Software Engineering

Session 5 – Main Theme
High-Level Analysis and Design

Dr. Jean-Claude Franchitti

New York University
Computer Science Department
Courant Institute of Mathematical Sciences

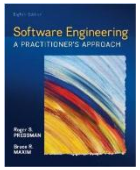
Agenda

1. Introduction
2. High-Level Analysis and Design
3. Architecture Blueprinting
4. Sample Architecture Blueprints
5. Architectural Mapping Process Illustrated
6. Reference Architecture Cataloguing Framework
7. Summary and Conclusion

What is the class about?

- Course description and syllabus:
  - http://www.nyu.edu/classes/jcf/g22.2440-001/
  - http://www.cs.nyu.edu/courses/spring15/G22.2440-001/

- Textbooks:
  - *Software Engineering: A Practitioner's Approach*



    Roger S. Pressman
    McGraw-Hill Higher International
    ISBN-10: 0078022126, ISBN-13: 978-0078022128, 8th Edition (01/23/14)
  - Recommended:
    - Code Complete: A Practical Handbook of Software Construction, 2nd Edition
    - The Mythical Man-Month: Essays on Software Engineering, 2nd Edition

High-Level Analysis and Design in Brief

- High-Level Analysis and Design Processes
- Architecture Blueprinting
- Sample Architecture Blueprints
- Architectural Mapping Processes
- Reference Architecture Cataloguing Framework
- Summary and Conclusion
  - Readings
  - Individual Assignment #1 (due)
  - Individual Assignment #2 (assigned)
  - Team Assignment #1 (ongoing)
  - Course Project (ongoing)

Icons / Metaphors

Information

Common Realization

Knowledge/Competency Pattern

Governance

Alignment

Solution Approach

Agenda

1. Introduction
2. High-Level Analysis and Design
3. Architecture Blueprinting
4. Sample Architecture Blueprints
5. Architectural Mapping Process Illustrated
6. Reference Architecture Cataloguing Framework
7. Summary and Conclusion

High-Level Analysis and Design

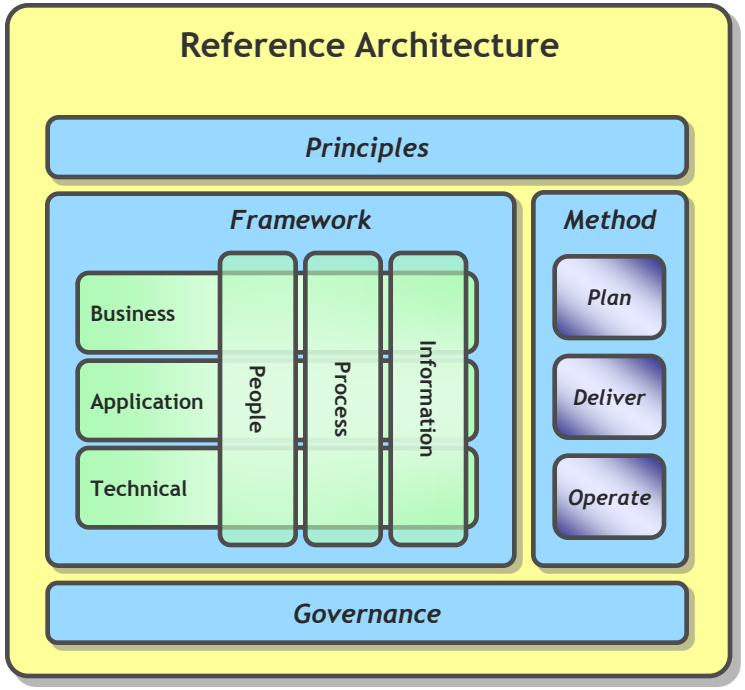
- The goal of high-level analysis and design is to quickly produce a **high-level model** that reflects the current understanding of the future state architecture
- This high-level model is helpful in putting together high-level program/project estimate and providing a **view** of the future state that can be used as a starting point
- Various architecture models may be used to represent this view and they are typically based on **blueprinting notations/process** and **blueprints** that have been standardized within the Enterprise working on the high-level analysis and design
- There are currently no industry standard blueprinting notation/process and/or blueprints; the blueprinting process typically goes top down to document the various facets of the future state architecture starting from the **Enterprise** level and going through the **business**, and **technology architectures**
- Technology architecture blueprinting can be conducted in parallel at the application, data, and technical levels

Architecture Models

- An Architecture provides the organizing logic for mapping business onto IT capabilities
- Creating models to describe an Architecture is a complex exercise as various levels of abstractions may need to be considered to effectively cover all requirements in increasing levels of detail
- Architecture Models are typically based on an integration of existing reference architecture styles
  - e.g., OMA and SOA, SOA and BPM, etc.

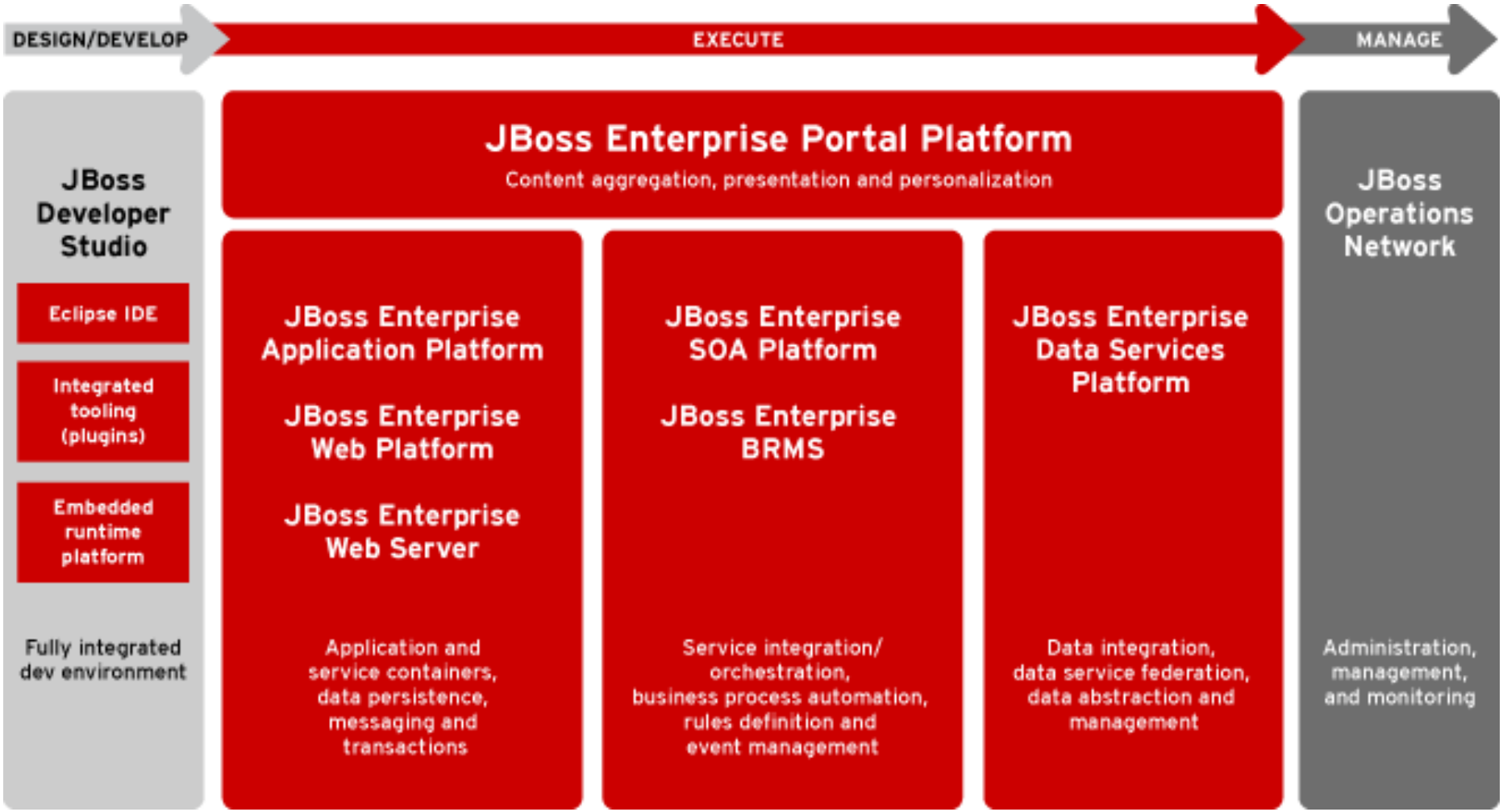
Reference Architecture

A Reference Architecture consists of foundational principles, an organizing framework, a comprehensive and consistent method, and a set of governing processes and structures.



