Mindtree
A Larsen & Toubro Group Company

The Evolution of **Enterprise Architecture**

A white paper by Dr. Shailesh Kumar Shivakumar, Mindtree Limited

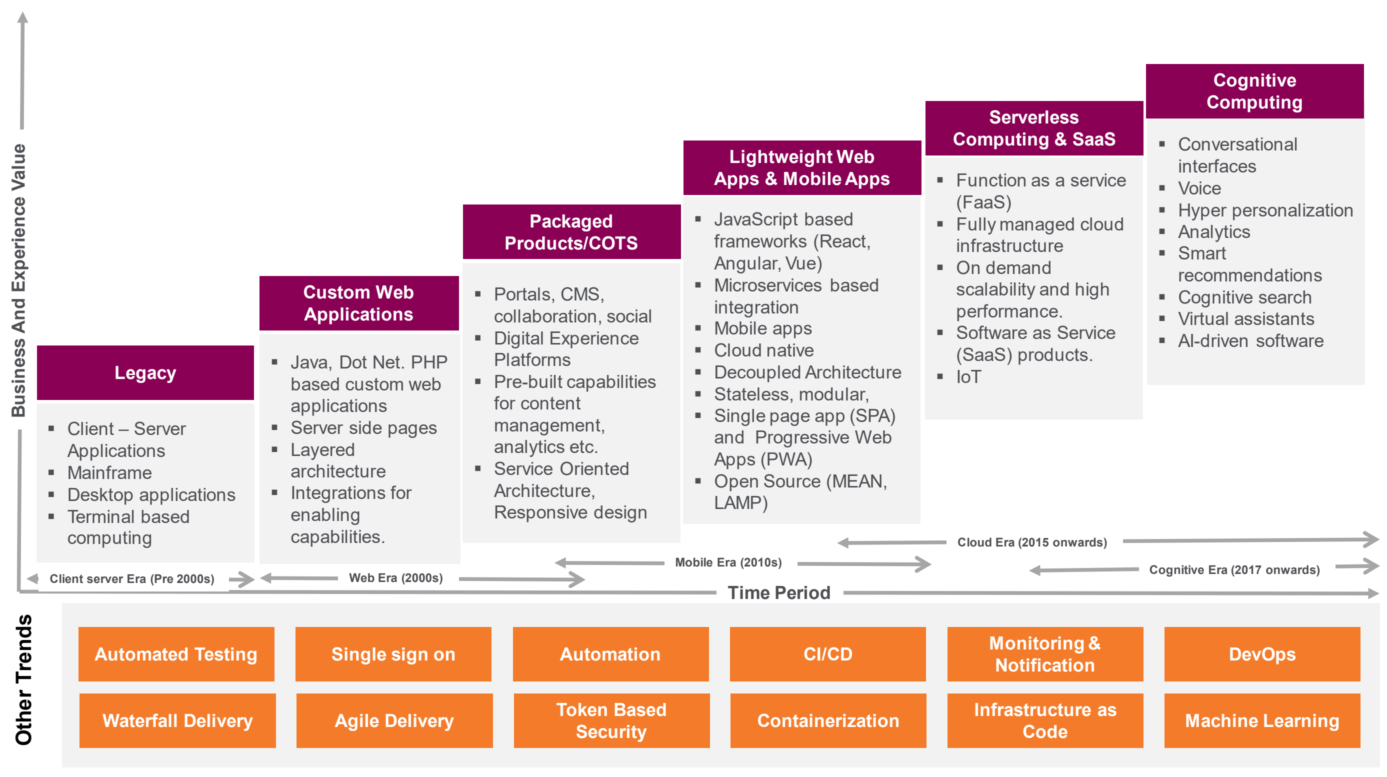
Welcome to possible

Enterprise applications are the main digital channels for information delivery, shaping user experience and business outcomes. While web has traditionally been the main end-user facing digital channel for most enterprises, in recent times, mobile apps have taken the front seat in terms of providing an engaging user experience.

In this white paper, I have traced the evolution journey of enterprise architecture starting with legacy applications. I have also detailed the key tenets for each of the architecture phases and the use cases.

Evolution of enterprise web applications

This is a high-level overview of enterprise archicture evolution in Figure 1.



