

Introduction to Web Application Architecture

Well begun is $\frac{1}{2}$ done

Architecture

- ▶ **Architecture** is an abstract plan that can include design patterns, modules, and their interactions.

In this course we will focus on

- ▶ Architecture **Implementation or Realization**
which incorporates
- ▶ **Frameworks** - *architected* "physical" structures on which you build your application.
specifically we will use
- ▶ The **Spring Framework**, an **Enterprise Application Development** environment for building large scale enterprise applications.

Web Application Architecture

This course is concerned with the realization of **scalable web applications** as defined by a **Web Application Architecture**.

A Web Application Architecture
is a **part** of an **Enterprise Architecture**

A Scalable Web Application is an Enterprise Application...

An application - that is large & complex –
well beyond the individual or small business use case.

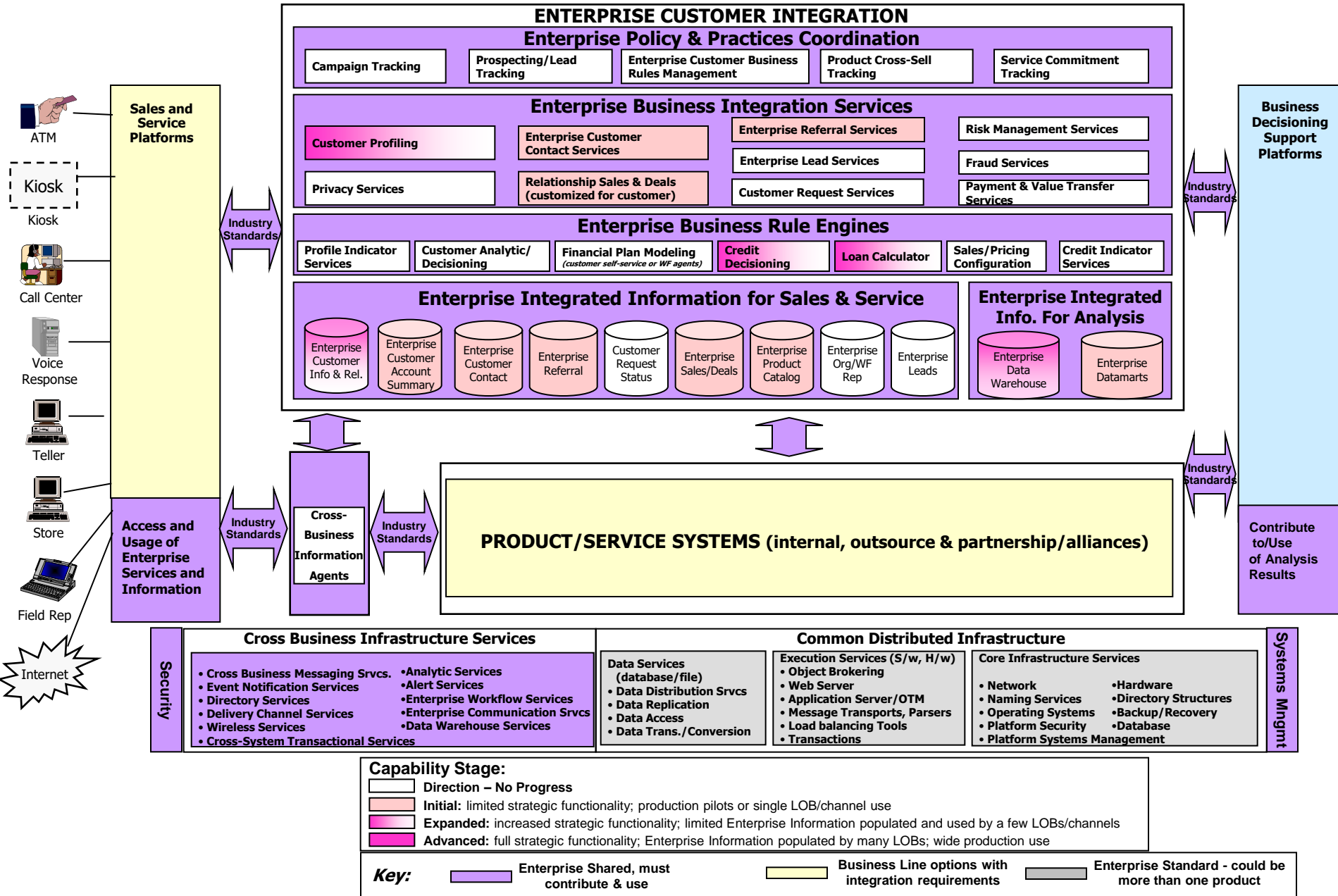
- “...for creating large-scale business applications, you need ...[to understand] the design and architecture of enterprise applications...”

