Microservices: Plug into the Power of Small

The Incremental Approach to Migration from Monolithic to Microservices

product engineering services

wipro

The appeal of microservices for enterprises can be attributed to the fact that it reduces release cycles from a few weeks to a few days.

Organizations that have large, monolithic architecture are painfully aware of the slow application redeploys, and lack of agility and resilience when new features and functionality need to be added. In response to the problem, Microservices Architecture has begun to emerge as the apt option. Reason: the microservices approach - of developing small services that are independently deployable and bound together with lightweight communication mechanisms - requires smaller, autonomous teams that don't need to know the entire application. Scaling organizations becomes easier and quicker.

Microservices architecture brings many advantages. Firstly, with microservices, services can be scaled based on load and they can be upgraded faster because they are independent of other services. They can be changed and deployed without having to touch other parts of the system. This contains the impact and cost of failure. Second, as services are individually testable and manageable, the architecture simplifies development. Thirdly, it is not mandatory to develop all services using the same technology (language). This provides developers with a considerable amount of freedom to use a technology that is the most appropriate for a specific service. Finally , organizations don't have to co-ordinate between large teams that know the entire application. Small, specialized teams can be deployed resulting in autonomy, efficiency and accountability.

Designing microservices is not simple - more components mean more complexity. Here, system integration and testing need detailed planning. And, an increased number of services also implies an increase in operational, deployment and monitoring effort. In most instances, as components are deployed independently and communicate with each other over networks, the need for infrastructure (virtual machines) increases. As the system evolves, understanding it as a whole can become daunting. However, a smooth migration to microservices is becoming increasingly manageable as the tools and technologies required to enable microservices are on a quick path to maturity (see Table 1).

Although microservices are in demand, there are aspects of transforming from a monolithic application to a microservices-based architecture that are not well understood. Evaluation of the need and fit is pivotal and so is a right approach to the transition.

