Application Modernization Introduction

WW Developer Advocate Team

IBM Developer

App modernization is inevitable

Evolution of application architectures

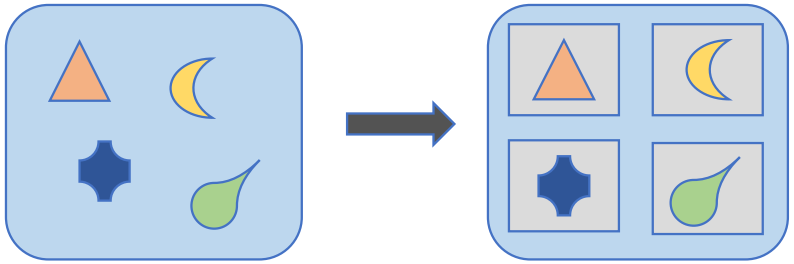
Late 90's	**Enterprise Application (EAI) Services and Models** Addressed integration and transactional challenges primarily by using message oriented middleware. Mostly proprietary systems needing a proliferation of custom interfaces.
Mid 00's	**Service Oriented Architectures** Based on open protocols like SOAP and WSDL making integration and adoption easier. Usually deployed on an Enterprise ESB which is hard to manage and scale.
Early 10's	**API Platforms and API Management** REST and JSON become the defacto standard for consuming backend data. Mobile apps become major consumers of backend data. New Open protocols like OAuth become available further simplifying API development .
2015 and beyond	**Microservice Architecture** Applications are composed of small, independently deployable processes communicating with each other using language-agnostic APIs and protocols.

Why microservices ?

Efficient teams	Simplified deployment	Right tools for the job	Improved application quality	Scalability
• End to end team ownership of relatively small codebases	• Each service is individually changed, tested, and deployed without affecting other services	• Teams can use best of breed technologies, libraries, languages for the job at hand	• Services can be tested more thoroughly in isolation	• Services can be scaled independently at different rates as needed
➢ Teams can innovate faster and fix bugs more quickly	➢ Time to market is accelerated.	➢ Leads to faster innovation	➢ Better code coverage	➢ Leads to better overall performance at lower cost

Key tenets of a microservices architecture

1. **Large monoliths are broken down into many small services**
   - Each service runs into its own process
   - Generally accepted rule is one service per container
2. **Services are optimized for a single function**
   - One business function per service
     - The service will have only one reason to change
3. **Services are tightly encapsulated behind concrete programming interfaces**
   - A balance between evolving the interface and maintaining backward compatibility
4. **Communication via REST API and/or message brokers**
   - Avoids tight coupling and allows for flexibility of synchronous and asynchronous access
5. **Per-service continuous integration and continuous deployment (CI/CD)**
   - Services can evolve at different rates
6. **Per-service HA and clustering**
   - Services can be scaled independently at different rates as needed



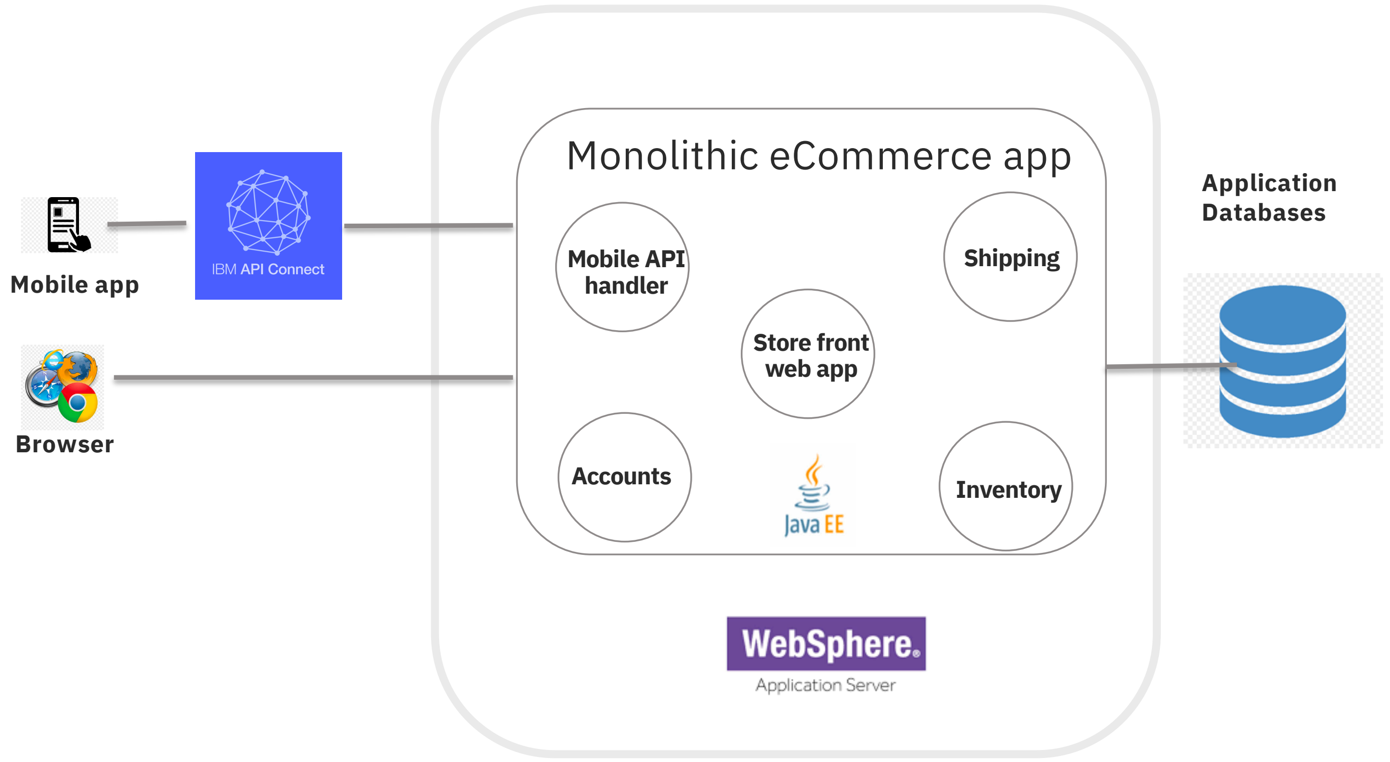
Microservices Architecture
Cultural change considerations