▶ [Enterprise Design & Architecture - Microsoft](#)

Enterprise Application

- **Large**
 - Large-scale, multi-tiered, scalable, reliable, and secure network applications. Designed to solve the problems encountered by large enterprises.
- **Business Oriented**
 - Meets specific business requirements; business policies, processes, rules, and entities.
- **Mission Critical**
 - Sustain continuous operation, scalable and deployment, provide for maintenance, monitoring, and administration.

Enterprise Architecture

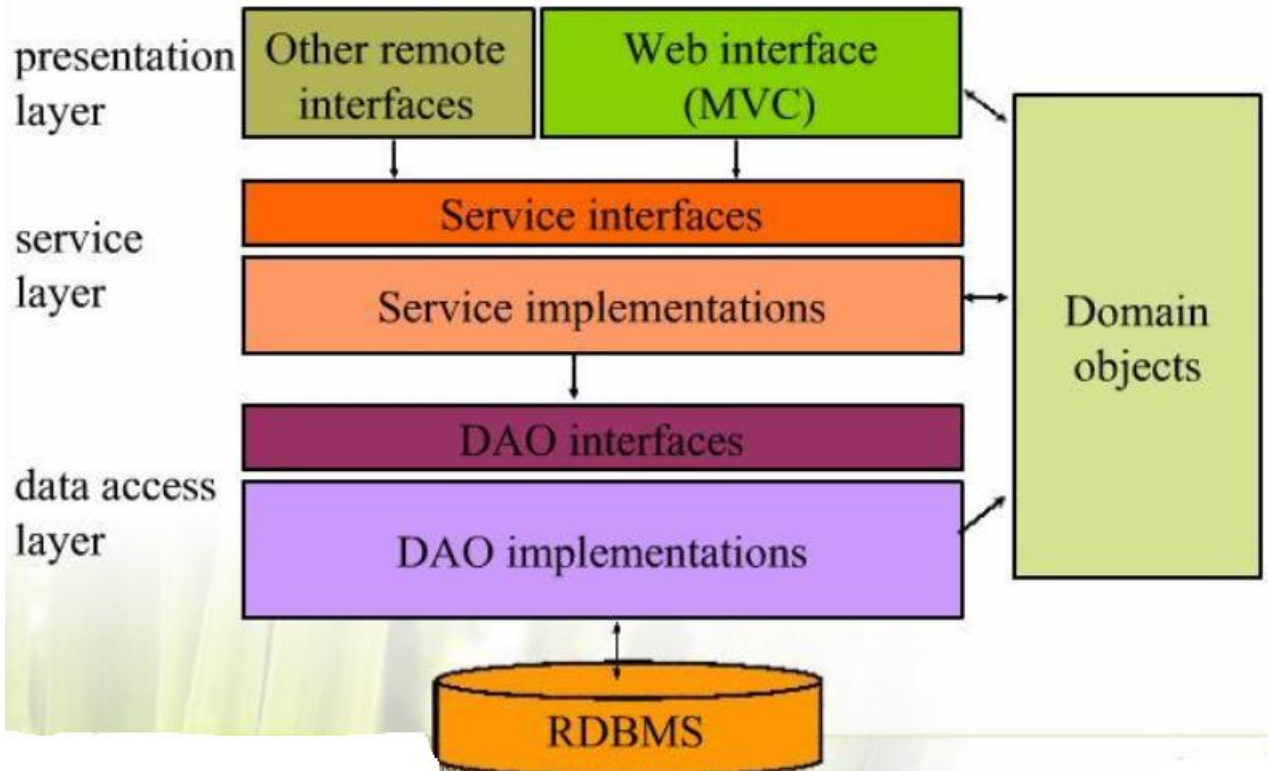


Underlying N-Tier Software Architecture

Separation of Concerns

Domain Model
central, organizing component

Typical application layering



Design to Interfaces

“Types” of N-Tier architectures

▶ **Monolith**

- ▶ Single Project
- ▶ Single Presentation layer
- ▶ Boundaries between tiers “blur” over time

