INTRODUCTION TO ENTERPRISE ARCHITECTURE

Home of All the Laws of Nature

Enterprise Architecture

- An overall architectural vision for an organization
- An architecture in which the system in question is the ***whole enterprise***, especially the ***business processes***, ***technologies***, and ***information systems*** .
- Contains but is not limited to :

  Process architecture

  Applications architecture

  Security architecture

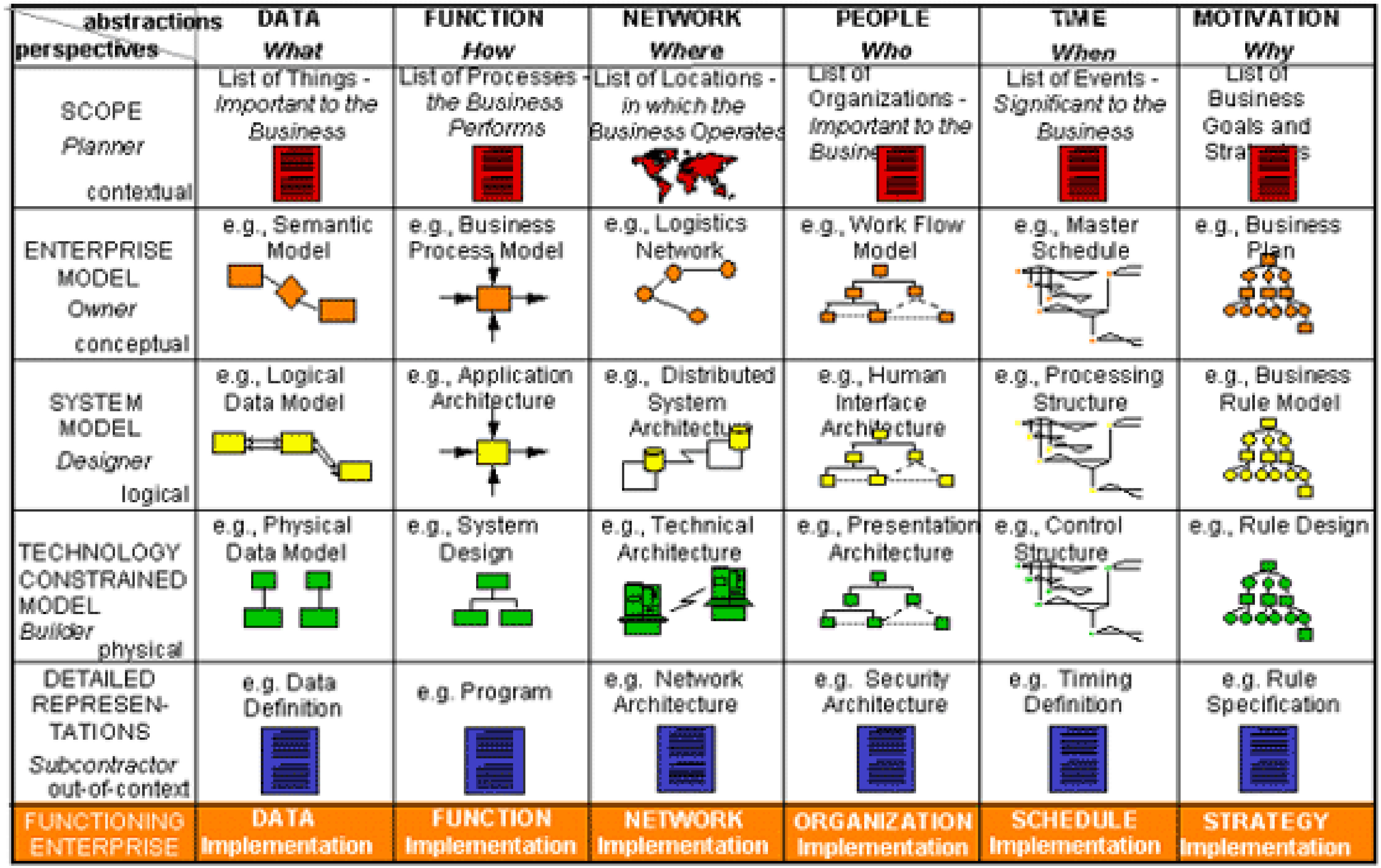
  Technology architecture

It's Not Just the Enterprise IT

And what about software architecture, system architecture, solution architecture, infrastructure architecture…

Zachman Enterprise Architecture Framework

One of Many...



