Solving the Microservices Adoption Problem in Telecom: Ideas & Strategies

Abstract

The telecom industry has witnessed a sea change in the last few years, with customers demanding more flexibility and greater experience quality than ever before. Services need to be delivered in real-time, instead of days or weeks. Adopting a digital-first and self-service ready support model is imperative – but this is out of sync with the existing tightly coupled, non-scalable monolithic architecture. With customers calling for zero touch integration across the entire lifecycle, from ordering and activation to billing and support, telcos cannot hold onto legacy systems. Simply put, an entire refresh is required including infrastructure, customer services, as well as operational and business support systems (OSS/BSS).

That's where microservices architecture comes into play as one of the solutions for the above mentioned demand. By approaching their application landscape through a microservices-oriented lens, telcos address the rigidity present in their current systems and pave the way for future cloud transformations. This implies a whole new way of designing and deploying telecommunications services.

TM Forum, has proposed a new architecture and set of best practices called the Open Digital Architecture (ODA). Through ODA, TM Forum hopes that telcos can replace legacy OSS/BSS with a flexible, API-based model. The goal is to achieve a Lego-like architecture where the application landscape depends on standardized microservices and a set of normalized APIs.

What all of this implies for the telecom sector, is the necessity of moving to a microservices environment as soon as possible. This will not only make future transformations easier but also provide a competitive advantage in the near term.

Challenges to Microservices Implementation

Given its clear advantages, what's impeding telcos from transitioning to a microservices architecture? The answer is multi-layered. First, there is the issue of monolithic systems which involves a large number of legacy components, technologies, and business processes. Second, the emergence of next-gen networking technologies like Software-Defined Network (SDN) and Network Function Virtualization (NFV). These are causing rapid expansion, driving the need for telcos to embrace microservices architecture transformation as they need to be agile in a fast changing and connected world. Last but not the least, the fact remains that no standardized framework has been published till date, capable of handling all types of microservices implementation scenarios and carrying recommendations for telcos who want a complete overhaul.

Challenges in microservices adoption:

- A monolithic structure uses a singular database; when this data is distributed across multiple microservices, a new strategy for maintaining data concurrency is required
- Given the complex nature of a telecommunications environment, identifying and mapping business capabilities to microservices is a genuine problem
- The scope of the microservices structure and capabilities need to be defined, reducing any ambiguity
- There are no clear best practices to address the impacts after implementing microservices

These are only some of the challenges faced by telcos when commencing towards microservices adoption.

That's why a prerequisite for kickstarting a transition to microservices architecture is a well-articulated "Business WHY" – telcos must assess each of their business processes and initiatives, specifying the to-be-achieved goal to aim for. Also required is a preemptive ROI analysis along with expected timelines and impact to the business.

Understanding Microservices Development Scope In-sync With Business Initiatives

Microservices essentially entails the decoupling of monolithic architecture, into modular building blocks or 'micro' applications which are flexible and portable. When considering the transition to microservices architecture, each business initiative/process must be assessed against various decoupling patterns. This will help arrive at the right design and implementation approach for that particular initiative.

For example, imagine a scenario where the application is read-intensive, with an additional interface where users can view details on mobile. Here, the best decoupling pattern would be to create a separate data instance for 'read' and then scale it according to the load as required. The data will be updated incrementally in near-real time, reducing loads and resource requirements. This data will be displayed on a mobile device through a specific API that works through an API management layer. An app used by telecom users to only check the status of their service requests is among the more basic read-intensive use cases Telcos could expect.

If we break down this example, two technical areas emerge – how to handle read data and scaling, and how to link the API for mobile app consumption. We could break this down even further and identify patterns for data transparency and other granular elements. It's only by considering all of these possible steps can we arrive at a pattern for each and consequently reach a comprehensive microservices design, applicable for the scenario.

