvement
• Maturity score and level for each parameter
• MS maturity level
• Next steps

Figure 5 – Assessment Methodology

7. Maturity Score Card

The Maturity Score Card describes the maturity scores with respect to each of the Microservices architecture parameters and their maturity level on basis of the maturity assessment exercise, conducted across the enterprise. The following diagram depicts the sample Maturity scorecard.



MS Maturity Assessment Parameters	Maturity Level
Architecture	Inception
Functional Decomposition	Initial
Code base	Initial
Data	Inception
Infrastructure	None
Deployment	None
Monitoring	Initial
Apps Comm	Inception
Testing	Initial
Governance	Initial
Overall	Initial

Figure 6 – Microservices Maturity Score Card

8. Conclusion

Microservices Maturity within the architecture framework will vary across the business capability, technology architecture, deployment, testing, infrastructure and monitoring as well as the architecture blueprint. This is an ever-evolving process of Enterprise that leads to an efficient, effective responsive development and support organization.

The Microservices Reference architecture blueprint is not a document that we produce once, store on the shelf and reference on occasion. It is a plan and a methodology and allows for better analysis for future decisions. To achieve success with Microservices, enterprises must first build a well-designed application according to existing platform standards; refactor the application into a collection of Microservices as necessary to meet business needs. With the right people, processes, and tools, Microservices can deliver faster development and deployment, easier maintenance, improved scalability, and freedom from technology locking.

The most important parameters considered in the Microservices adoption are:

- o Refactoring Existing Applications
- o Service Identification
- o Technology choices
- o Framework choices
- o Patterns and integration
- o Deployment Automation
- o Monitoring, Troubleshooting and Management

Acknowledgements

The authors would like to thank Hari Kishan Burle, Raju Alluri of Global Enterprise Architecture Group of Wipro Technologies for giving the required time and support in many ways in bringing Up the article as part of Global Enterprise Architecture Practice efforts.

References

1. https://info.leanix.net/hubfs/leanIX_Microservices-Study.pdf
2. S. Newman, Building Microservices – Designing Fine-Grained Systems, O'Reilly, 2015
3. E. Wolff, Microservices: Flexible Software Architecture, Addison-Wesley, 2016

Author Profile

Dr. Gopala Krishna Behara is a Lead Enterprise Architect in the SCA Practice division of Wipro. He has a total of 21 years of IT experience. Reached at gopalkrishna.behra@wipro.com

Tirumala Khandrika is a Senior Architect in the SCA Practice division of Wipro. He has a total of 16 years of IT experience. Reached at tirumala.khandrika@wipro.com

Disclaimer

The views expressed in this Article are that of authors and Wipro or any other organization does not subscribe to the substance, veracity or truthfulness of the said opinion