abstractions perspectives	DATA *What*	FUNCTION *How*	NETWORK *Where*	PEOPLE *Who*	TIME *When*	MOTIVATION *Why*
SCOPE *Planner* contextual	List of Things - *Important to the Business*	List of Processes - *the Business Performs*	List of Locations - *in which the Business Operates*	List of Organizations - *Important to the Business*	List of Events - *Significant to the Business*	List of Business Goals and Strategies
ENTERPRISE MODEL *Owner* conceptual	e.g., Semantic Model	e.g., Business Process Model	e.g., Logistics Network	e.g., Work Flow Model	e.g., Master Schedule	e.g., Business Plan
SYSTEM MODEL *Designer* logical	e.g., Logical Data Model	e.g., Application Architecture	e.g., Distributed System Architecture	e.g., Human Interface Architecture	e.g., Processing Structure	e.g., Business Rule Model
TECHNOLOGY CONSTRAINED MODEL *Builder* physical	e.g., Physical Data Model	e.g., System Design	e.g., Technical Architecture	e.g., Presentation Architecture	e.g., Control Structure	e.g., Rule Design
DETAILED REPRESENTATIONS *Subcontractor* out-of-context	e.g. Data Definition	e.g. Program	e.g. Network Architecture	e.g. Security Architecture	e.g. Timing Definition	e.g. Rule Specification
FUNCTIONING ENTERPRISE	DATA Implementation	FUNCTION Implementation	NETWORK Implementation	ORGANIZATION Implementation	SCHEDULE Implementation	STRATEGY Implementation

Enterprise SOFTWARE Application

- **[Moderate to] Large**

  Multi-tiered, scalable, reliable, and secure network applications. Designed to solve 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 Design & Architecture - Microsoft

Software Architecture – Application Frameworks

- **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.

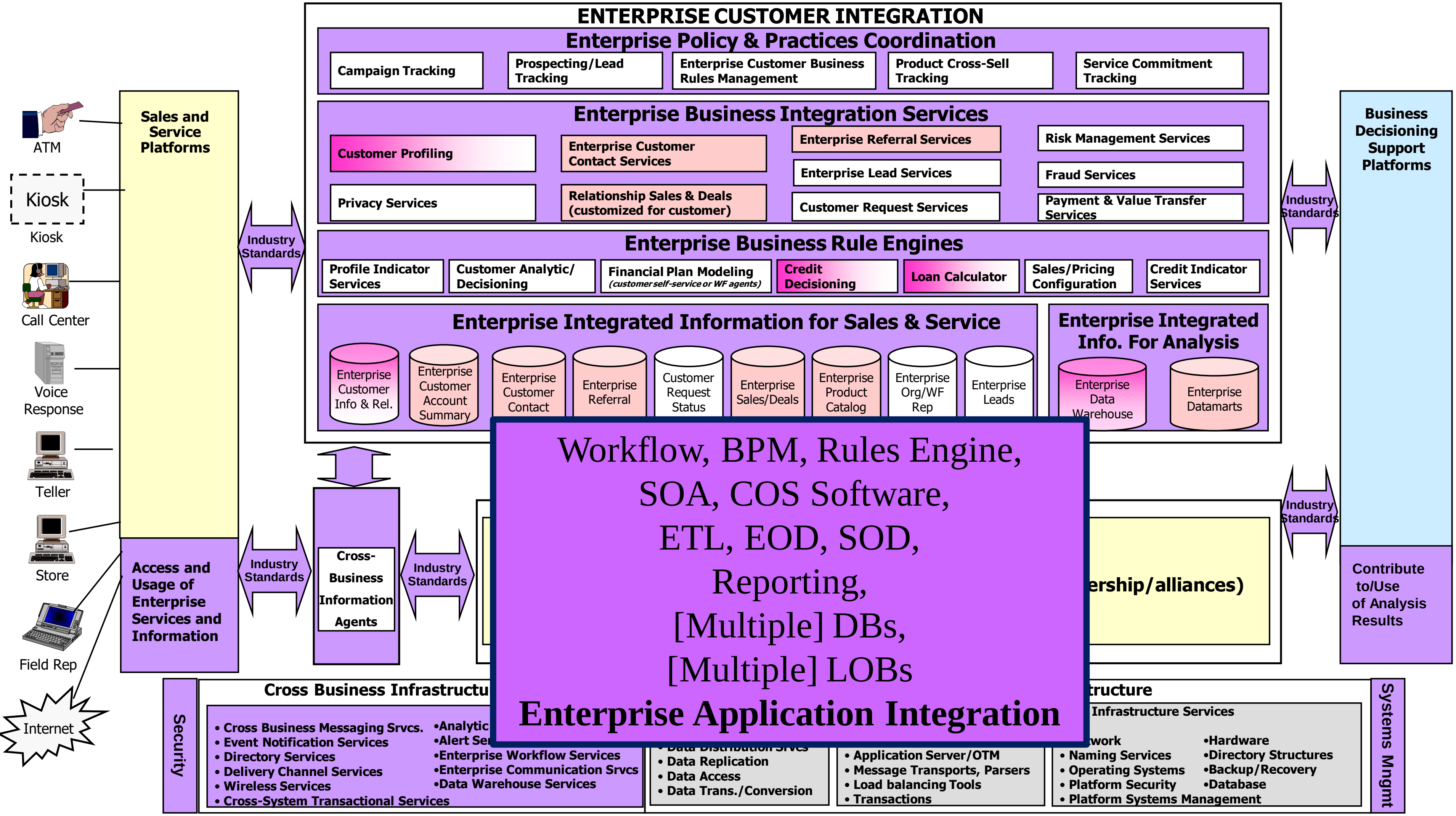
specifically we will use

- The **Spring Framework**, an **Enterprise**

  Development environment for buildin
  enterprise applications.