- **Principles provide the foundation upon which the Reference Architecture is based. It includes a set of architectural terms as well as numerous principles, policies, and guidelines for governing the architecture.**
- **Framework is the organizing basis for the Reference Architecture and defines the architectural domains and disciplines that enable separation of concerns and IT to business alignment.**
- **Method is the comprehensive set of defined repeatable processes that are followed for a consistent and controlled realization of the Reference Architecture.**
- **Governance is the set processes and organizational structures that ensure conformity to the Reference Architecture.**

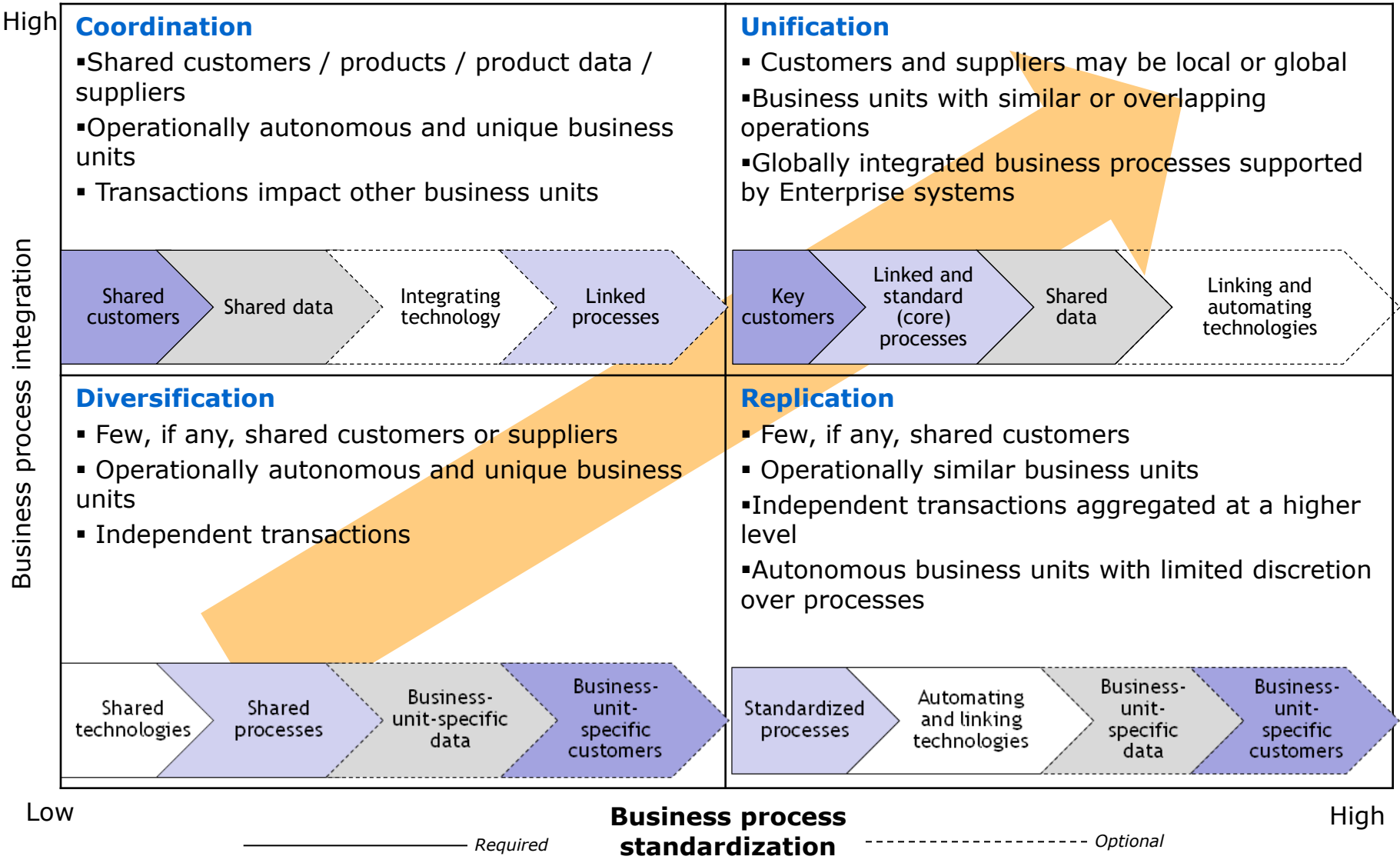
Sample Application Server Reference Architecture



Reference Architecture Models

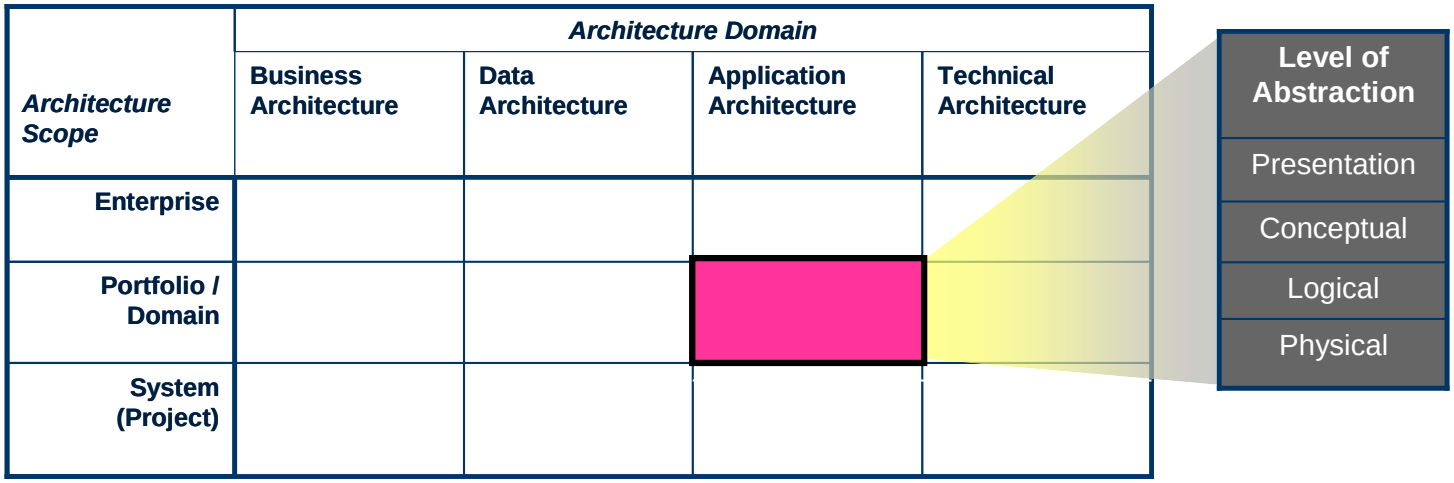
- A “reference” architecture model is an accepted representation of the architecture that drives the mapping of business capabilities onto IT capabilities
- A reference architecture model may not represent any specific organization needs
- A reference architecture model is rarely developed using a top-down or bottom-up approach, it is typically put together by integrating requirements from various architectural domains according to accepted heuristics (e.g., reuse via unification or best practices / standardization) and using accepted frameworks

Enterprise Architecture Modeling Heuristics



Typical Architecture Asset Catalog and Architecture Mapping Process

- Architects working at the Enterprise, Portfolio, and System levels use different models to represent their own views of a given architecture



- An Architecture Asset Catalog enables the representation of stakeholder views in matrix form to help catalog such models

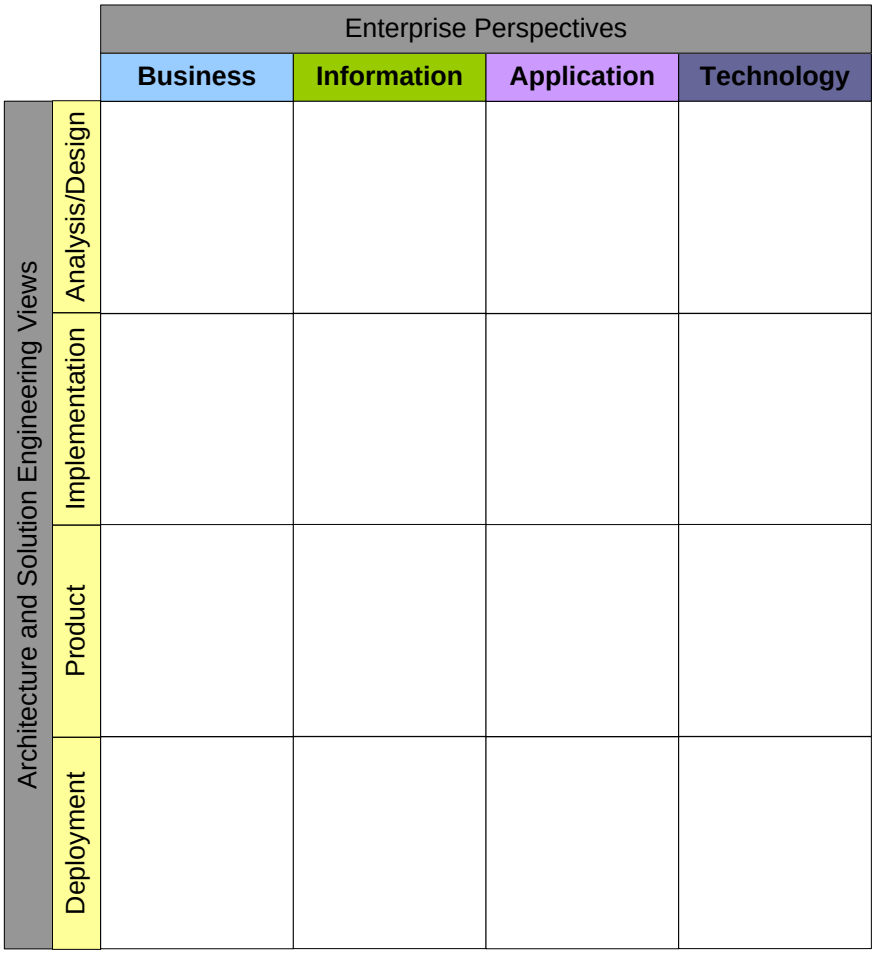
What is the Level of Abstraction?