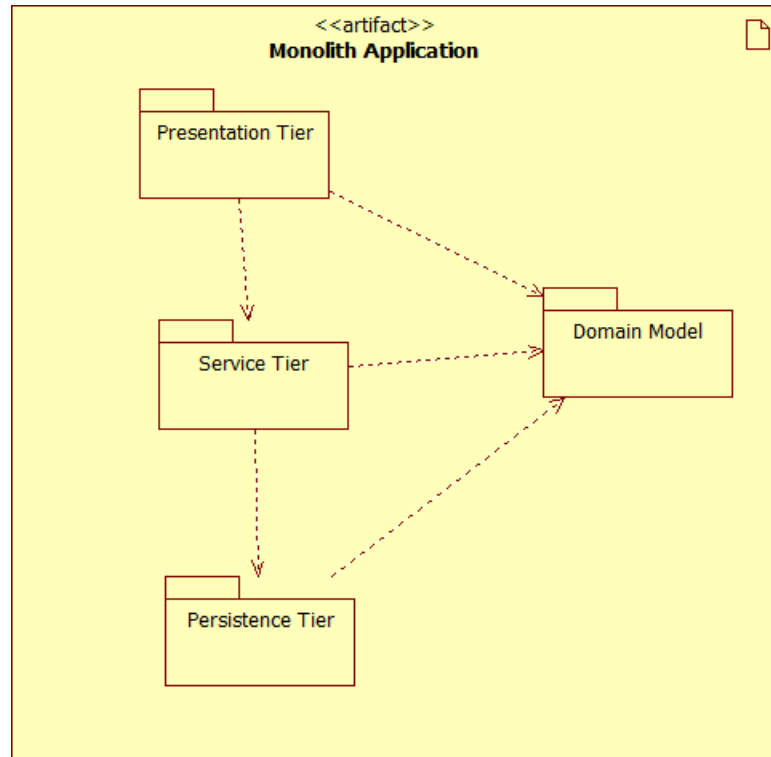
▶ **Technical Functional Layering**

- ▶ Project per functional layer [Presentation, Service, Persistence, Domain]
- ▶ Increase re-use
- ▶ Clean layer separation
- ▶ more flexible....scalable

▶ **Component Services Business**

- ▶ Project per business domain
- ▶ “Services” oriented

Monolith N-Tier



EAExample

src/main/java

edu.mum.controller

- ControllerExceptionHandler.java
- HomeController.java
- LoginController.java
- MemberController.java

edu.mum.dao

- CredentialsDao.java
- GenericDao.java
- MemberDao.java

edu.mum.dao.impl

edu.mum.domain

- Authority.java
- Credentials.java
- Member.java

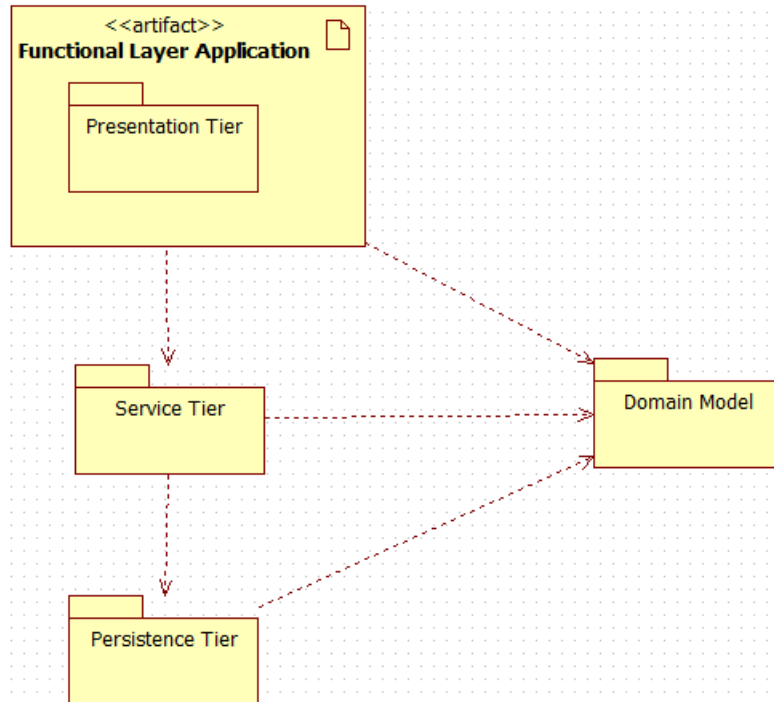
edu.mum.main

edu.mum.service

- CredentialsService.java
- MemberService.java

edu.mum.service.impl

Functional N-Tier



FunctionalExample

- src/main/java
 - mum.edu.controller
 - ControllerExceptionHandler.java
 - HomeController.java
 - LoginController.java
 - MemberController.java
 - mum.edu.interceptor
- src/main/resources