Other Frameworks:
.NET
LAMP
Ruby-on-Rails
Grails
Jboss Seam
Google Guice
JEE 7 Container

LARGE Enterprise Architecture



ENTERPRISE CUSTOMER INTEGRATION

Enterprise Policy & Practices Coordination

- Campaign Tracking
- Prospecting/Lead Tracking
- Enterprise Customer Business Rules Management
- Product Cross-Sell Tracking
- Service Commitment Tracking

Enterprise Business Integration Services

- Customer Profiling
- Privacy Services
- Enterprise Customer Contact Services
- Relationship Sales & Deals (customized for customer)
- Enterprise Referral Services
- Enterprise Lead Services
- Customer Request Services
- Risk Management Services
- Fraud Services
- Payment & Value Transfer Services

Enterprise Business Rule Engines

- Profile Indicator Services
- Customer Analytic/ Decisioning
- Financial Plan Modeling *(customer self-service or WF agents)*
- Credit Decisioning
- Loan Calculator
- Sales/Pricing Configuration
- Credit Indicator Services

Enterprise Integrated Information for Sales & Service

- Enterprise Customer Info & Rel.
- Enterprise Customer Account Summary
- Enterprise Customer Contact
- Enterprise Referral
- Customer Request Status
- Enterprise Sales/Deals
- Enterprise Product Catalog
- Enterprise Org/WF Rep
- Enterprise Leads

Enterprise Integrated Info. For Analysis

- Enterprise Data Warehouse
- Enterprise Datamarts

Cross Business Infrastructu

- Cross Business Messaging Srvcs.
- Event Notification Services
- Directory Services
- Delivery Channel Services
- Wireless Services
- Cross-System Transactional Services

- Analytic
- Alert Se
- Enterprise Workflow Services
- Enterprise Communication Srvcs
- Data Warehouse Services

- Data Distribution Srvcs
- Data Replication
- Data Access
- Data Trans./Conversion

- Application Server/OTM
- Message Transports, Parsers
- Load balancing Tools
- Transactions

tructure

Infrastructure Services

- twork
- Naming Services
- Operating Systems
- Platform Security
- Platform Systems Management

- Hardware
- Directory Structures
- Backup/Recovery
- Database

Workflow, BPM, Rules Engine,
SOA, COS Software,
ETL, EOD, SOD,
Reporting,
[Multiple] DBs,
[Multiple] LOBs

Enterprise Application Integration

Capability Stage:

□ **Direction – No Progress**

▨ **Initial:** limited strategic functionality; production pilots or single LOB/channel use

▧ **Expanded:** increased strategic functionality; limited Enterprise Information populated and used by a few LOBs/channels