- The level of abstraction refers to how far a blueprint is removed from “practical” considerations such as application servers, programming languages, DBMS technology, etc
- Although the levels of abstraction vary by architecture domain, four different types levels are recognized:

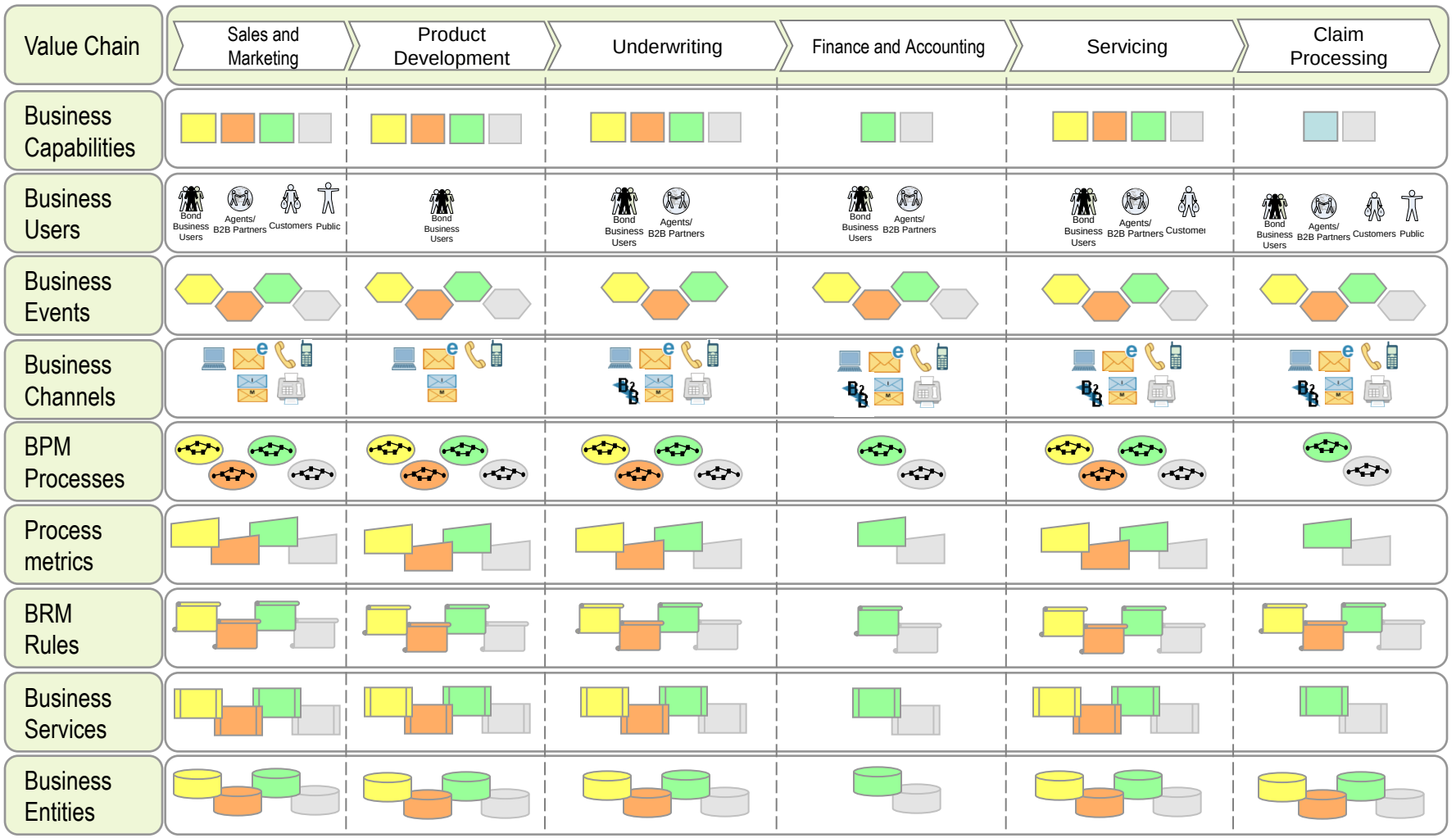
  - **Presentation** - a stylized model intended to greatly simply an architecture so that key messages can be effectively communicated, typically to business leadership
    - Presentation level diagrams are generally created by summarizing lower level architecture diagrams.
  - **Conceptual** - a highly generalized yet more formal depiction of an architecture that suppresses much of the actual detail -- either because the details are not important to the model and/or they had not been decided at the time the diagram was created
  - **Logical** - a detailed representation of an architecture that is generally independent of the underlying technology that is used (or will be) used to implement an architecture.
  - **Physical** -A representation that is typically dependent on the underlying technology that will be (or is) used to implement an architecture.

Sample Architecture and Solution Engineering Asset Catalog

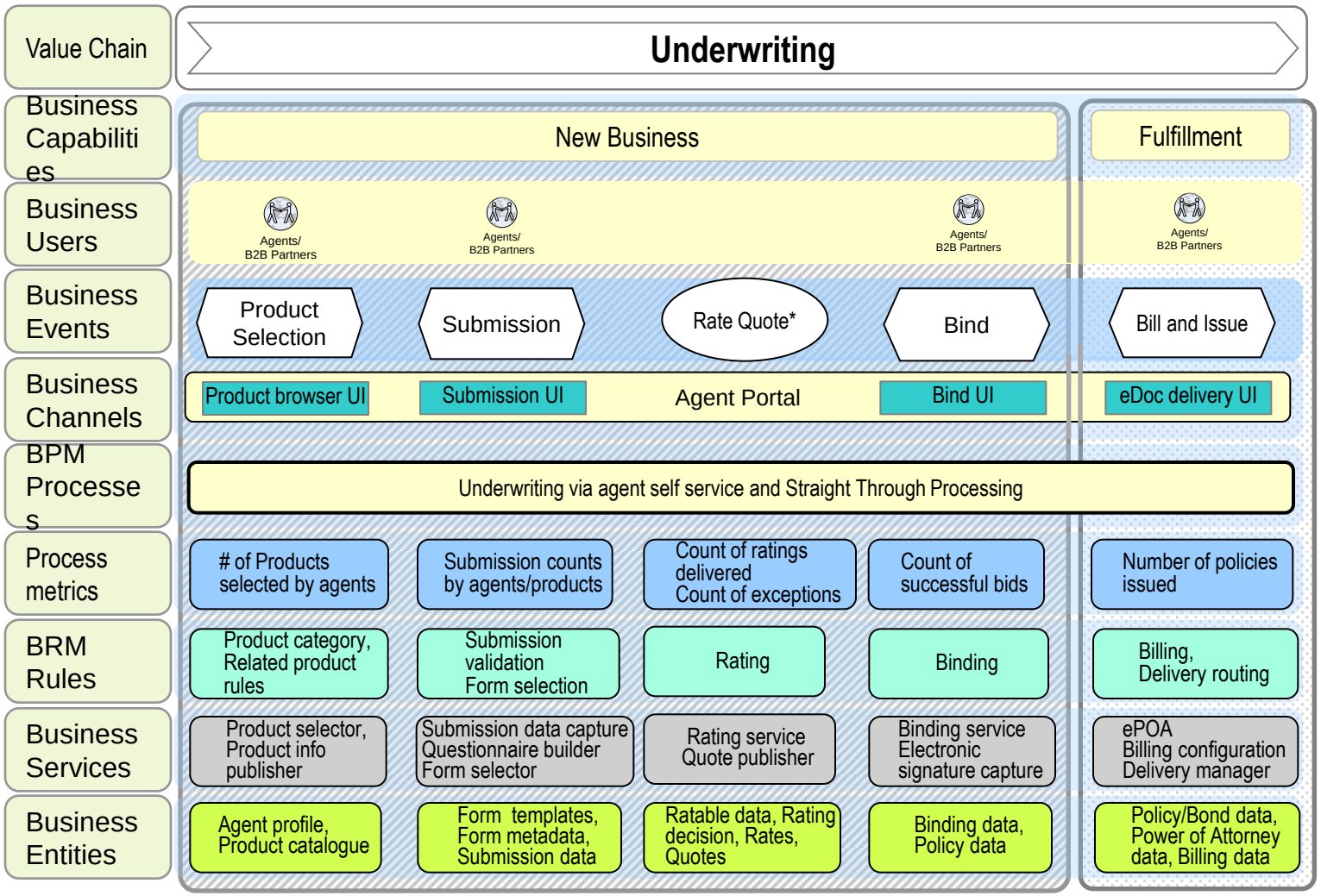
- In this context, relevant views are those that match the phases of an solution development life cycle



Conceptual business architecture framework



Conceptual Business Architecture Framework Illustration



Agenda

1. Introduction
2. High-Level Analysis and Design
3. Architecture Blueprinting
4. Sample Architecture Blueprints
5. Architectural Mapping Process Illustrated
6. Reference Architecture Cataloguing Framework
7. Summary and Conclusion

What is Blueprinting?

- Blueprinting is fundamentally concerned with the high-level representation of intangible assets (e.g., applications, databases, interfaces, networks, servers, etc.) so that:
  - The interrelationship between the various assets can be understood
  - The assets may be changed more reliably
  - Architectural level design decisions become observable

What is a Blueprint?

- A blueprint is an architectural drawing
  - Created using a consistent representation to represent a high-level model of the as-it, to-be, or in-transition IT environment
- Unlike UML models, which are software engineering level diagrams, blueprints are at an architectural-level of detail and provide the context needed to visualize the “big picture”
  - As such, blueprints are analogous to the “city-planning” level in the building construction industry
  - They enable architects to communicate the overall design of the city as opposed to the design of the individual buildings that make up the city

What Does a Blueprint Look Like and How is it Used?

