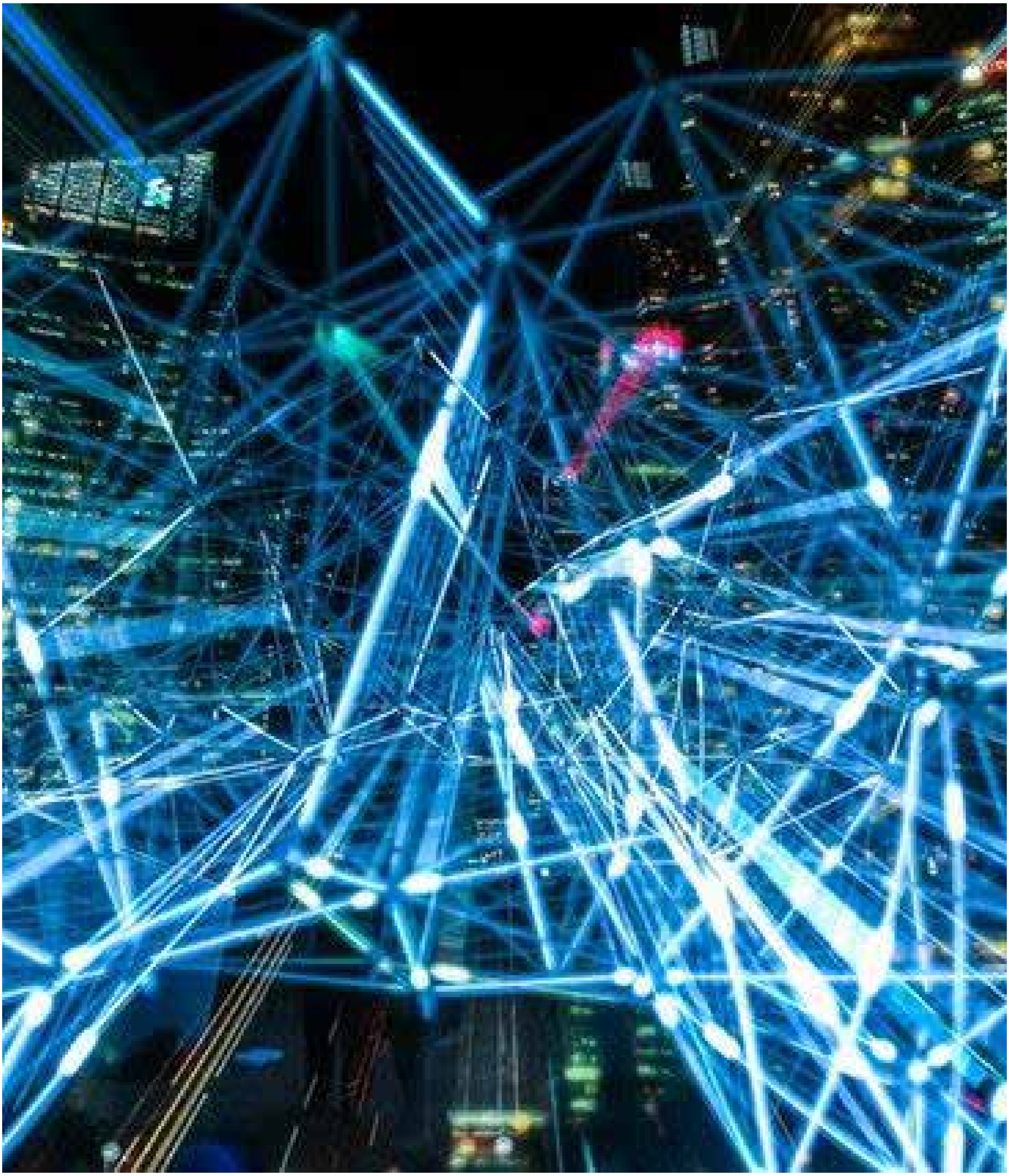
Microservices Approach – Constraints and Limitations

POV by Roger Smith, Enterprise Architect Consultant



Contents

Recently, much discussion has been done about Microservices [MS] and Enterprise Service Bus [ESB], while patterns discussions are strewn over event-driven, orchestrated and choreographic implementations. Begging your indulgence, we would suggest a slightly different approach focused more on flexibility and agnostic technology, at least to the degree possible.