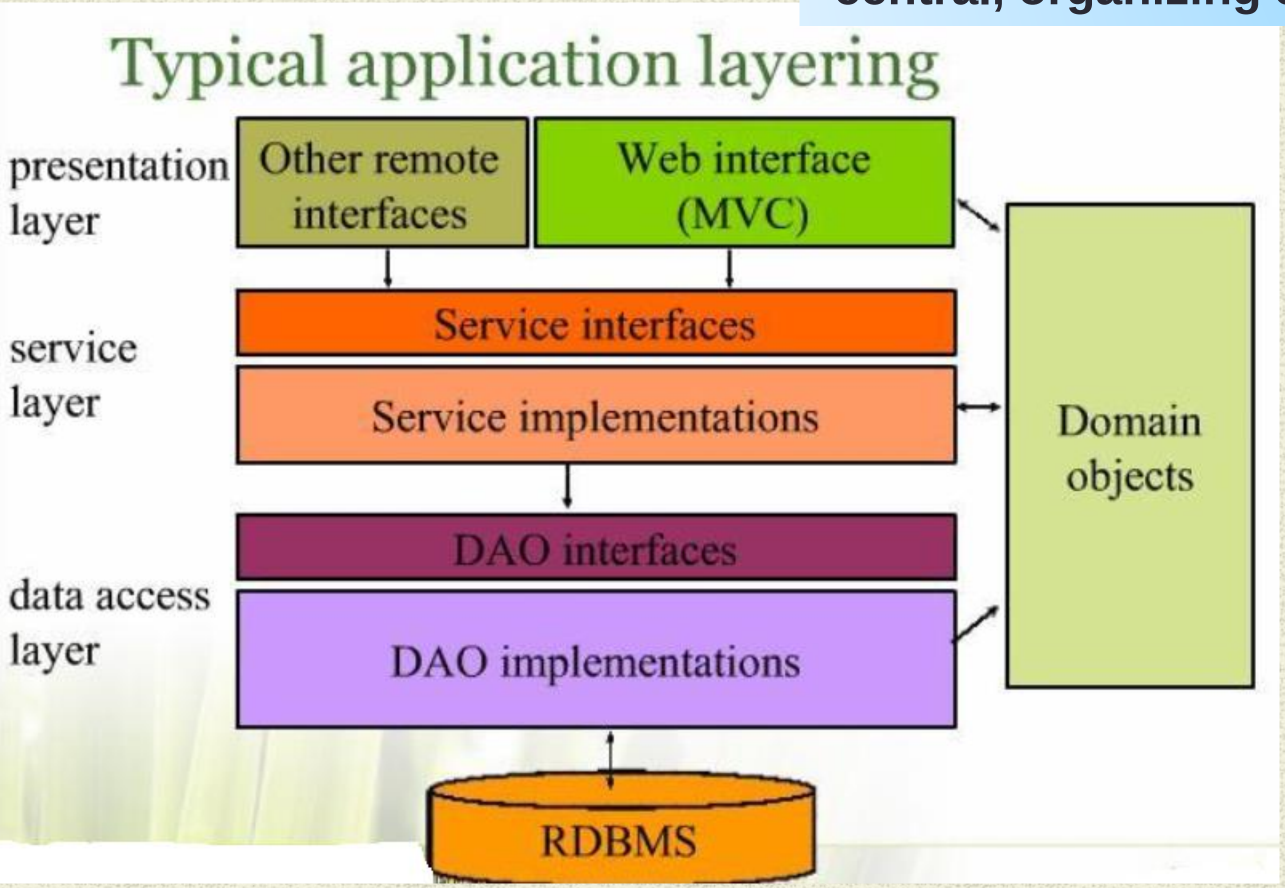
■ **Advanced:** full strategic functionality; Enterprise Information populated by many LOBs; wide production use

Key: 🟪 Enterprise Shared, must contribute & use 🟨 Business Line options with integration requirements ▩ Enterprise Standard - could be more than one product

Underlying N-Tier Software Architecture

Separation of Concerns

Domain Model
central, organizing component



Design to Interfaces

Service Layer – Interface driven

ALWAYS design to Interfaces
For Service Layer -- Extra important

Extremely Important

```
public interface MemberService {
        public void save(Member member);
        public void update(Member member);
        public List<Member> findAll();
        public Member findByMemberNumber(Integer memberId);
}
```

Interface Driven
Basic Design Pattern
Separation of Concerns
Testability
Scalability
Adaptability

EXTRA -- Extremely Important

Implementation of MemberService

```
@Service
@Transactional
public class MemberServiceImpl implements MemberService {
 @Autowired
 private MemberDao memberDao;

    public void save( Member member) {
        memberDao.save(member);
      }
    public void update( Member member) {
       memberDao.update(member);
    }
    public List<Member> findAll() {
      return (List<Member>)memberDao.findAll();
    }
    public Member findByMemberNumber(Integer memberId) {
     return memberDao.findByMemberNumber(memberId);
    }
```