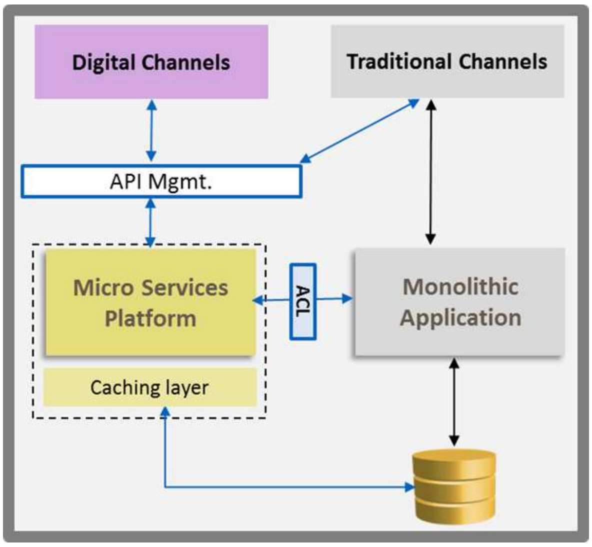
In other words, it's only by understanding every element of an application interaction that telcos can find the right pattern at each level and create an effective microservices design for the same.

Let's now look at some of the patterns telcos can consider when looking to transition a particular process/initiative into a microservices environment.

Five High-Level Patterns to Look For When Decoupling Applications

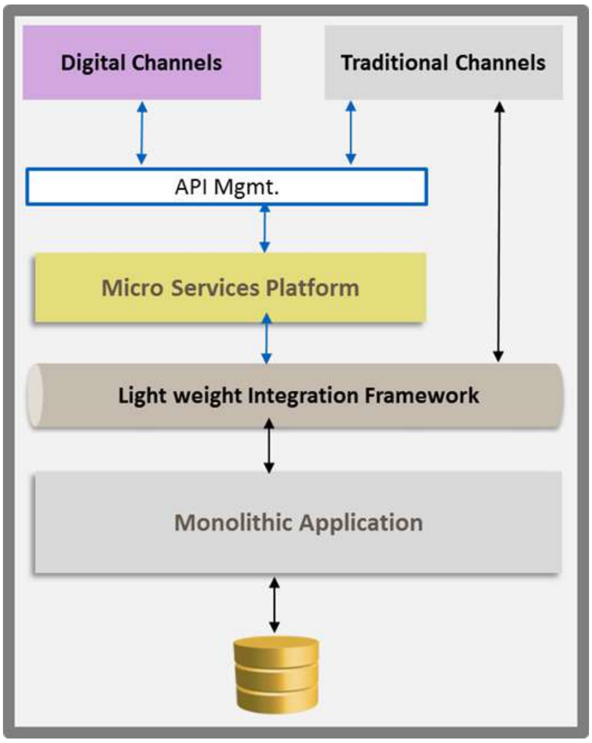
Pattern 1:

A tightly coupled business logic can be segregated from a legacy application, developing microservices incrementally and not at one go, by following the Single Responsibility Principle (SRP). This helps execute digital initiatives at an incremental pace because each component independently controls each functionality, without being impacted by changes to its surrounding parts. This is illustrated below.