1.
What is a Service Mesh?

So, what is a service mesh? An excellent question based on the ongoing definitional debate. However, let me confuse the subject with the term “Distributed Microbus” [DM]. The following paragraphs will discuss key aspects of DM and the combination of patterns that make up that construct.

If we decompose DM, we find that it contains all the previously mentioned elements, but in a slightly different configuration. Essentially, the ESB is broken into individual segments [ESBs], where each segment has an individual responsibility to manage one MS endpoint and manage the communications between that endpoint and others within the workflow.

This approach reduces the complexity of the *ESB* and allows them to be distributed in a cloud environment across multiple environments providing flexibility, scalability and resilience to the enterprise. This also provides the ability to move the processing to the edge providing a distributed Virtual Enterprise [VE] where deployment, multi-tenancy and zero-downtime updates can be more easily accomplished. These advantages are available at the enterprise level and extend to the microservice environment allowing capacity to be expanded and contracted as the workload changes periodically.

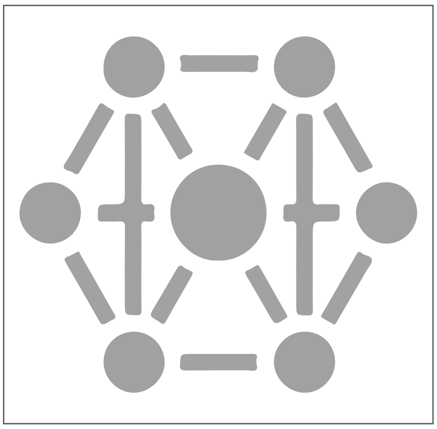
The microservice patterns within the DM consist of those patterns previously mentioned and utilized in the solution track that is the best fit for the capability required. The solution sets for transaction processing, inquiry, detection and performance can be intermingled, but in discrete sets for ease of development and deployment.

So, as you have already guessed, this is the anatomy of service mesh architecture.

The pluggable aspect of the service mesh and the polyglot capabilities make it the ideal solution framework for the current and future technology implementations. This provides the ability to change business functionality more easily and quickly, both of which are essential when SWOT analysis indicates the need for competitive change.

2.

Service Mesh Pattern for Mainframe Migration Complexity

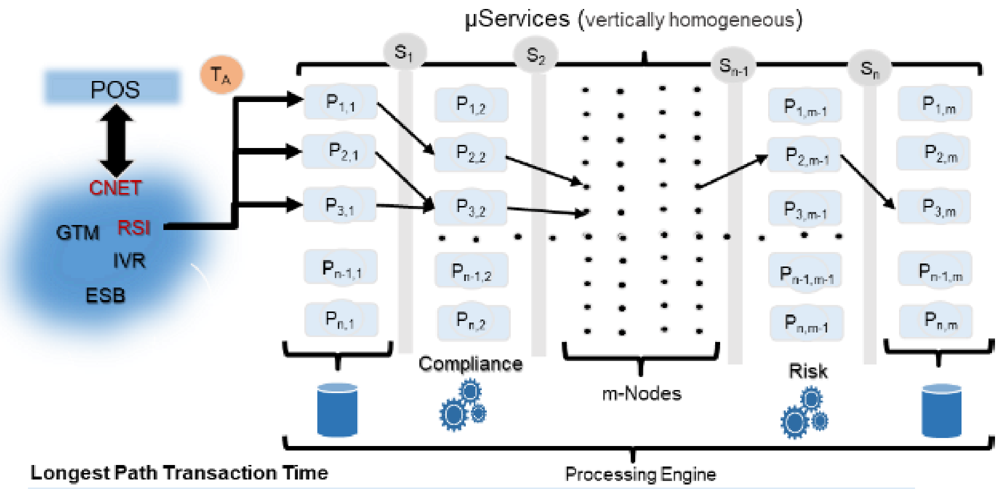


Let us now turn our attention to the more pertinent aspects of how and why the service mesh pattern would fit in solving the complexities of the mainframe migration.

Amdahl's Law is one of the most utilized theoretical arguments proposed by hardware, software and cloud providers alike. The law has its roots in Queuing Theory and in practical experience. Anyone that enjoys the lines at retail stores or airport counters can easily identify Amdahl's Law in action when the new counter opens or more ticket agents arrive to manage those in the "queue".