Spring Annotations to facilitate Application Management

“Auto-magic” Dependency Injection

Interface driven Data Access Layer

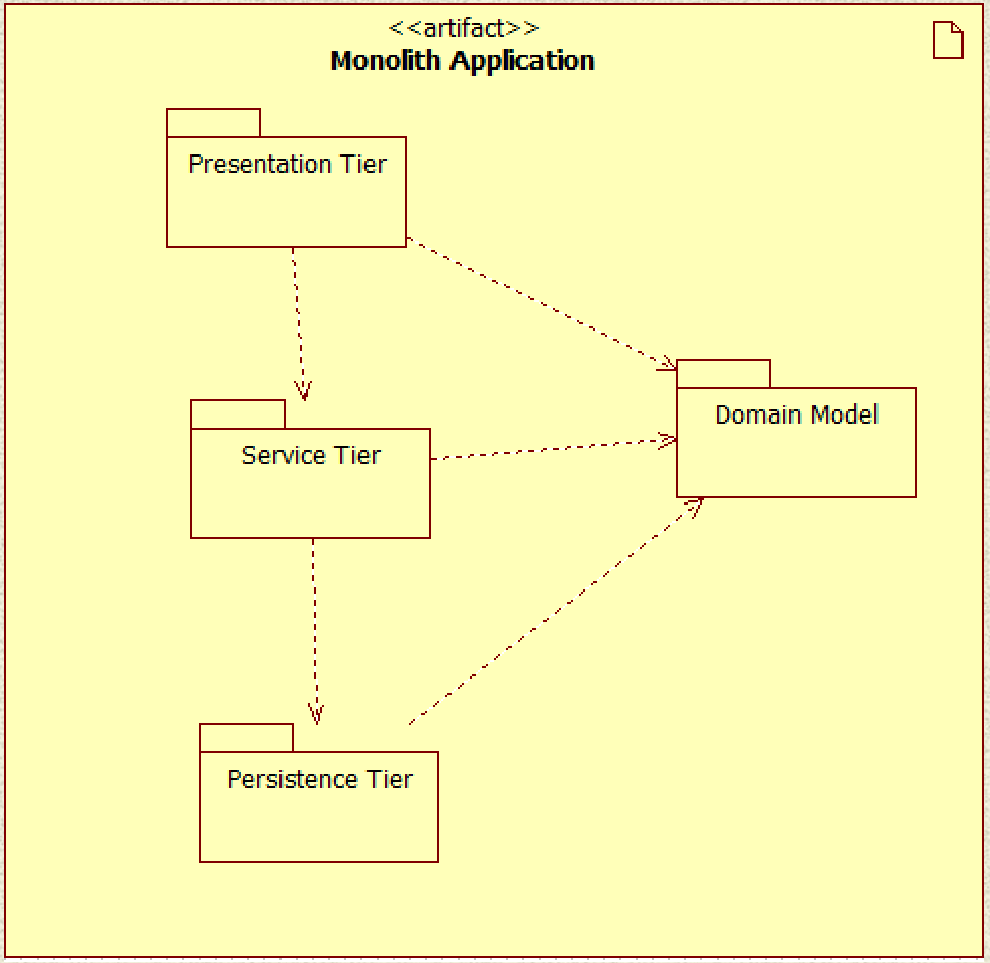
RESTful Implementation of MemberService

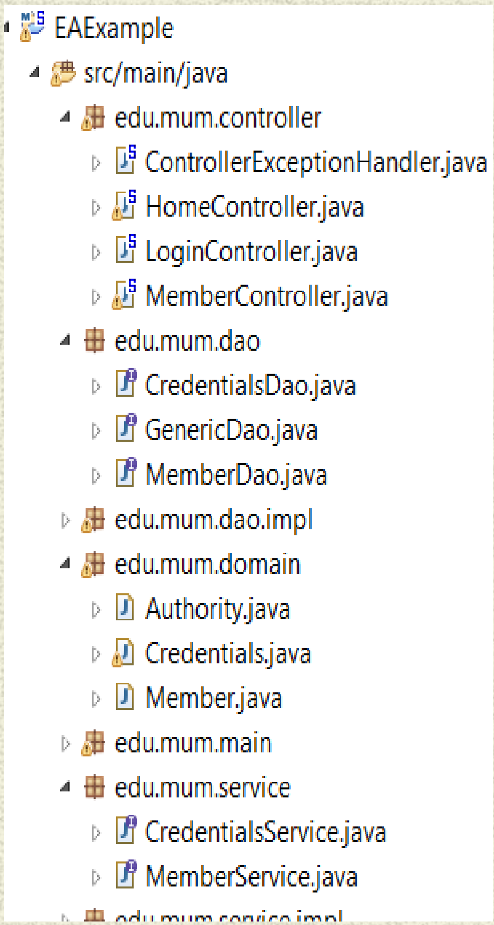
```
@Service
public class MemberRestServiceImpl implements MemberService {
 @Autowired
 private MemberRestService memberRestService;

    public void save( Member member) {
       memberRestService.save(member);
      }
    public void update( Member member) {
       memberRestService.update(member);
    }
    public List<Member> findAll() {
      return (List<Member>) memberRestService.findAll();
    }
    public Member findByMemberNumber(Integer memberId) {
     return memberRestService.findByMemberNumber(memberId);
    }
```