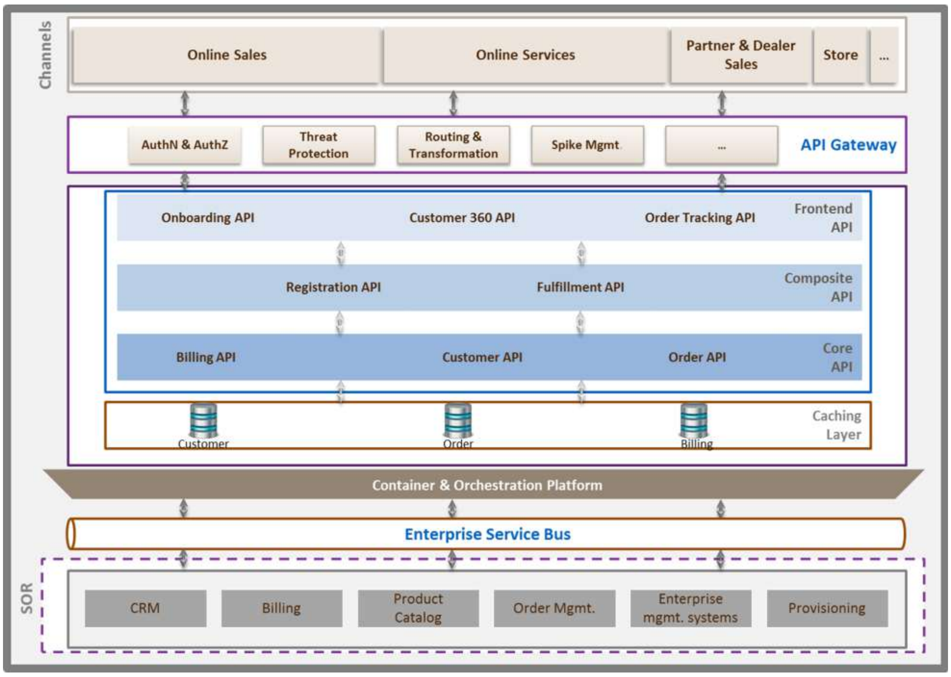
Pattern 2:

To streamline transition for decades-old telcos, business logic piled up in the Enterprise Service Bus (ESB) layer over many years must be extracted. Microservices are then gradually developed to replace the heavyweight ESB with a lighter integration framework, as illustrated below. This is essential, given that ESB was the cornerstone of service-oriented architecture and needs a whole new articulation for the microservices era.



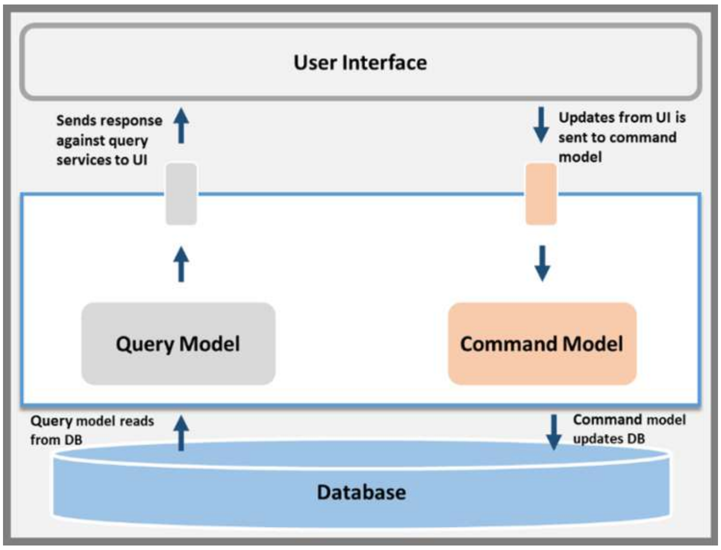
Pattern 3:

To support digital consumption, microservices can be built on top of the system of records. APIs are exposed via the API gateway and any existing SOAP-based webservices can be exposed through the existing ESB (to be consumed within the enterprise boundary). This pattern – illustrated below – tries to strike a balance between legacy ESB and the new API-based model, easing the transition.



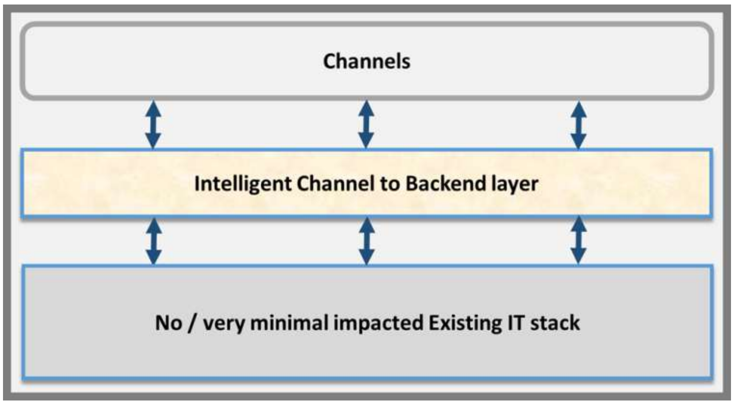
Pattern 4:

In our first example, we had mentioned data responsibility as one of the component elements when breaking down an application scenario. Below, we illustrate a pattern (as per Martin Fowler's CQRS pattern) for segregating data responsibility in order to balance load and scale for read & write tasks.


User Interface
Sends response against query services to UI
Updates from UI is sent to command model
Query Model
Command Model
Query model reads from DB
Command model updates DB
Database

Pattern 5:

In this last pattern, we deal with a very common use case for any modern Telco – handling omnichannel interactions and analytics. This requires an intelligent layer which can enhance channel experience without significantly impacting the existing IT stack at the backend.


Channels
Intelligent Channel to Backend layer
No / very minimal impacted Existing IT stack

Exploring Implementation Opportunities without Disturbing BAU

The vast distance between legacy monolithic applications and a next-gen microservices architecture simply cannot be ignored. At its very foundation, technical components communicate inside the memory when operating in legacy mode. In microservices, on the other hand, all communication between components take place over the network. This makes network design and implementation absolutely critical when planning a migration to microservices at any telecommunications environment. Therefore, before starting development cross-functional requirements such as network logs, monitoring, and security need to be made set in place.

Obviously, this migration would not be a one-time-only initiative, completed overnight. During the migration stage, a Strangler pattern (per Martin Fowler's blog post) is recommended, where new applications and services are structured in a way that they can be easily intercepted and replaced during future Strangler cycles. This approach would bring immense flexibility and agility for telcos, making room for 'n' number of iterations until finally, the target architecture is reached and the legacy application is decommissioned.

Two things should be noted here: the legacy system and microservices platform will need to run concurrently, at least for a while. That's why an anti-corruption layer (ACL) is introduced between sub-systems.

Finally, just as two technologies must co-exist for successful transformation, so should technical teams and business resources responsible for the old and new systems come together to achieve meaningful results. In fact, the team running legacy systems has a major role to play in microservices adoption, collaborating closely with the business team (who will provide details on the various processes and initiatives to be migrated). Once this seamless union of many perspectives and diverse contributions are achieved, telcos can look forward to streamlining a complex migration pathway into pragmatic and easy-to-implement milestones.

About The Author

Subhendu Chakraborty

Subhendu Chakraborty is an Enterprise Architect from TCS Communications, Media and Information Services Industry Advisory Group (IAG) team. He is an AWS certified solution architect and also TM Forum certified business development manager. He has over 18 years of IT experience in the areas of application architecture, Microservices design and migration, API-fication, Cloud migration and integration architecture. Subhendu holds a Master's in Software Systems from BITS, Pilani.

Contact

Visit the Communications, Media & Technology page on www.tcs.com

Email: global.cmi@tcs.com

Blog: Next Gen CMI

Subscribe to TCS White Papers

TCS.com RSS: http://www.tcs.com/rss_feeds/Pages/feed.aspx?f=w
Feedburner: http://feeds2.feedburner.com/tcswhitepapers

About Tata Consultancy Services Ltd (TCS)

Tata Consultancy Services is an IT services, consulting and business solutions organization that delivers real results to global business, ensuring a level of certainty no other firm can match. TCS offers a consulting-led, integrated portfolio of IT and IT-enabled, infrastructure, engineering and assurance services. This is delivered through its unique Global Network Delivery Model™, recognized as the benchmark of excellence in software development. A part of the Tata Group, India’s largest industrial conglomerate, TCS has a global footprint and is listed on the National Stock Exchange and Bombay Stock Exchange in India.

For more information, visit us at www.tcs.com

TCS Design Services I M I 05 I 19