- The appearance of a blueprint varies considerably depending upon a number of factors including:
  - The architectural domain being modeled (e.g., application architecture versus technical architecture)
  - The scope of the blueprint (e.g., Enterprise, Portfolio, Project)
  - The level of abstraction (e.g., Presentation, Conceptual, Logical, Physical)
  - The communication objectives of the model
- Blueprints are also used to document and define three different states of technology evolution
  - A current state called the **As-Is** or POD
  - A future state called the **To-Be** or POA (typically 12 - 24 months out)
  - One or more transition states, each one called a **Transition** or planned landing point between the as-is and to-be state
    - Once implemented, a Transition represents a new As-Is state

Why is there a Need for a Common Way to Document Architectures?

- In the absence of standardized blueprinting techniques, architectural models would be highly individualized and would range from artifacts that may be fairly structured to models that would be very general and stylistic
- As a result, the readers interpreting the models would be required to ask (and assume an answer to) a number of critical questions including:
  - *What concepts is the model attempting to explain?*
  - *Are the concepts highly abstract or is the model depicting a precise design?*
  - *What do the symbols on the diagram represent?*
  - *What architecture domain is being modeled?*
  - *Does the design apply to the Enterprise as a whole, a LOB, a portfolio, or a project?*
  - *Does the model represent the As-Is , To-Be, or Transition architecture?*
  - *If the model represents a Transition architecture, what changes to the IT environment are being planned?*
  - *etc.*

Blueprinting and UML at a Glance

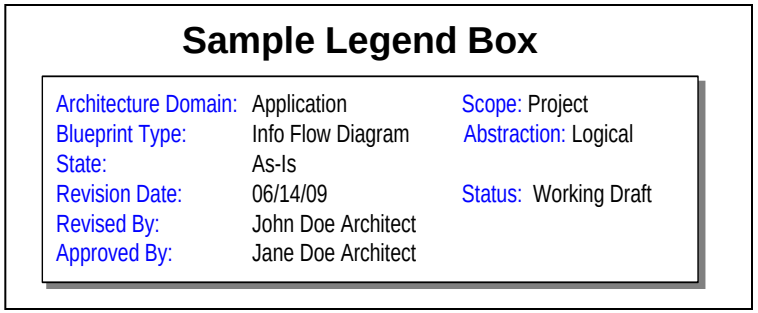
- Blueprinting and UML are intended to be used together on the same project
- Blueprint artifacts are used to document the end-to-end high-level designs for projects
  - *Blueprints are analogous to the "city-planning" level in the building construction industry*
  - *They enable architects to communicate the overall design of a city (project) as opposed to the design of the individual buildings (applications) that make up the city*
- UML artifacts are used for software engineering tasks (e.g., architecting the buildings)

	UML	Blueprinting
Focus	Software development	Application integration
Use	Analyze and design software systems and modules, typically using an OO approach	Describe or prescribe an end-to-end design without delving into details
Level of detail	High to low-level	High-level
Central Element of Granularity	A system and its subsystems	System of systems
Learning Curve	Significant	Minimal

Lack of Blueprinting Standards

- *According to Research Analysts and reports…*
  - Modeling at the Enterprise and Portfolio levels tends to be fairly generalized
    - The goal at these levels is to communicate the “big picture“ (as opposed to “application-level” designs)
  - UML is an OO modeling system with schematics and notations for application development (e.g., "building-level" designs)
    - It is not well suited for modeling portfolio and Enterprise level architectures (e.g., the "city-level" or “big picture” designs)
  - There may never be industry standards at the Enterprise and Portfolio levels

Legend Box



- The Legend Box is a text box that must appear on all blueprints. It is used to denote important information that is needed by the reader to correctly interpret a blueprint. The following information is included in the Legend Box:
- **Architecture Domain** - Used to specify what aspect of the environment is the subject of architecture artifact - One of the following domains must be specified:
  - *Business Architecture* —specify this when the model depicts the company's business capabilities, business processes, organizational structure, major locations, or relationships with partners and customers
  - *Application Architecture* — specify this when the model depicts the application assets that support business capabilities and processes
  - *Data Architecture* —specify this when the model depicts the company's business rules, business data and/or information types, along with their interrelationships
  - *Technical Architecture* —specify this when the model depicts hardware and facilities, system software, data storage resources, networks, and other underlying technologies
    - Technical architecture provide the platform that supports the activities and interfaces of the other domains

Legend Box (continued)

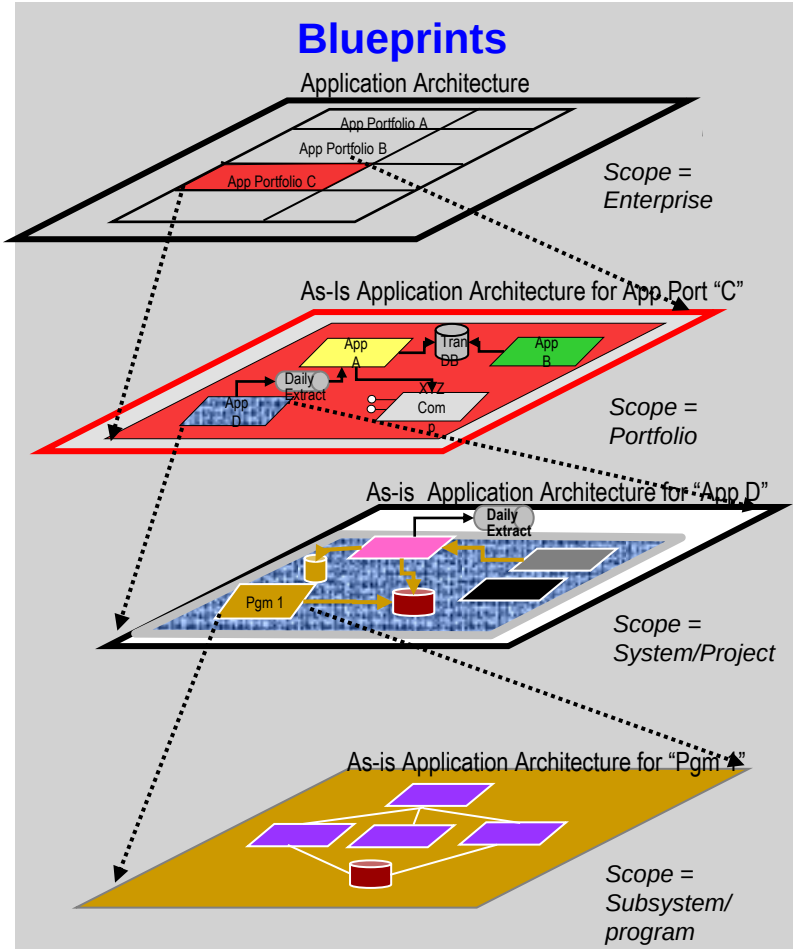
- **Scope** - Defines the breath (or scope of authority) for a blueprint. Several different scopes are recognized:
  - *Enterprise* - A model that generally depicts a company's environment as a whole
  - *Portfolio* - A model that depicts the architecture of a portfolio (e.g., Field Management)
  - *Program/Project* – A model that depicts the architecture of a program or project
  - *Asset* - A diagram that depicts the architecture of an asset

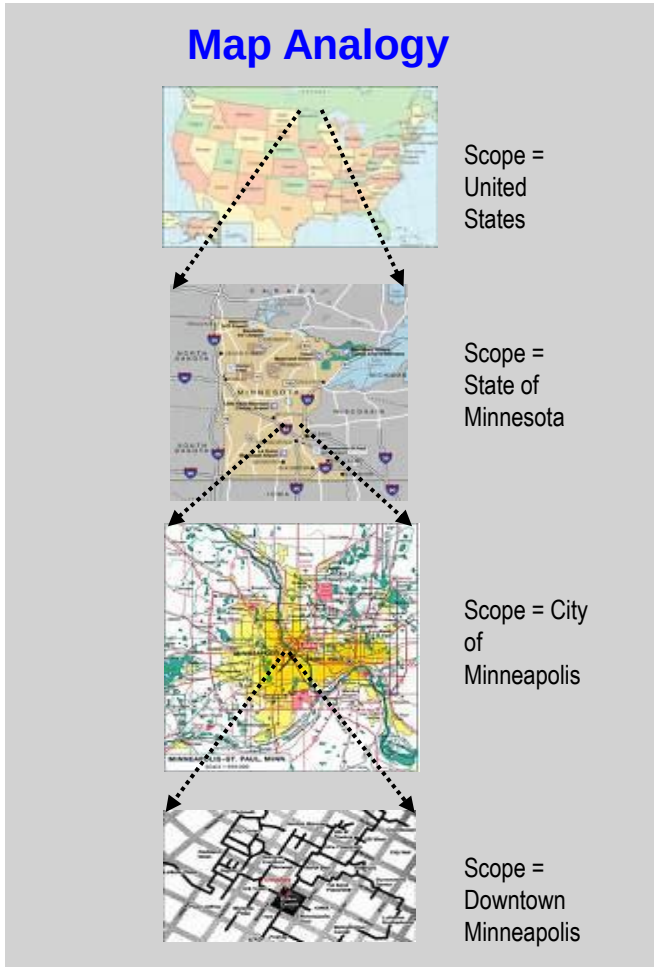
- **Abstraction** - Refers to how far the model is removed from "practical" considerations such as application servers, programming languages, etc
  - Four different levels are recognized: Presentation, Conceptual, Logical and Physical