Picking the right provider

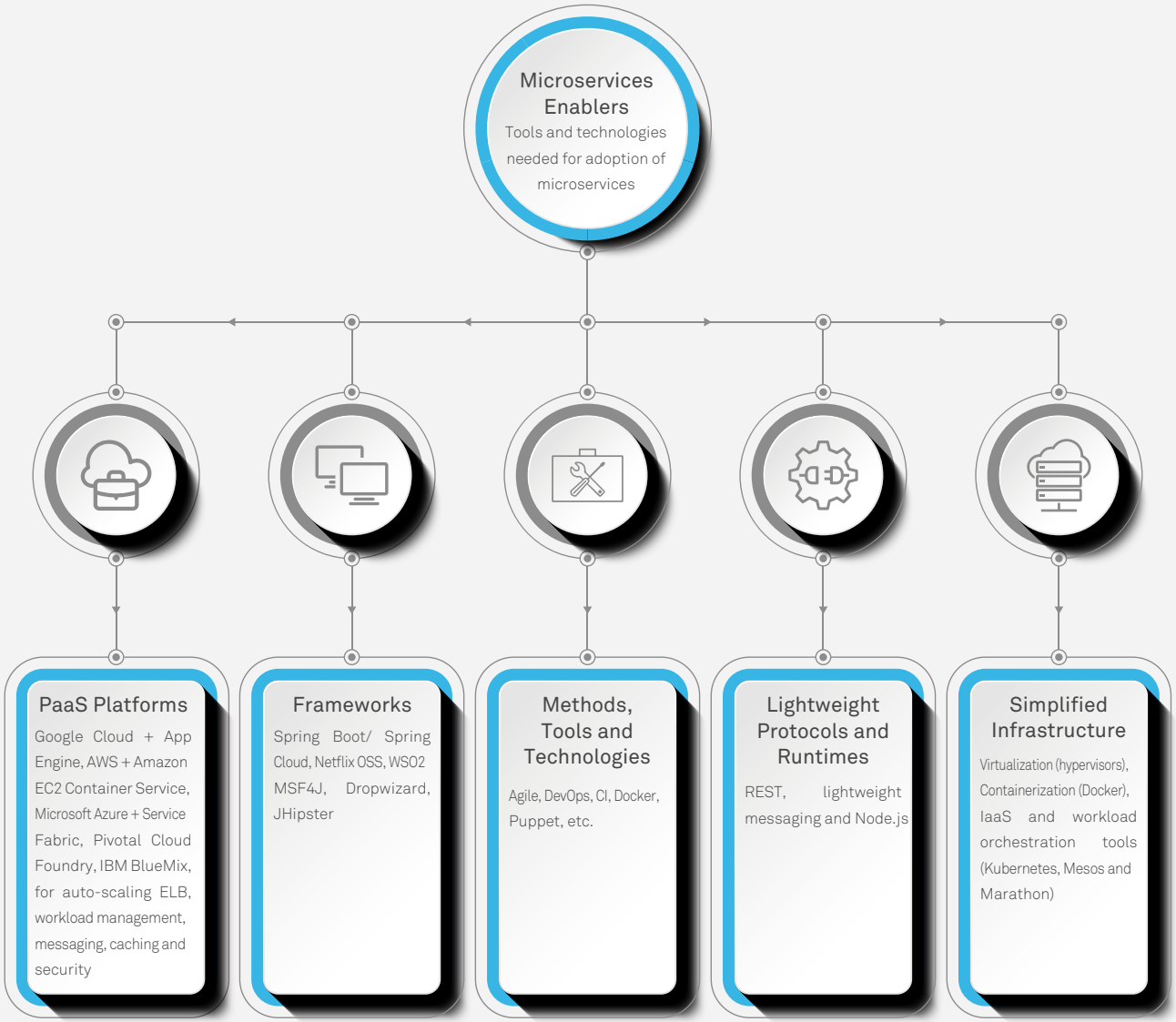

Table 1: Microservices Enablers

The path to Microservices

Microservices architecture can definitely bring in many benefits, only if applied the right way! Take the case of a large Digital Video Recorder (DVR) vendor with operations across multiple geographies. The vendor had an application to handle meta data, recording, logging of user behavior data for analysis, etc., that, over a period of time, bloated beyond control. The code became unmanageable and the number of APIs grew substantially. The level of complexity meant that new features were taking several weeks to go into production. And if anything failed in

production, the entire release had to be rolled back. Then, the release cycle had to be repeated.

The problem was that the DVR vendor was forced into scaling the entire application instead of targeting only those components that needed scaling. In addition, with the application written in Java 6, developers were unable to take advantage of the latest Java release or other languages like Python which would have ensured agility in development. The solution was to deploy microservices architecture and kill the pain of long release cycles along with the associated rollbacks.

Problems faced by the enterprise mentioned in this case was addressed by freezing the existing application and releasing all new features as new microservices-based applications. A plan to incrementally move the features of an existing application to a new microservices application was developed.

In the modified environment, old applications and the new microservices applications exist side by side. They communicate with each other to fulfil user requests. This means scaling of features and new releases can be done easily. Failure means that only the failed component has to be rolled back, making it faster and easier to create a fix. The new architectural style results in independent, easy-to-deploy and easy-to-fix components (See Figure 1).

This incremental approach works best and is recommended for businesses that are considering microservices adoption