Most proponents of cloud-based replacement of on-prem enterprise systems will exercise their estimates on the $\mathbf{M_a/M_p/c}$ queuing model where $\mathbf{M_a}$ is the Markov estimator for transaction arrival time, $\mathbf{M_p}$ is the Markov estimator for transaction processing time and c is the number of available servers for servicing the transactions. Let me submit a caveat to the queuing model, namely $\mathbf{M_a/M_p/S_i/c}$ where $\mathbf{S_i}$ is the switch time between nodes.

Let us consider the processing nodes in Figure 1 which are labeled $\mathbf{P_{1\text{-}n,n\text{-}m}}$ where ***n*** represents the number of homogenous microservices in a cluster that is available to process the first step in a



Longest Path Transaction Time

$$T_A = MAX(P_{1,1}, P_{2,1}, P_{3,1}) + MAX(S_1[P_{1,1} \rightarrow P_{2,1}, P_{2,1} \rightarrow P_{3,2}, P_{3,1} \rightarrow P_{3,2})] + MAX(P_{2,2}, P_{3,2}) + MAX(S_2[P_{2,2} \rightarrow P_{x,y}, P_{3,2} \rightarrow P_{p,q})] + \cdots + S_{n-1}(P_{x,y} \rightarrow P_{2,m-1}) + P_{2,m-1} + S_n(P_{2,m-1} \rightarrow P_{3,m})$$

Figure 1

transaction corridor and ***m*** is the number of steps in the corridor, variable by a corridor. What is depicted in Figure 1 is an *ESB* pattern where the process step returns to a common point, $\mathbf{S_i}$ before proceeding to the next step. If the tasks carried out at the common point become part of the process, combining $P_{1\text{-}n,1}$ with $\mathbf{S_1}$ for example, and runs in parallel, then time savings would be realized for the transaction processing. $\mathbf{S_1}$ would not be a singular forwarding endpoint in this case, but would be determined by corridor specifics.

It would be remiss if we did not discuss the obvious concern related to physical transport time between services or from service to a common point. Service mesh would reduce traffic by half by not returning to the common bus and transport time can be improved by using high throughput capabilities available from the cloud provider, such as HTC capabilities available in AWS.

SAGA framework is the embodiment of service mesh, but also has event-driving and orchestration capabilities which makes it a great fit for the tasks ahead for mainframe migration. SAGA will enable best-fit implementation and the ability to pivot to other technologies if the need arises. A flexible architecture that can adapt to the situation is an absolute requirement for any enterprise that wants to be agile, competitive and profitable.

3.

Theory to Practice – Mphasis Experience in High-performant Systems

Mphasis has significant expertise ranging from real-time systems to financial transactions including telecommunications, airline operations, oil exploration and military aircraft on-board systems. This expertise and the practical applicability extend through all facets of high-performance transaction processing be it a financial payment or an ACARS message related to aircraft status. We understand the payments domain and the platforms that make them successful.

Mphasis has projects with a Large Global Bank [LGB] that embody the service mesh philosophy with the blue/green implementation strategy. This strategy ensures uninterrupted migration to the LGB private cloud by being involved in the design of the service mesh architecture and 100% involved in the rewrite of applications for migration. Other projects involve innovation in migrating COBOL to Java microservices utilizing CICS and batch so that the microservices can later be migrated to the cloud with minimal refactor as LGB migrates bounded contexts to their cloud implementation.

Mphasis is also engaged in several projects with LGB related to payments modernization where the involvement included 40% architecture, 70% design, and 100% development in payment initiation (COBOL, Mainframe, Tandem), funds control, and processing (Kafka, Java), clearing and settlement integration (Java) and payment investigation (Camunda, Pega).

Mphasis has several projects with a Global Logistics Company [GLC] including a microservice project that was designed and built for core processing for onboarding client partners for participation in the manifesting and scheduling services for shipping products to various locations. Mphasis engaged in 90% of the design and 85% of the development for Domain-driven delivery of bounded contexts to the new environment. Microservice deployment was based on Spring Boot and included the deployment of distributed data environments including mainframe data sources with distributed cache using GemFire and Hazelcast for consistency and fault tolerance.