“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

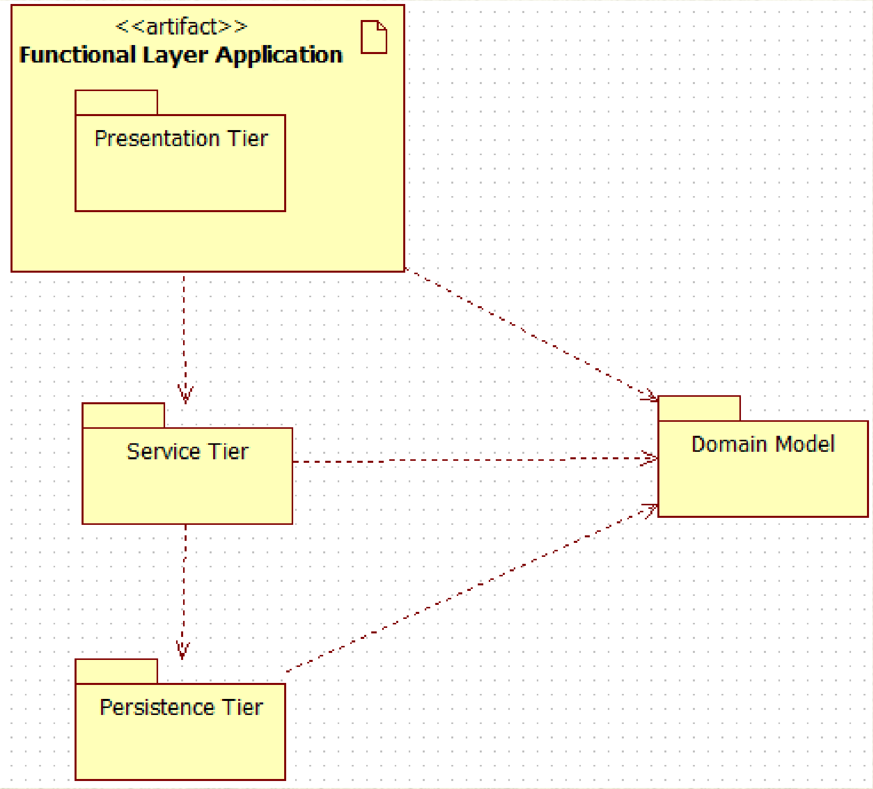
Monolith N-Tier

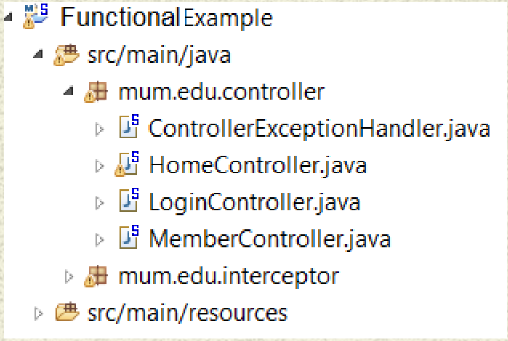




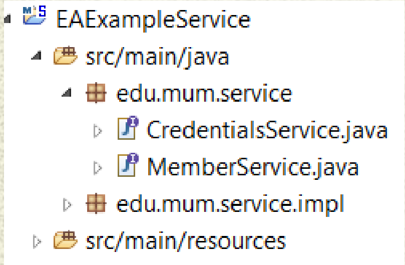
```
EAExample
  src/main/java
    edu.mum.controller
      ControllerExceptionHandler.java
      HomeController.java
      LoginController.java
      MemberController.java
    edu.mum.dao
      CredentialsDao.java
      GenericDao.java
      MemberDao.java
    edu.mum.dao.impl
    edu.mum.domain
      Authority.java
      Credentials.java
      Member.java
    edu.mum.main
    edu.mum.service
      CredentialsService.java
      MemberService.java
    edu.mum.service.impl
```

Functional N-Tier

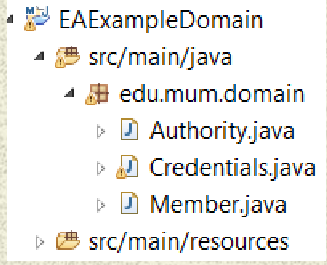




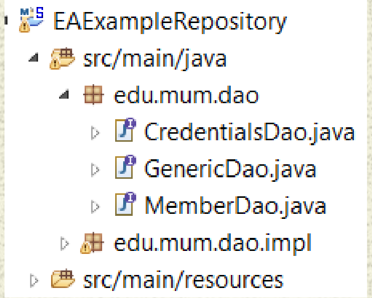
```
FunctionalExample
  src/main/java
    mum.edu.controller
      ControllerExceptionHandler.java
      HomeController.java
      LoginController.java
      MemberController.java
    mum.edu.interceptor
  src/main/resources
```