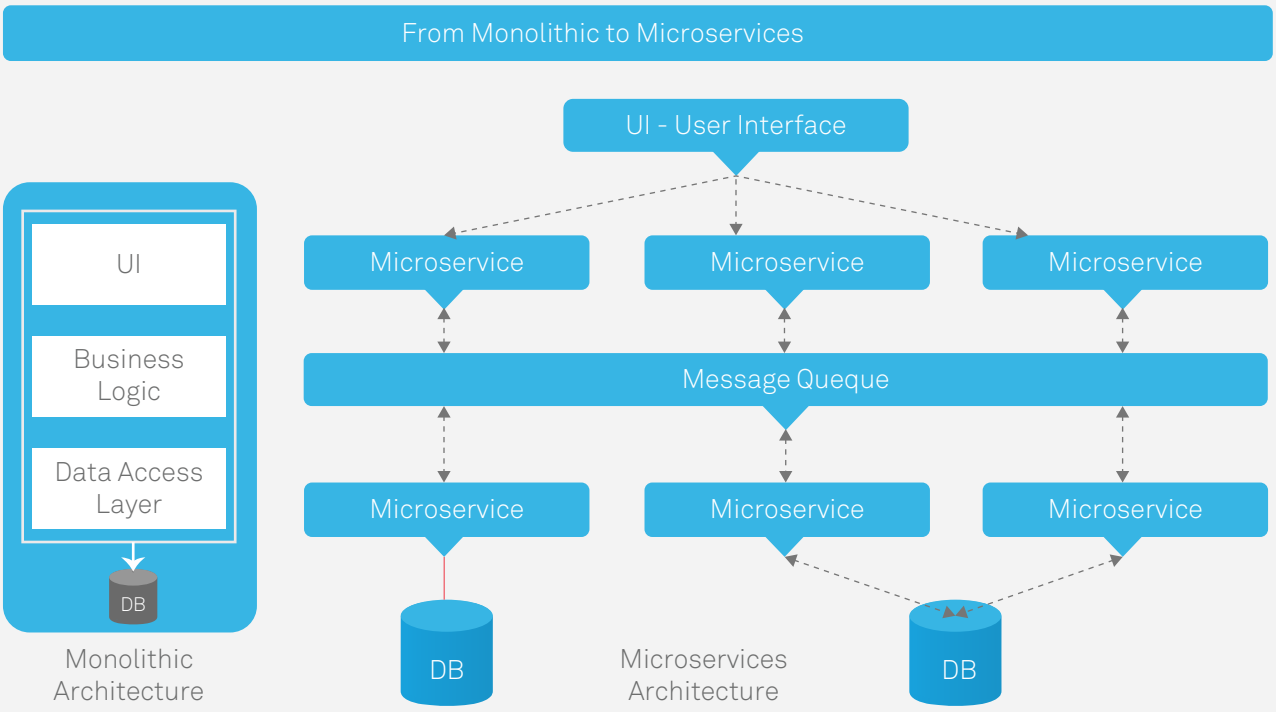

Figure 1: From Monolithic to Microservices

The incremental approach works better. Organizations can do two to three releases every day and meet user demands. It also allows developers the benefit of using the language best suited for a given component. It must be emphasized that the Big Bang approach is not only more difficult to manage due to the size and complexity of the existing application but the approach may also lead to significantly longer time for realization and presents the risk of unstable systems.

Key to success: Drawing the boundaries

The organizations that opt for migrating to microservices benefit from scale, agility and the ability to meet customer expectations. However, when designing microservices it is important to define the context and the boundaries of the service and ensure that too many nano services and databases don't unnecessarily add to complexity, data duplication and operational overheads. Microservices also become unmanageable without complete automated control over the software development cycle. Adoption of a DevOps culture and setting up DevOps pipelines reduces the complexity of managing services.

The Big Bang approach
does not work in migrating from monolithic applications to microservices-based applications. Executing the entire transformation in one go is not recommended.

About the author

Rajith Kumar
Practice Director, PES, Wipro Limited.

Rajith Kumar is a Practice Director at Product Software and Software services practice of Wipro's Product Engineering Services. He has over 18 years of experience in IT and engineering services across Banking, e-Governance, Retail and Manufacturing industries. He is currently leading key initiatives around scalable platforms and providing solutions in hyper scale architecture, product modernization and product sustenance areas.

For more information, write to Rajith at rajith.nuthalapati@wipro.com.

About Wipro Limited
Doddakannelli, Sarjapur Road,
Bangalore-560 035, India

Tel: +91 (80) 2844 0011
Fax: +91 (80) 2844 0256
wipro.com

Wipro Limited (NYSE: WIT, BSE: 507685, NSE: WIPRO) is a leading global information technology, consulting and business process services company. We harness the power of cognitive computing, hyper-automation, robotics, cloud, analytics and emerging technologies to help our clients adapt to the digital world and make them successful. A company recognized globally for its comprehensive portfolio of services, strong commitment to sustainability and good corporate citizenship, we have a dedicated workforce of over 170,000, serving clients across six continents. Together, we discover ideas and connect the dots to build a better and a bold new future.

For more information, please write to us at **info@wipro.com**

IND/BRD/JUL 2017-JUN2018