EAExampleService

- src/main/java
 - edu.mum.service
 - CredentialsService.java
 - MemberService.java
 - edu.mum.service.impl
- src/main/resources

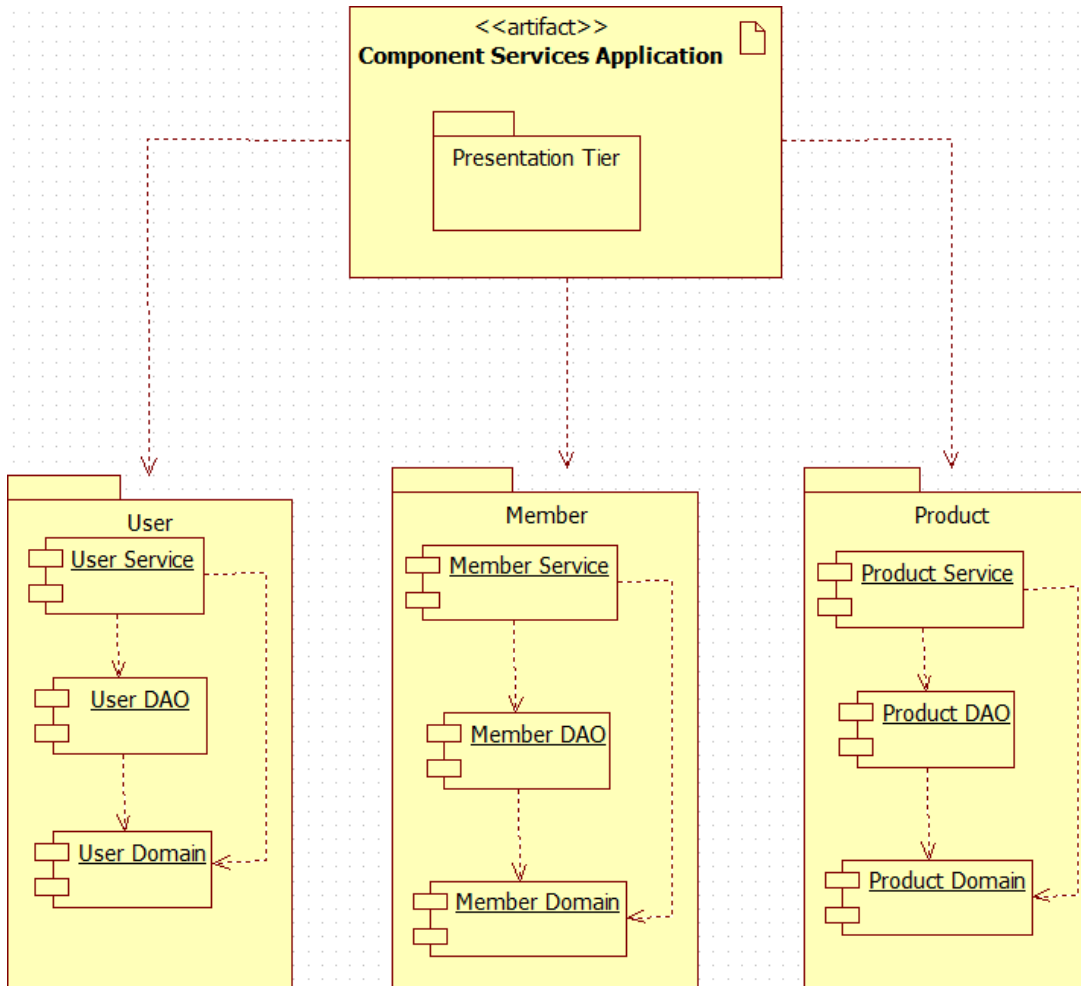
EAExampleDomain

- src/main/java
 - edu.mum.domain
 - Authority.java
 - Credentials.java
 - Member.java
- src/main/resources