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

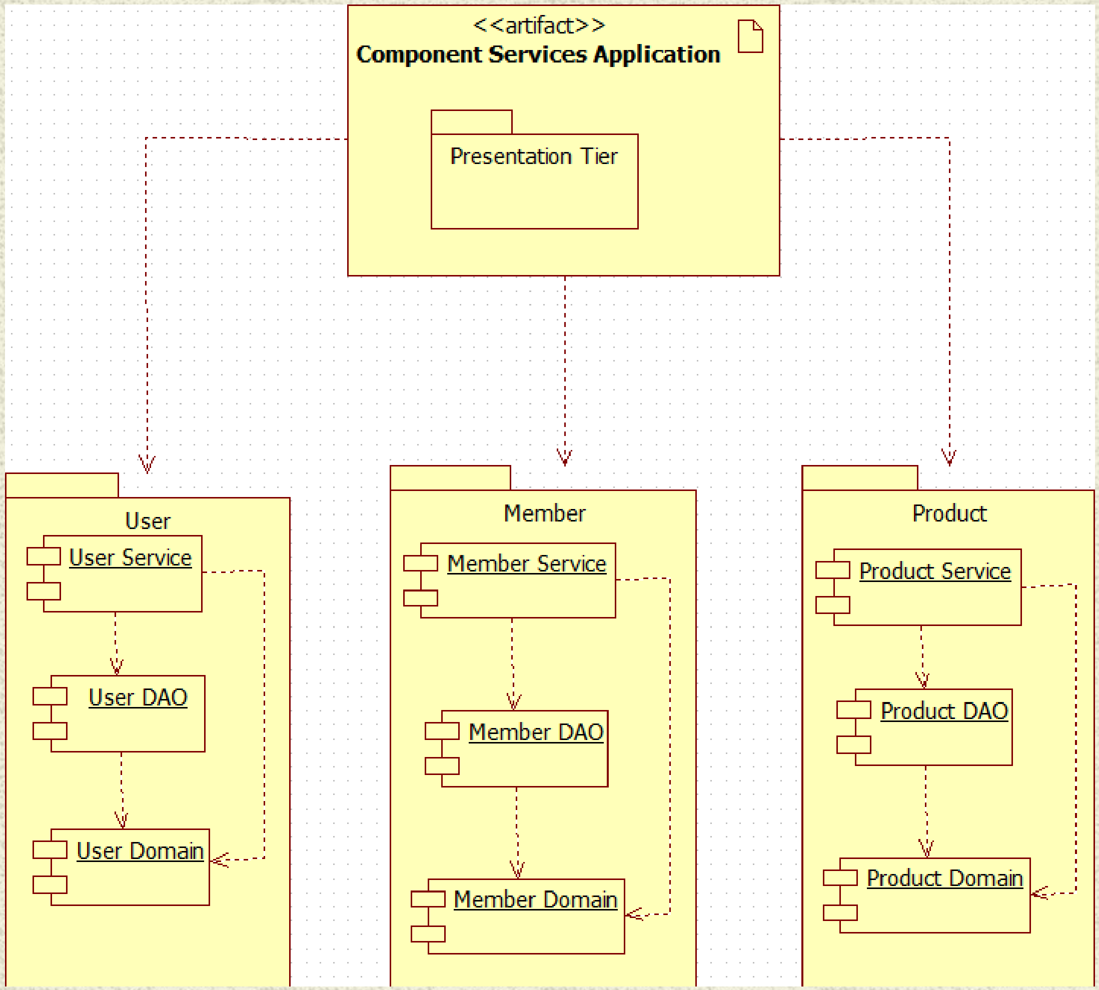


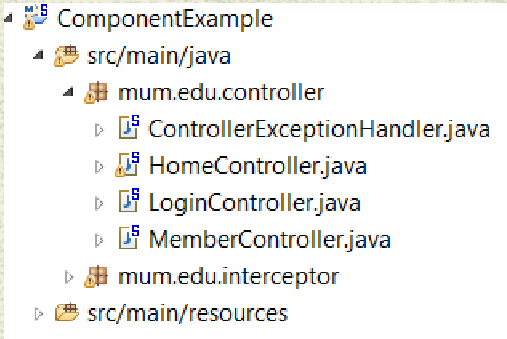
```
EAExampleDomain
  src/main/java
    edu.mum.domain
      Authority.java
      Credentials.java
      Member.java
  src/main/resources
```



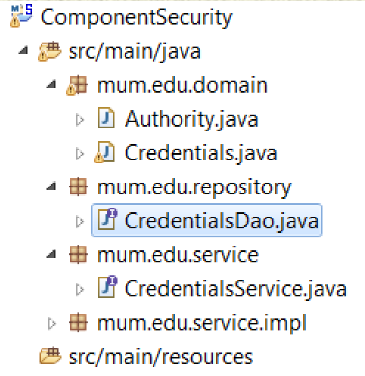
```
EAExampleRepository
  src/main/java
    edu.mum.dao
      CredentialsDao.java
      GenericDao.java
      MemberDao.java
    edu.mum.dao.impl
  src/main/resources
```

Component N-Tier

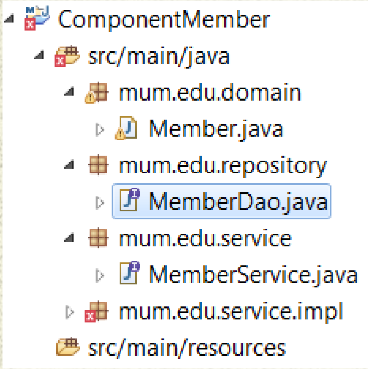




```
◢ ComponentExample
  ◢ src/main/java
    ◢ mum.edu.controller
      ▷ ControllerExceptionHandler.java
      ▷ HomeController.java
      ▷ LoginController.java
      ▷ MemberController.java
    ▷ mum.edu.interceptor
  ▷ src/main/resources
```



```
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
```



```
◢ ComponentMember
  ◢ src/main/java
    ◢ mum.edu.domain
      ▷ Member.java
    ◢ mum.edu.repository
      ▷ MemberDao.java
    ◢ mum.edu.service
      ▷ MemberService.java
    ▷ mum.edu.service.impl
  src/main/resources
```