Business And Experience Value
Legacy
Client – Server Applications
Mainframe
Desktop applications
Terminal based computing
Custom Web Applications
Java, Dot Net. PHP based custom web applications
Server side pages
Layered architecture
Integrations for enabling capabilities.
Packaged Products/COTS
Portals, CMS, collaboration, social
Digital Experience Platforms
Pre-built capabilities for content management, analytics etc.
Service Oriented Architecture, Responsive design
Lightweight Web Apps & Mobile Apps
JavaScript based frameworks (React, Angular, Vue)
Microservices based integration
Mobile apps
Cloud native
Decoupled Architecture
Stateless, modular,
Single page app (SPA) and Progressive Web Apps (PWA)
Open Source (MEAN, LAMP)
Serverless Computing & SaaS
Function as a service (FaaS)
Fully managed cloud infrastructure
On demand scalability and high performance.
Software as Service (SaaS) products.
IoT
Cognitive Computing
Conversational interfaces
Voice
Hyper personalization
Analytics
Smart recommendations
Cognitive search
Virtual assistants
AI-driven software
Cloud Era (2015 onwards)
Mobile Era (2010s)
Cognitive Era (2017 onwards)
Client server Era (Pre 2000s)
Web Era (2000s)
Time Period
Other Trends
Automated Testing
Single sign on
Automation
CI/CD
Monitoring & Notification
DevOps
Waterfall Delivery
Agile Delivery
Token Based Security
Containerization
Infrastructure as Code
Machine Learning

Figure 1 High-level view of Enterprise architecture evolution

The key categories of enterprise architecture in the evolution journey are legacy applications, custom web applications, package products based application, lightweight web apps and mobile apps, serverless computing and cognitive computing.

I have elaborated each of these categories in subsequent sections.

Legacy applications

Legacy applications mainly include client-server and mainframe applications which used terminals for interaction. Desktop-based applications are prominent in this category.

TCP/IP, CORBA, , SQL, Mainframe and database technologies are popular in the client-server era.

Key tenets of legacy applications

- Majority of the applications are desktop-based.
- Most of the applications are function-based monolith (a single layer handling all application responsibiliites)
- The key communication method comprises client server-based applications

Custom web applications

Custom web applications were a main part of the web era, which started in the early 2000s. These applications were mainly developed with server-side technologies such as Java, Dot Net, PHP and others. JSP, ASP and PHP were the popular web technologies used. The web page refresh needed a blocking server-side call. Service Oriented Architecture (SOA) was the popular methodology wherein the integrations happened through heavyweight webservices. HTML 1.1, HTML 2.0, XML and SOAP were the main web standards.

AJAX and XML HTTP requests were later used to asynchronously load the data from the server to improve the performance.

Open source frameworks such as Spring MVC, Struts, JSF, MEAN stack (MongoDB, ExpressJS, Angular, and NodeJS), LAMP Stack (Linux, Apache web server, MySQL, PHP) were used to develop the web applications.

Key tenets of custom web applications

- Model View Controller (MVC)-based layered archicture and presentation layer was mainly web focused
- Layered architecture was developed using separation of concerns principle
- Custom development was heavily used across all layers
- MVC was the most preferred pattern and the applications were monolith in nature

Packaged Products/ COTS

Packaged web products such as horizontal portal and content management applications came up with pre-built capabilities such as presentation management, content management, site and page development tools and out-of-the-box integrators with external systems. The packaged solutions improved the developer productivity through ready-to-use modules.

Digital Experience Platforms (DXP) further improved upon the experience capability of the packaged solution and provided advanced capabilities such as personalization, collaboration, search, security, analytics, campaign management, mobile experience and e-commerce, that are mainly used for digital marketing initiatives.

JSR portlets, OSGi, REST, SOAP and WSRP are the main standards used in this category.

Key tenets of packaged products

- Inbuilt support for key features such as presentation, content management, ecommerce, campaign management, integration adaptors etc.
- Service oriented architecture (SOA) is mainly used for integrations
- Responsive web design (RWD) was extensively used to mobile-enable the applications

Lightweight web apps and mobile apps

In early 2010, mobile started gaining prominence and enterprises began using mobile apps as the primary channel to provide online experiences to their customers.