- **Smaller teams with broader scope**
  - Mini end to end development orgs in each team vs large silos across the entire development team
- **Top down support with bottom up execution**
  - Change can't happen effectively w/o executive sponsorship
  - Change needs to be executed at the smallest organizational unit to take hold
- **Teams own all metrics related to operations and development**
  - Have to minimize downtime + number of bugs while also maximizing the rate at which needed features are added and minimizing the time to market of those new features
- **Trust**
  - Teams need to build trust with other teams that they collaborate with rather than relying on one size fits all checklists and rules
- **Reward based on results not compliance**
  - Cultures only change when people are measured and rewarded for outcomes consistent with the changes
  - Smaller more autonomous teams work better with less central micromanagement and more focus on broad measurable goals

Example microservice transformation

Monolithic eCommerce app

- Store front web interface
- Customer Accounts
- Inventory
- Shipping
- Back end for mobile app



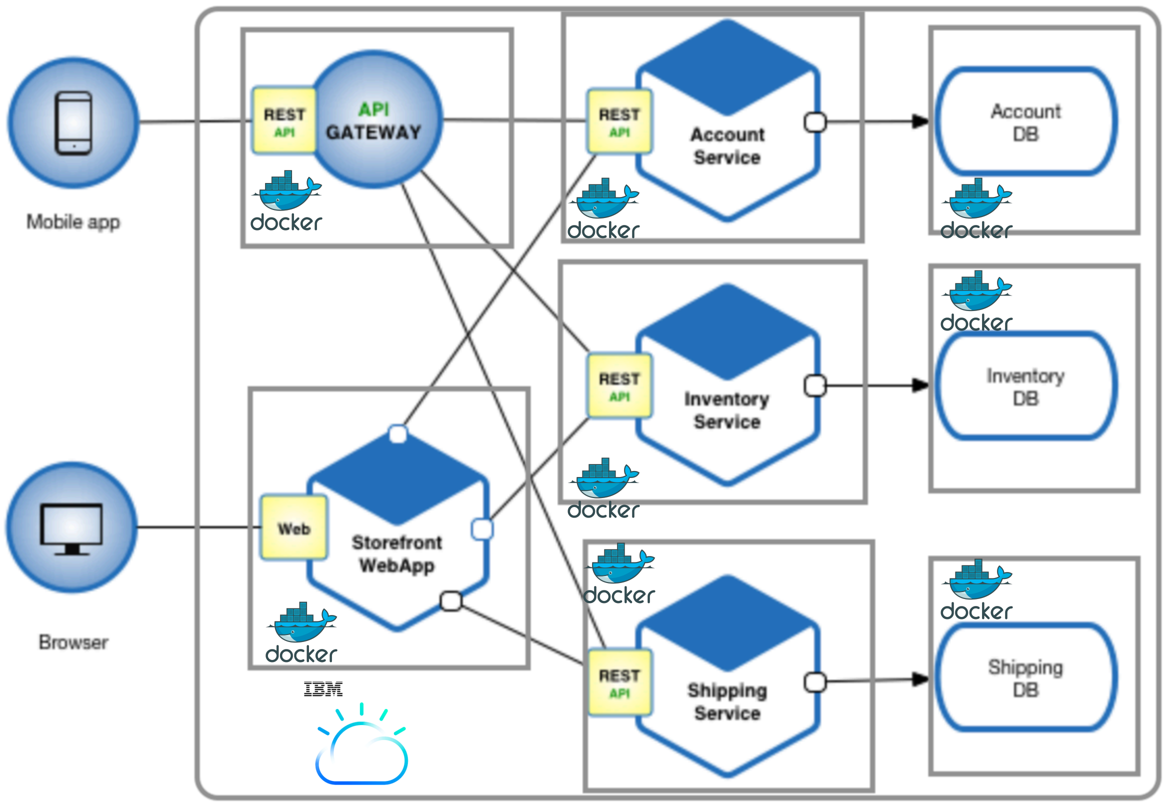
An eCommerce Java EE app on Websphere

Transformed application

Key technologies

- Containers (Docker)
- Container orchestration (Kubernetes)
- 12-Factor Best Practices
- CI/CD tools (e.g Jenkins)
- (IBM) Transformation Adv

Kubernetes Cluster



An eCommerce microservices app on a Kubernetes cluster

IBM Developer