- **State** – Used to answer the question: *Does this model represent the current state or some proposed future state?* Three different states are typically recognized:
  - As-is - the current state.
  - To-be - the desired future state that is to be achieved in a specified time period (typically 12 – 24 months). In reality, the to-be state is a moving target that generally represents an aspiration, as opposed to a fixed target that will be achieved
  - Transition - a planned landing point between the current state and the to-be state
    - A Transition diagram shows progress towards the future state
    - Once implemented, a transition architecture represents the new As-Is and the previous current As-Is becomes the As-Was

Importance of the Scope of a Blueprint

- Specifying the scope of a diagram is critical because there is a direct correlation between the scope and the amount of detail that can be depicted on a blueprint. The reason is that the amount of generalization (e.g., simplification, feature selection, grouping, etc.) must increase with the scope of the blueprint. The following examples illustrate this key point:

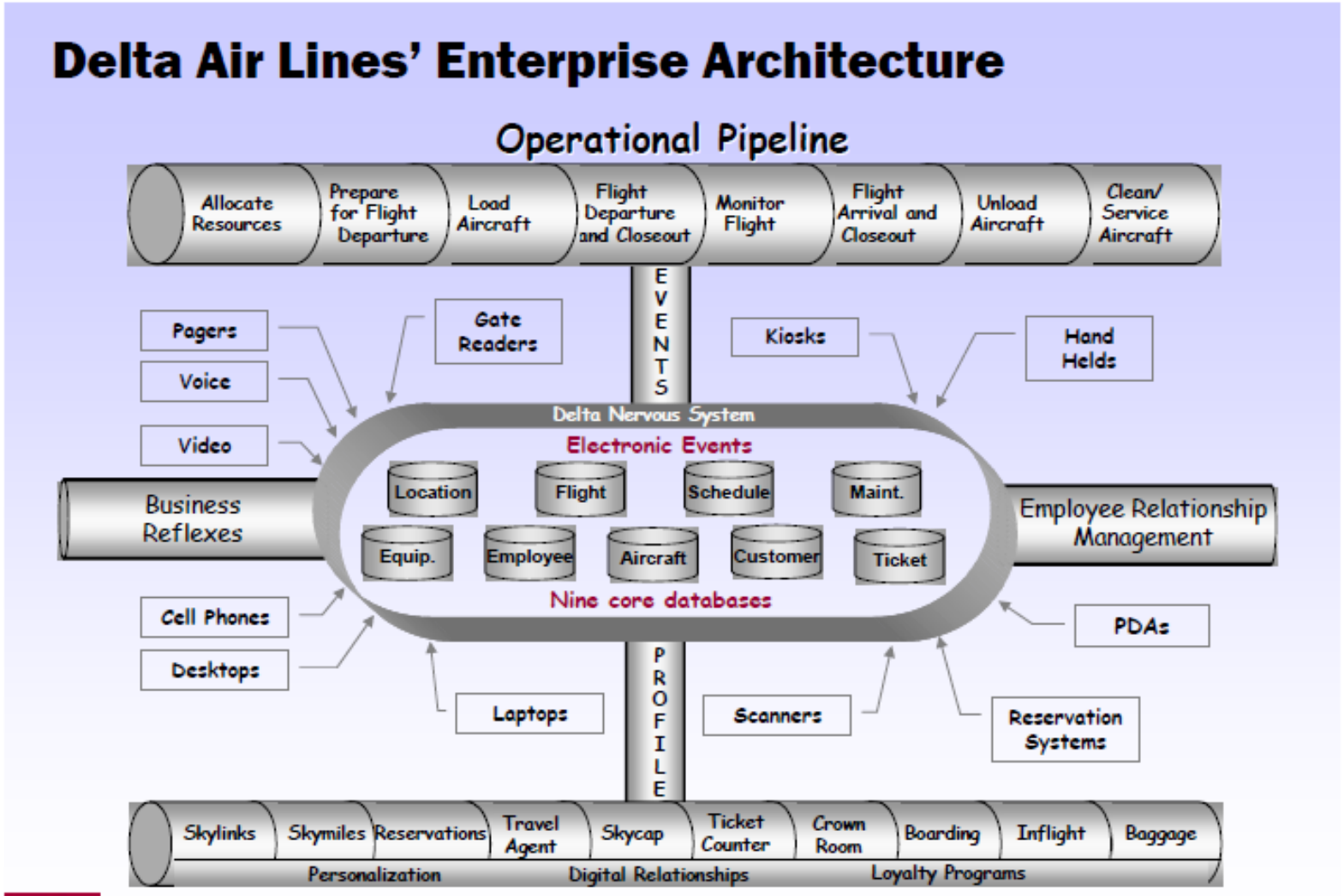




Agenda

1. Introduction
2. High-Level Analysis and Design
3. Architecture Blueprinting
4. Sample Architecture Blueprints
5. Architectural Mapping Process Illustrated
6. Reference Architecture Cataloguing Framework
7. Summary and Conclusion

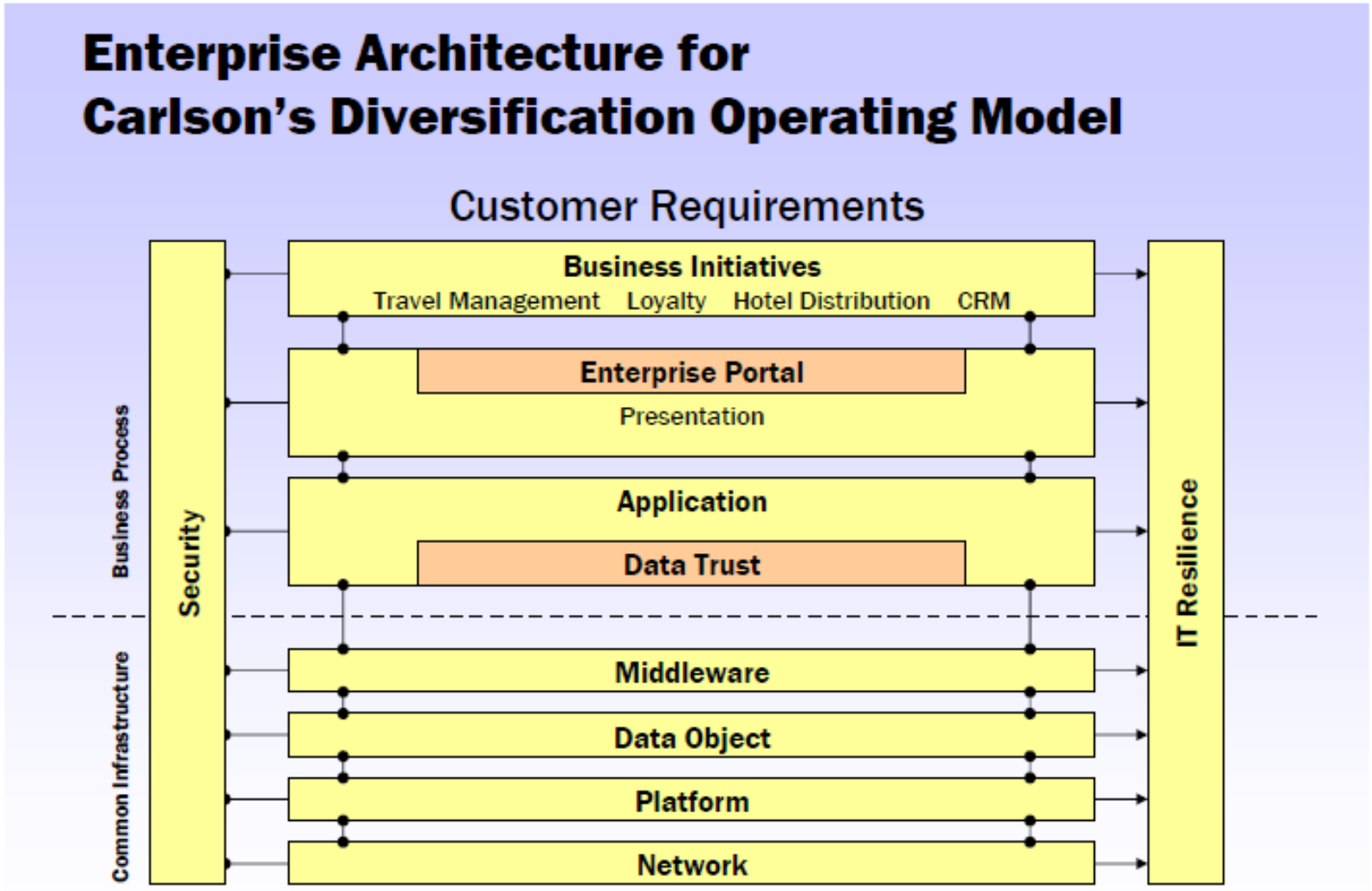
Sample Enterprise Architecture Blueprint for Unification Operating Model