Lightweight web apps provide highly responsive, interactive and an omnichannel-enabled experienced mainly through JavaScript frameworks. The UI layer interacts asynchronously with back-end microservices. Single Page Applications (SPA) and Progressive Web Applications are the most popular applications in this category. To provide high engaging user experience mobile apps were used.

Lightweight UI and integrations, modular and independently scalable microservices, and cloud deployments contributed to highly scalable and high performing applications.

HTML 5, CSS 3, Angular, NodeJS, React, Vue, microservices (such as Spring Boot microservices), open source technologies (such as MEAN stack, LAMP stack) are the popular technologies used in this category.

Key tenets of lightweight web apps and mobile apps

- The applications mainly used lightweight/ lean model in development and integration
- Decoupled architecture used well-defined service contracts and stateless APIs for building the enterprise applications.

- Mobile-friendly progressive web apps (PWA) and Single Page Applications (SPAs) are mainly used to enable a rich mobile experience
- Cloud-native and cloud-compatible applications are developed to provide on-demand scalability, high performance and high availability

Serverless Computing

Serverless computing abstracts the underlying server (such as application server), infrastructure (OS, database, VM), quality (such as performance, scalability) and management from the developers. Cloud platforms provide fully managed services so that developers can focus on developing the underlying business logic and the Function as a Service (FaaS).

AWS Lambdas, Azure Functions are examples of serverless computing.

Key tenets of serverless computing

- Cloud-native and fully managed services
- On-demand scalability, high availability and high performance
- Cloud-managed infrastructure and platform

Cognitive Computing

Cognitive computing is an emerging and evolving area that employs Machine Learning algorithms coupled with Natural Language Processing and Predictive Analytics to extract and understand relevant information. Cognitive and voice search, virtual assistants and smart recommendations enable users to make informed decisions faster.

Key tenets of cognitive computing

- Contionous learning and improvement based on historical data
- Conversational interfaces, chatbots and virtual assistants
- AI-led recommendation, personalization and search

Over the different eras, we have also noticed other trends such as waterfall delivery, automated testing (leveraging tools such as Selenium, LoadRunner to automate regression testing and functional testing), single-sign-on (providing seamless access to authenticated applications), agile delivery (quicker time to market through shorter sprints), automation (automating repeated activities such as release management, build, monitoring etc.), token-based security (for supporting stateless APIs), CI/CD (providing continuous integration and continuous delivery), containerization, monitoring and notification (real time monitoring and notification), infrastructure as code, DevOps, Machine Learning and others. Microservices are assuming a pre-dominant role in integrations in modern digital applications. We have discussed and compared the key tenets of monolith applications with microservices based applications in Table 1.

Table 1 Tenets of Monolith and Microservices based architecture

	Monolith Architecture	**Microservice Architecture**
Flexibility	Less flexible as changes to large monolith tend to take time.	High flexibility as changes and extensions are easier in microservices due to modular service. Different microservices can be developed in various technologies, providing a flexible team competency model.

Release cycles	Normally monolith applications have larger release cycles as development and testing effort is high.	We can enable faster release cycles due to continuous integration and deployment of Microservices.
Extensibility	Difficult to add new functionalities or extend existing functionality. Monolith application is tightly coupled.	Easier to add new features and modify existing features. Loose coupling provides flexibility and extensibility.
Scalability	Session replication and complexity of the monolith application pose scalability challenges.	Microservices provide independent scalability wherein we can scale individual microservices. The stateless nature of microservices enhance scalability.
Agility	Larger cycles for development, deployment and release management.	Provides high operational agility due to continuous integration and continuous deployment.
Adaptation of new technologies	Challenges to adapt and use newer technologies.	Microservices can easily adapt to newer technologies. We can build newer microservices using different technologies such as Spring Boot, NodeJS.
Data Management	Centralized data. Uniform data model; no scope for using different database technologies.	Distributed data; each service can handle service specific data. Each service-specific database can have distinct data model.

Monolith vs Microservices vs WebService vs Serverless Architectures

The salient features of monolith application, microservices, SOAP based webservices and serverless technologies have been compared with each other in table 2.

