A Microservices Approach to Implementing RESO Web API

Peter Jang
Chief Architect
@ dynaConnections Corp
Agenda

- Modern-day Requirements for Enterprise Software
- Traditional Web Applications Architecture
- Problems
- The Microservices Solution
- Real-world Examples
- Demo
Modern-day Requirements

• Short Turnaround Time for Fixes, Updates, Enhancements

• More demands from existing endusers

• Future/Anticipated Consumers
  • Increasing needs from IoT
  • Startups waiting in the wings
Traditional Web Application Architecture

Simple to develop, test, deploy and scale
Problems

• Gets large quickly
• Layers start merging
• Customer demands > cut corners to meet
• Long QA cycles for simple fixes/updates
• Fear of change
• Long term commitment to a technology stack
• Large Technical Debt
TECHNICAL DEBT IS COMING
What is technical debt?

• From Wiki, Technical debt is a metaphor referring to the eventual consequences of any system design, software architecture or software development within a codebase

• Nights spent deploying massive updates

• Wasted time spent doing manual QA

• Hours spent merging code and debugging

• more…
Solution - Microservices

What is a Microservice?

- From Wiki, Microservices is a software architecture style in which complex applications are composed of small, independent processes communicating with each other using language-agnostic APIs.

- Martin Fowler’s definition: a component that is
  - independently replaceable
  - independently upgradable
Solution - Microservices (contd.)
Benefits of Microservices

- Shorter dev/QA cycles
- Enables/empowers Agile development
- Less likely to create coupling/debt
- Flexible scaling options/Resiliency
- Ability to try and incorporate newer technology
- “Front-end of choice”, “datashare” environments
Microservices with Traditional Applications

- Traditional Webapp
- Load Balancer/API Gateway/Service Discovery
- ODATA Service
- RETS 1.X Service
- Listings Service
- Media Service
- Membership Service
- Authentication Service
- Event Bus/Message Queue
- SQL
- NOSQL
The Scale Cube

X Axis – Horizontal Duplication

Y Axis – Split by Function or Service

Near Infinite Scale

Z Axis – Lookup Oriented Splits

Start by service or similar information

Many systems, each a clone and load balanced.

Further slice data or users

One, monolithic System/Service

The ART OF SCALABILITY
Scalable Web Architecture, Processes, and Organizations for the Modern Enterprise
SECOND EDITION

MARTIN L. ABBOTT  MICHAEL T. FISHER
FOREWORD BY BRIAN DARMIS, Managing Director, Capital One, and a Three-time All-Pro NFL Lineman

DynaConnections
CORPORATION
What’s the catch?

• Complexity

• Strong Devops Team
  • Debt moves from Developers to Operations

• Requires more automation and monitoring
• Netflix
• Amazon
• PayPal
• Uber
• Linked In
• more…
“Thank You”