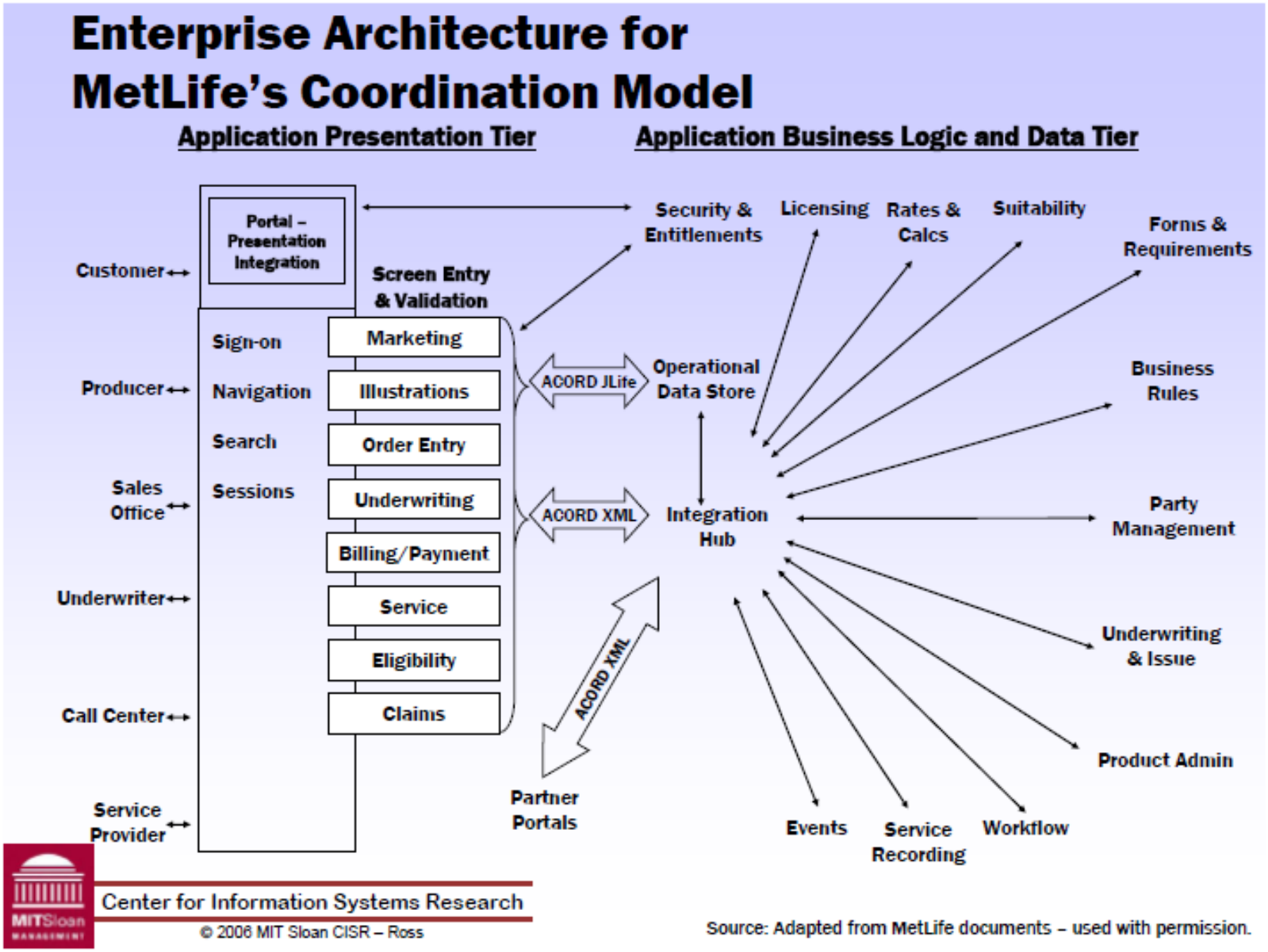
Center for Information Systems Research

Source: Adapted from Delta Air Lines documents – used with permission.

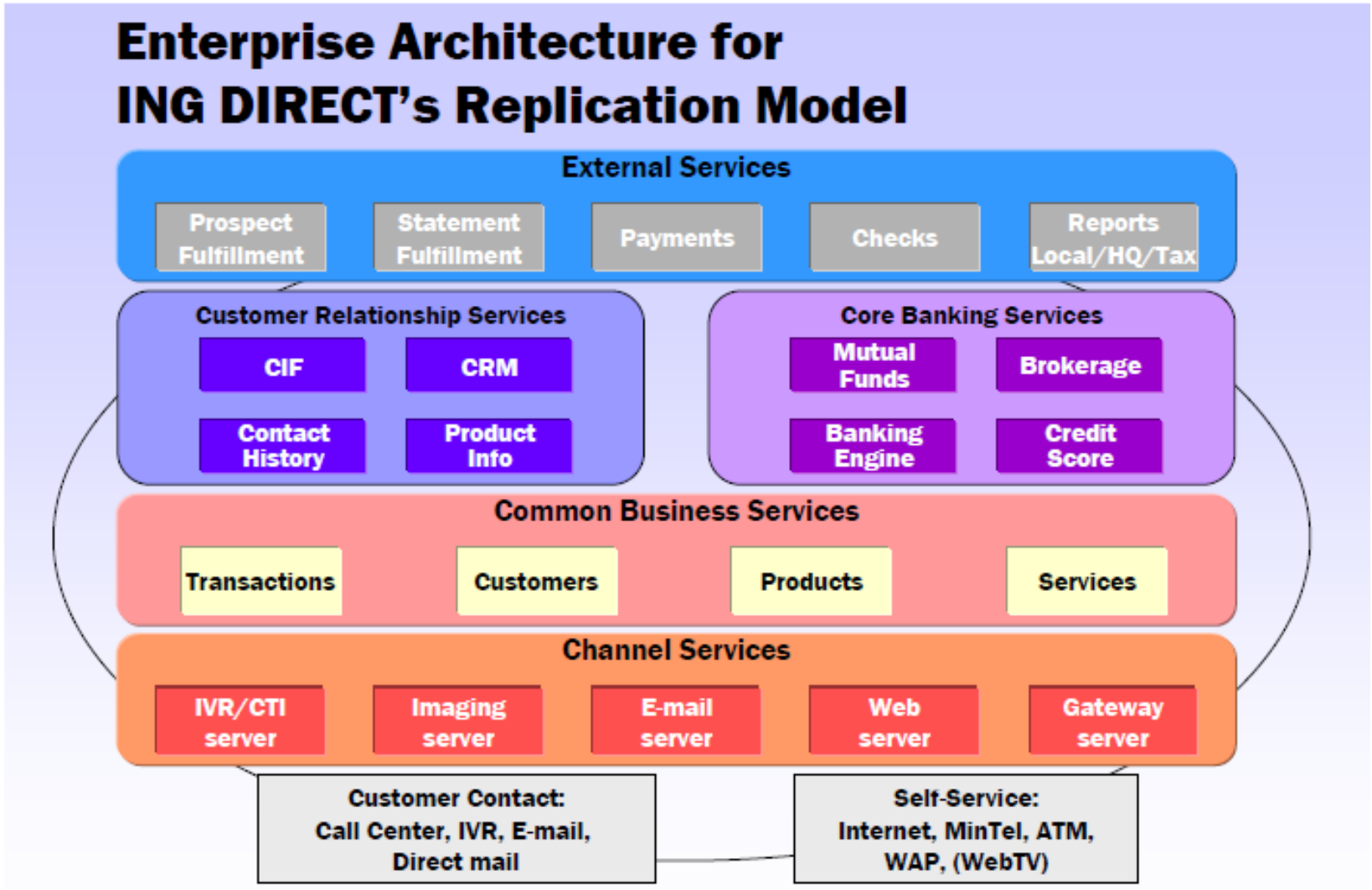
Sample Enterprise Architecture Blueprint for Diversification Oper. Model



Sample Enterprise Architecture Blueprint for Coordination Oper. Model



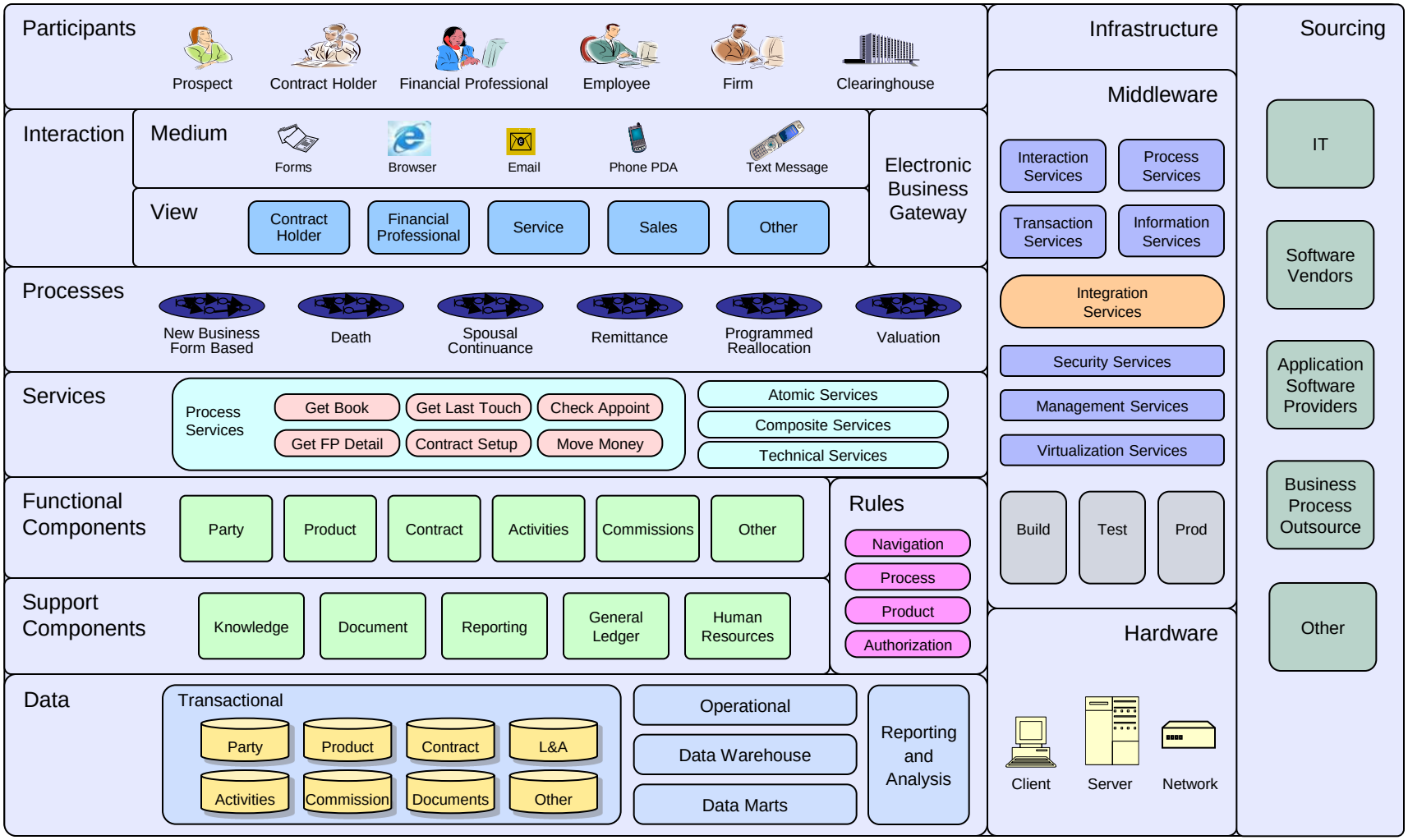
Sample Enterprise Architecture Blueprint for Replication Operating Model