4.

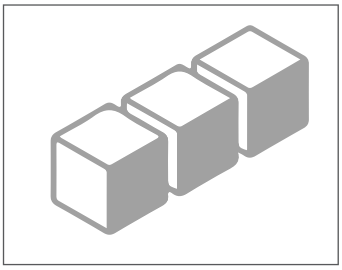
Law of Diminishing Returns for the Proposed Solution

Finally, let us turn our focus to the Law of Diminishing Returns [LODM] with respect to the proposed solution.

The solution is designed with considerable usage of object-oriented design precepts in that every system component is conceptually an object, an object-oriented virtual enterprise.

This approach provides the means to break down the complexity of the system into more manageable pieces to be considered independently and then aggregated for corporate integration. Additionally, each piece can be further broken into constituent parts so that maximum utility can be realized through microservice flexibility.

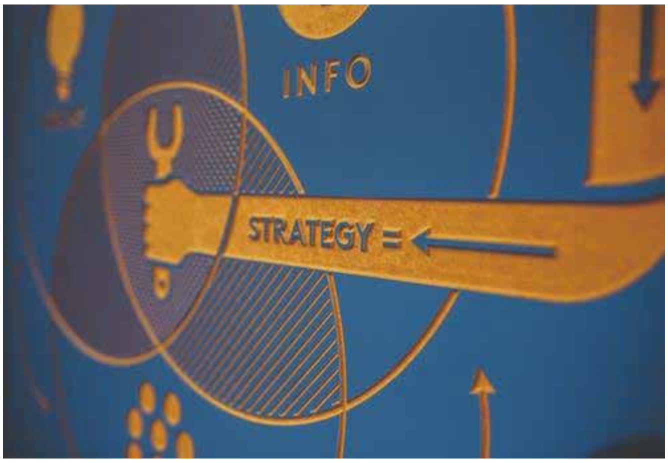
When focusing on LODM, three considerations come to the forefront:



First, care must be taken with this approach to resist breaking the "objects" into many fragmented pieces especially when microservice architecture is involved. The phrase "death by a thousand cuts", used more often in data strategy discussions, is very apropos when applied to microservices so attention to cohesion and coupling is extremely important.

Second, replicating the enterprise "object" to zones where the localized traffic may not be sufficient to cover the costs, needs to be closely monitored.

Third, related to replicating, deployment is an all-or-none proposition, so identifying a minimal enterprise configuration that can employ autonomic expansion is of utmost importance for keeping costs low while serving a growing number of customers.



Mitigation for circumstances related to LODM would entail employing a Minimal Configuration Virtual Enterprise [MVE] with minimal scale to accommodate the low traffic regions which can dynamically (autonomically) expand to handle larger traffic volumes as required.
This organic expansion/contraction is essential to maintaining optimal cost/benefit margins as well as providing leverage to enter new markets with a competitive cost strategy.

Author

Roger Smith

Enterprise Architect Consultant

Roger is an Enterprise Architect for Mainframe Modernization at Mphasis. In this role, he is responsible for designing digital solutions that increase security, flexibility, reliability and resilience for on-prem, hybrid and cloud-based enterprise environments.

Roger has over 40 years of IT experience in multiple domains and across many different technologies. His focus has been on secure system design, security engineering (authentication, authorization, encryption, algorithm analysis), performance engineering, quality assurance, systems integration and system improvement (DevOps, CI/CD/CM).

Roger's industry experience includes Financial (Banking, Investment, Audit), Retail, Utilities, Telecommunication, Decision Support, Identity Verification, Transportation (Airline Maintenance/Operations, Cargo, Rail), Oil and Gas and Defense Systems.

For more information, contact: marketinginfo.m@mphasis.com

USA
Mphasis Corporation
41 Madison Avenue
35th Floor, New York
New York 10010, USA
Tel: +1 (212) 686 6655

UK
Mphasis UK Limited
1 Ropemaker Street, London
EC2Y 9HT, United Kingdom
T : +44 020 7153 1327

INDIA
Mphasis Limited
Bagmane World Technology Center
Marathahalli Ring Road
Doddanakundhi Village, Mahadevapura
Bangalore 560 048, India
Tel.: +91 80 3352 5000

www.mphasis.com