EAExampleRepository

- src/main/java
 - edu.mum.dao
 - CredentialsDao.java
 - GenericDao.java
 - MemberDao.java
 - edu.mum.dao.impl
- src/main/resources

Component N-Tier



- ComponentExample
 - src/main/java
 - mum.edu.controller
 - ControllerExceptionHandler.java
 - HomeController.java
 - LoginController.java
 - MemberController.java
 - mum.edu.interceptor
 - src/main/resources

- ComponentSecurity
 - src/main/java
 - mum.edu.domain
 - Authority.java
 - Credentials.java
 - mum.edu.repository
 - CredentialsDao.java
 - mum.edu.service
 - CredentialsService.java
 - mum.edu.service.impl
 - src/main/resources

MVC & the N-tier architecture

MVC is the primary “pattern” associated with the presentation tier

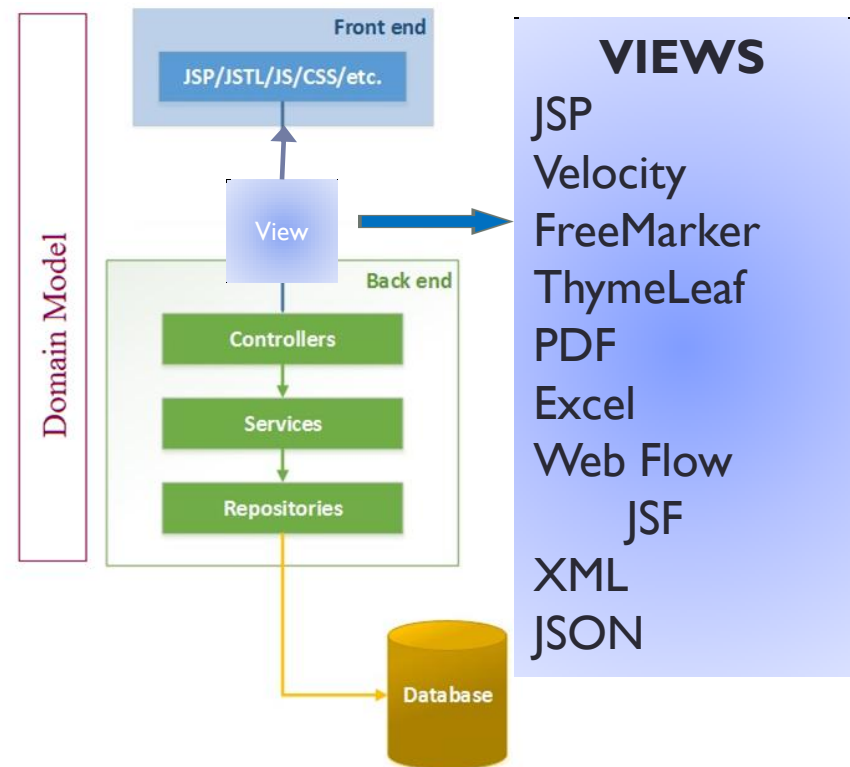
It is commonly identified as
page rendering on the server

The growth of the **Consumer
Web[2.0]** has emphasized
alternative solutions

Specifically:

SPA & microservices
Technologies

“Classic” Spring MVC



SPA & Microservices Definitions

- ▶ A SPA is a web application or web site that interacts with the user by dynamically rewriting the current page rather than loading entire new pages from a server. This approach avoids interruption of the user experience between successive pages, making the application behave more like a desktop application.
- ▶ Microservices is a variant of the service-oriented architecture (SOA) architectural style that structures an application as a collection of loosely coupled services. In a microservices architecture, services should be fine-grained and the protocols should be lightweight.
 - ▶ single business goal
 - ▶ simple, well-defined communication interface
 - ▶ runs a unique process
 - ▶ manages[usually] its **own database**.