Center for Information Systems Research

Source: Robertson, D. "ING DIRECT: The IT Challenge (B)", 2003, IMD-3-1345. Used with permission.

Sample Reference Enterprise Architecture Blueprint



Sample Detailed Level Business Architecture Blueprint

Client and Channel Mgmt

- Market Research
- Market Segmentation
- Channel Management
- Contractholder Relationship Mgmt
- Firm Relationship Mgmt
- FP Relationship Mgmt
- Client Profile
- Asset Retention

Product

- Product Portfolio Planning
- Product Conceptualization
- Product Definition and Design
- Product Deployment
- Product Performance and Pricing
- Regulatory Filing
- Investment Management

Sales

- Promotion and Brand Mgmt
- Sales Generation and Enablement
- Licensing and Appointments
- Campaign Management
- Sales Compensation
- Commissions
- Suitability

Service

- Contract Setup
- Money-In
- Contact Non-Financial Servicing
- Contract Financial Servicing
- Reallocations
- Annuitization
- Payments
- Valuation
- Correspondence Management
- Value Added Services

Risk and Financial Mgmt

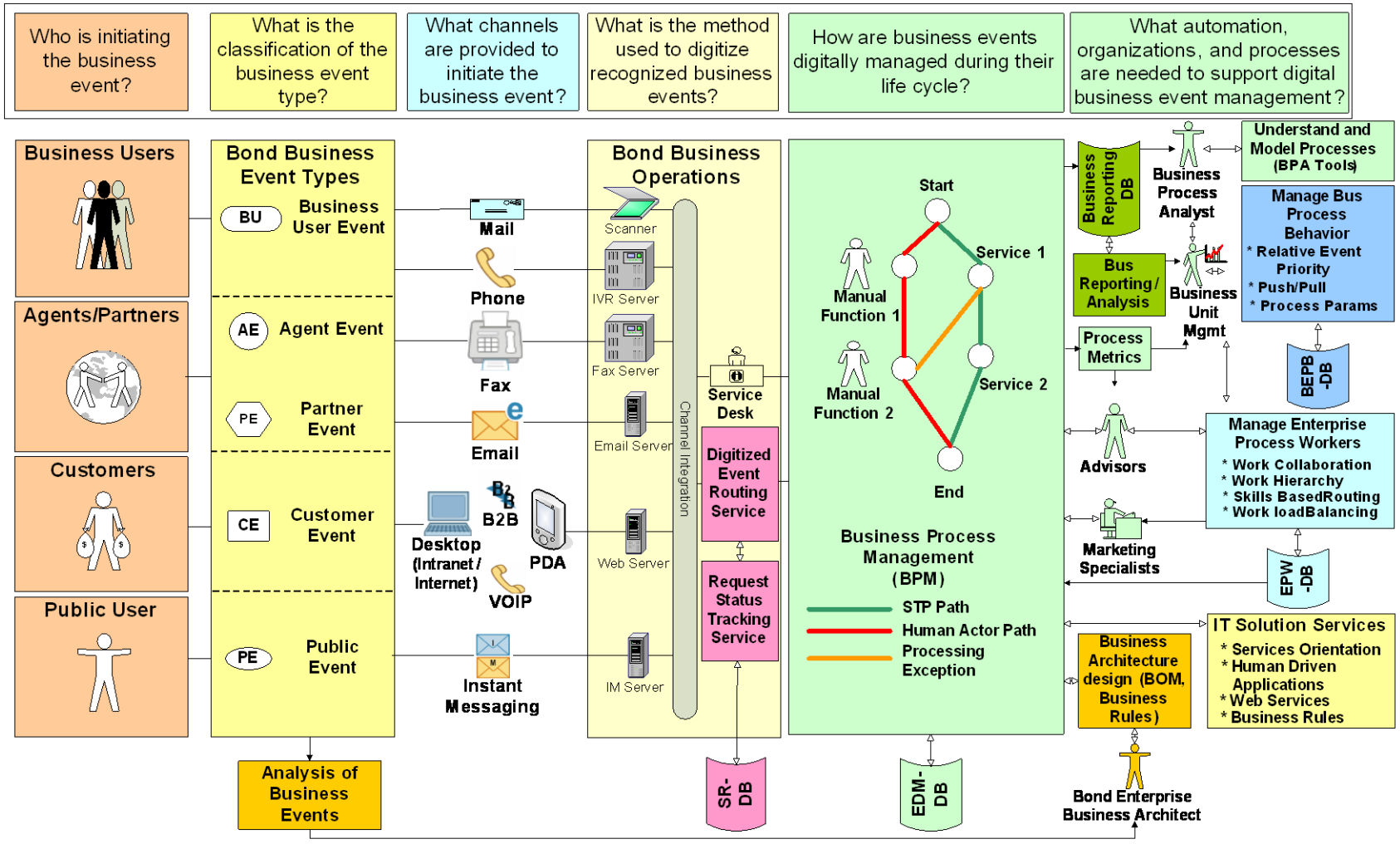
- Risk Management
- Regulatory and Compliance
- Legal
- Financial Risk Management
- Financial Management
- Financial Reporting and Metrics
- Mergers and Acquisitions
- Business Continuity

Business Management and Support

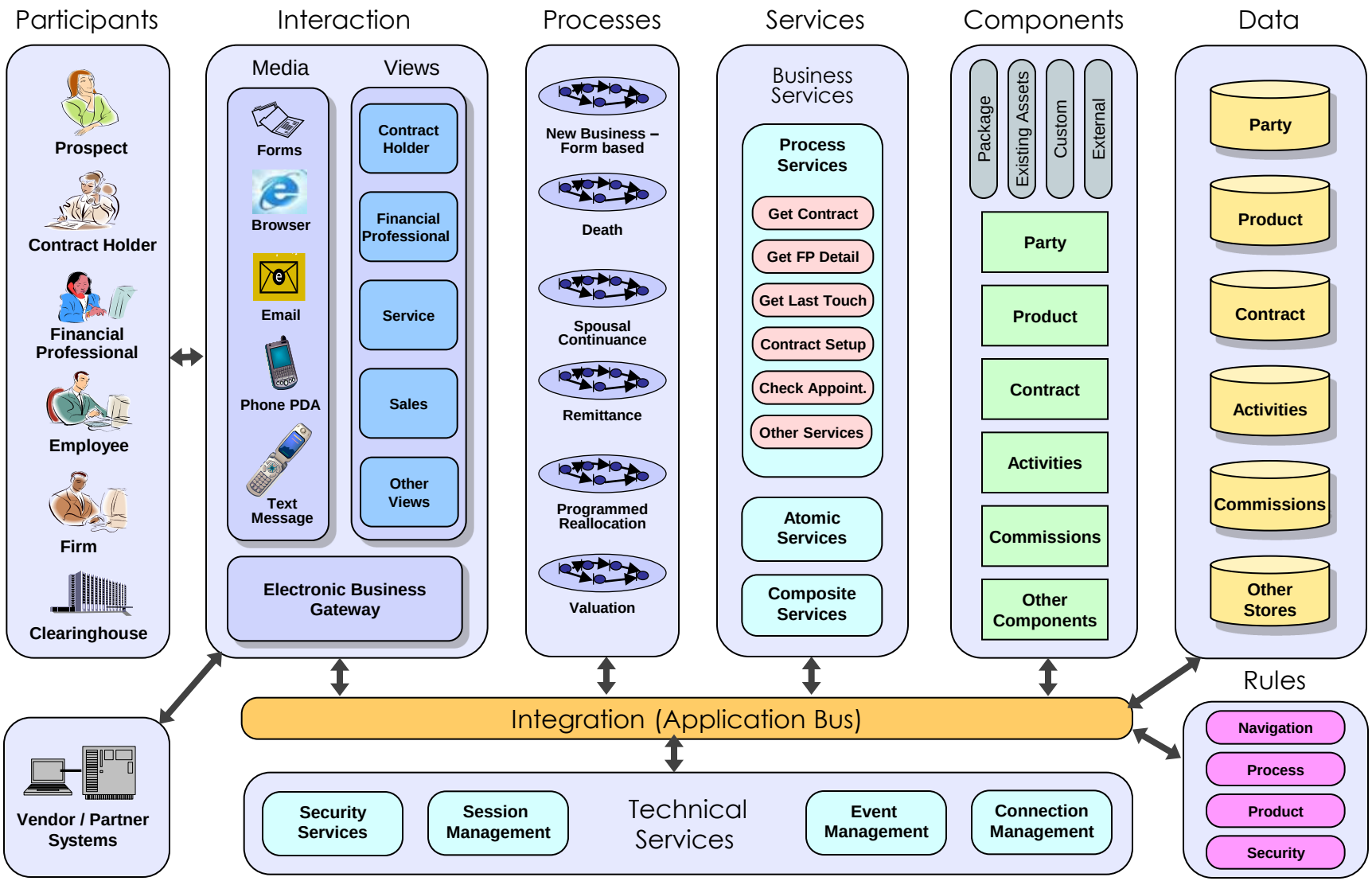
- Communications and Public Relations
- Human Resources
- Accounting
- Information Technology
- Procurement and Vendor Mgmt
- Document and Content Mgmt
- Security and Privacy
- Training and Knowledge Mgmt
- Auditing
- Analytics and Reporting
- Facilities Management