Table 2 Monolith vs microservice vs webservice vs serverless

	Monolith Application	Microservice	Webservice (SOAP)	Serverless
Team size	Medium (about a 8 people team)	Small (about a 5 people team) model.	Medium (about a 8 people team)	Small (about a 5 people team)

Deployment	Longer release time and usually follows waterfall delivery model.	Shorter release time Continuous delivery, continuous deployment	Longer release time and usually follows waterfall delivery model.	Shorter release time, out of box deployment
Technology	Single technology stack	Supports multiple technologies	Mainly SOAP and REST	Cloud-based FaaS framework
Performance	High based on horizontal / vertical scaling	High due to independent scaling	Less compared to microservices	High due to in-built cloud support
Use case	Simple functionality, uniform technologies	Large scale complex applications, need for multiple technologies	Applications for legacy domains such as banking enterprise applications with complex workflow needs	Short-term process applications that need on-demand scalability.
Maintainability	High	Low	High	Low
Agility	Low	High	Low	High
Change Management	Entire application has to be redeployed	Only the impacted microservice has to be redeployed.	Entire service and dependencies need to be deployed	Minimal overhead
Responsibility	The monolith handles large complex task	Each of the microservices handle independent tasks	SOA architecture handles multiple business tasks	A serverless function handles single business function
Scalability	Less compared to microservices	Highly scalable due to independent deployment feature	Less compared to microservices	Auto scalable

Enterprise Technology Fitment Summary

As we have discussed and compared various architectures, we will layout the relevant scenarios and use cases that best fit these technologies.

Monolith architecture

Given below are the main use cases where monolith architecture can be used:

- Simple web applications that need minimal integrations with external interfaces can be built in a monolith manner
- Static websites, microsites, seasonal campaign sites, surveys and blog sites are few examples that fall in this category
- Applications that can be built with just a

single layer (such as UI layer)
- Applications that integrate with just a single system. For instance, a solution that is integrated with only a database
- In organizations where the organization culture comprises lengthy release cycles using a waterfall delivery model

Microservices architecture

Given below are the main use cases where we can use microservices architecture:

- Large and complex solution ecoystems that need to be deployed quickly to the market can leverage microservices.
- The enterprise solution that needs numerous integrations can leverage microservices architecture. The services layer can be designed with granular microservices to provide business functionality.
- B2C digital platforms, e-commerce platforms, digital marketing platforms and back-end for mobile apps can leverage microservices architecture
- The application that has multiple systems that interact through APIs or through headless integration
- Applications that need high agility and decoupling between layers can leverage microservices
- If the application eco-system needs multiple tools and technologies and multiple storage/ database systems, it can be best achieved through microservices architecture
- Real-time applications, analytics applications, event-based handling and big data applications can leverage microservices architecture

SOAP-based web services

Given below are the main use cases where SOAP-based web services architecture can be used:

- If existing legacy platforms and ERP systems have SOAP-based webservices, this can be leveraged in the in the service oriented architecture. Microservices can be built on top of traditional webservices to provide multi-speed IT model.
- B2B systems, ERP platforms and database sytems are common platforms to find SOAP-based webservices.

Serverless Computing

Given below are the main use cases where serverless architecture can be used:

- If the developers want to focus mainly on business logic and need managed infrastructure and platform services, serverless computing can be leveraged
- Real-time analytics applications, backup jobs, serverless websites, data processing jobs and event processing are some of the common use cases for serverless functions
- Orchestration or workflow steps can be implemented as serverless functions to provide on-demand execution and scaling
- Functions that process message streams, manipulation of multimedia content, process sensor messages, event processing jobs, batch jobs, audit jobs, IoT data processing jobs, database change capture jobs, real-time processing, ETL jobs, monitoring jobs, real-time notification jobs and chatbot jobs are ideal candidates for serverless functions
- Serverless functions can also be used for building auto-scalable mobile and web back end

About the Author

Dr. Shailesh Kumar Shivakumar has 18+ years of experience in a wide spectrum of digital technologies including, enterprise portals, content management systems, lean portals and microservices. Shailesh holds a PhD degree in computer science and has authored eight technical books published by the world's top academic publishers such as Elsevier Science, Taylor and Franscis, Wiley/IEEE Press and Apress. Shailesh has authored more than 12 technical white papers, five blogs, 12 research papers, eight textbook chapters and multiple articles.Shailesh holds two granted US patents, apart from five patent applications. He has successfully led severage large scale digital engagements for Fortune 500 clients. Shailesh can be reached at Shaileshkumar.Shivakumarasetty@mindtree.com