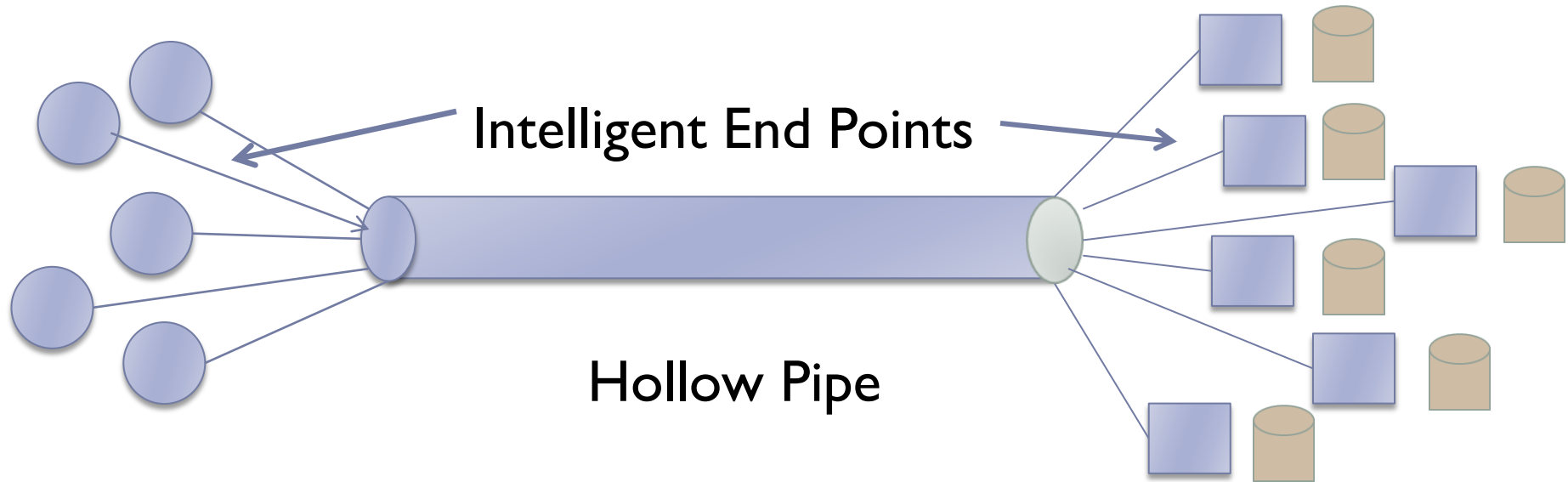
SPA technologies[JavaScript MVC]
AngularJS, Backbone.js,
Ember.js, React, Vue.js

“Modern” High End

SPAs

Consumer Web [2.0]

Micro-Services



High Page Volume Ultra Many Users

“Primary” Candidates

Internet companies – web page IS product

Facebook, Instagram, Twitter, AccuWeather

Internet Companies – with “products to sell”

Netflix, Amazon

Hollow Pipe/Services

Node.js, Play2.0/Scala
Spring Framework/Microservices

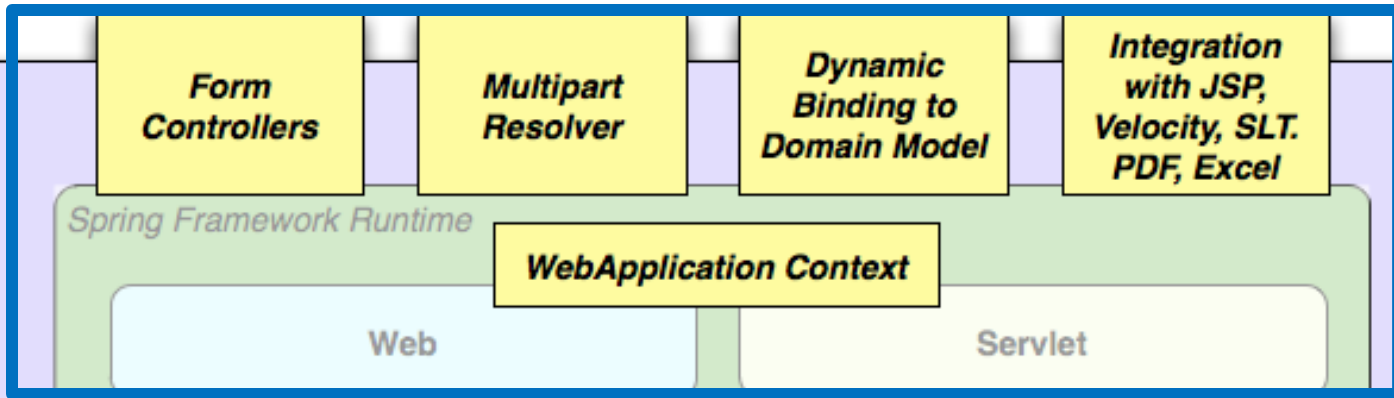


Spring Model Architecture

CS 545 WAA Focus



85%



15%