Core N-Tier Enterprise Architecture Position Statement

Corporate Enterprise Environments are an

Of Technologies

A “Java/Spring” shop is “maybe” 70-80% Java

New technologies arise to solve new use cases

Example: Consumer Web [2.0]

However, a consumer-facing technology is not necessarily the solution for core enterprise software infrastructure

So Our Focus will be:

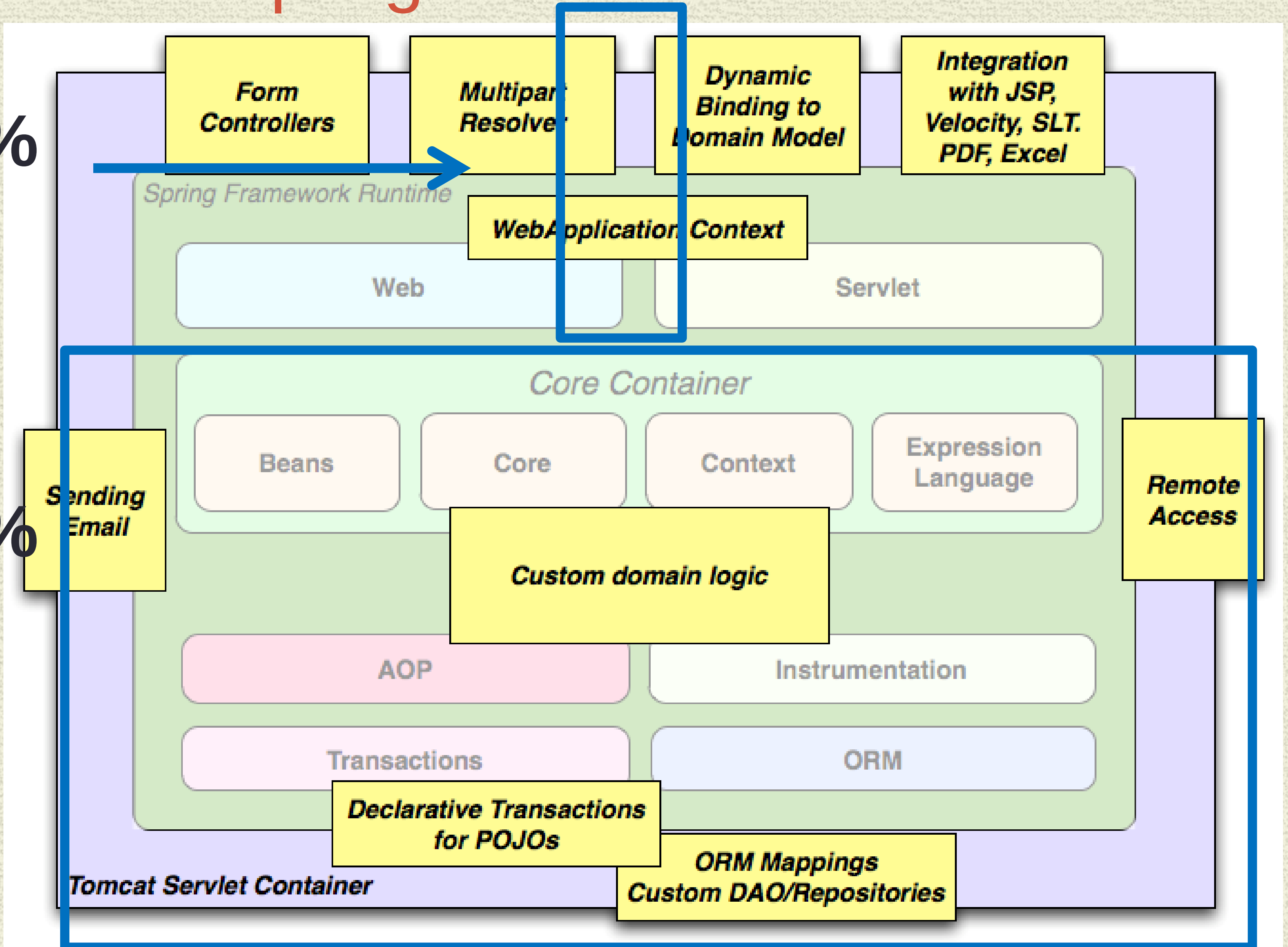
Enterprise := N-Tier Architecture

CS 544 EA Focus

Spring N-tier Architecture

15%

85%



Main Point

A software framework encapsulates the knowledge of experts, allowing the developers to take advantage of sound solutions and focus on the project qualities.

Science of Consciousness: *Through the practice of Transcendental Meditation, a person taps the value of Pure Consciousness which encapsulates knowledge of all the laws of nature..*