Sample High-Level Business Architecture Blueprint

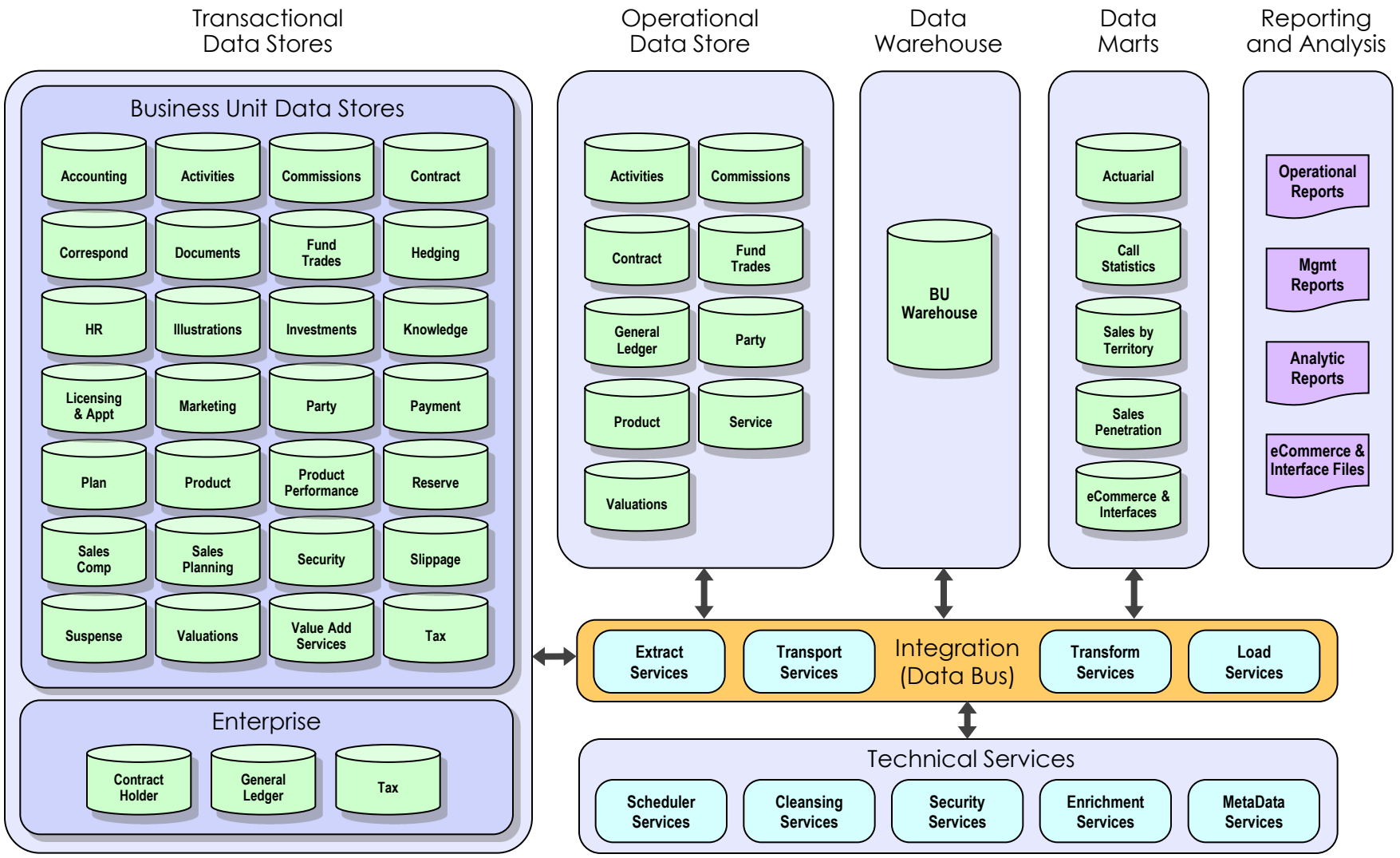
Conceptual Technology Architecture Blueprint



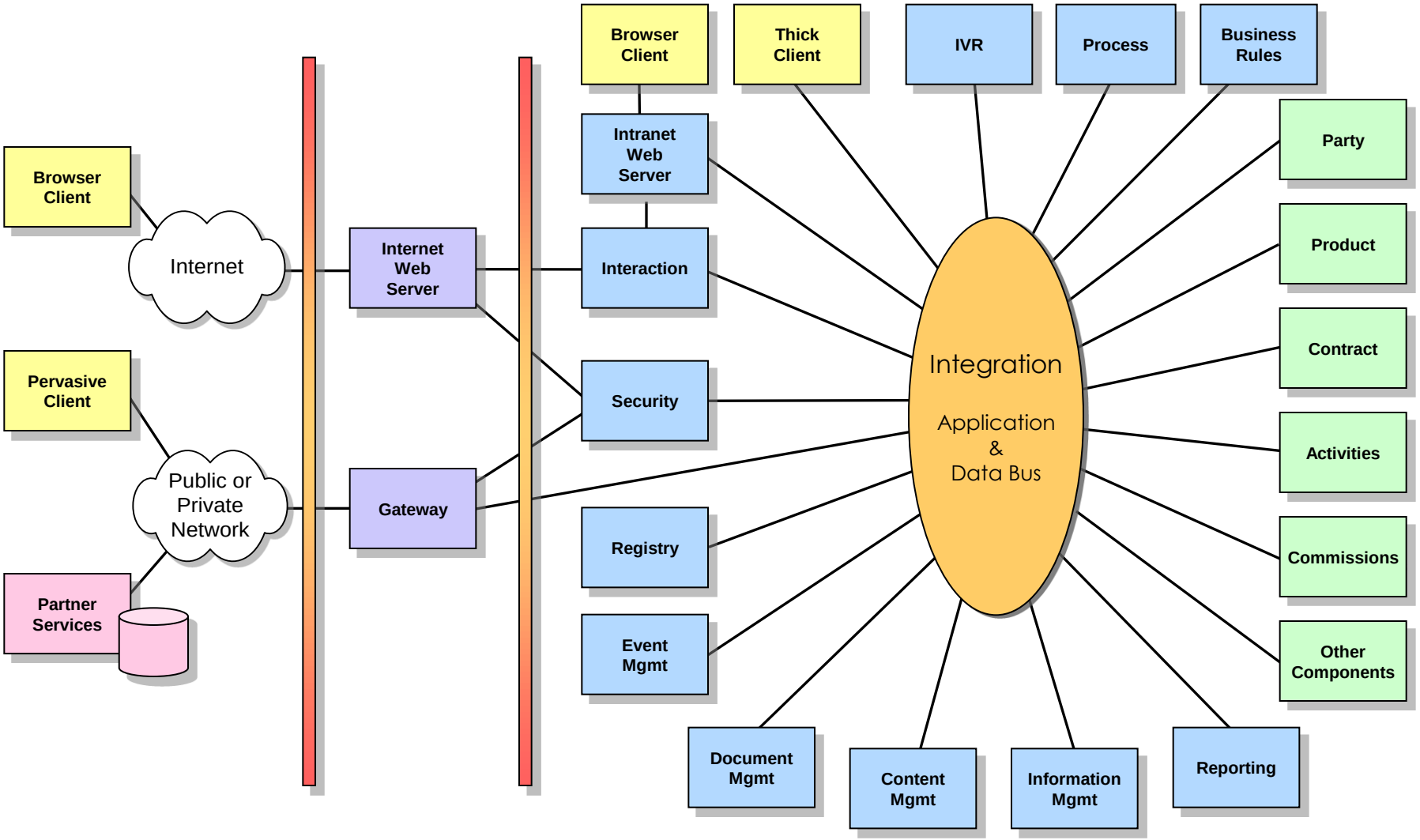
Sample Application Architecture Blueprint



Sample Information Systems Architecture Blueprint



Sample Infrastructure Architecture Blueprint



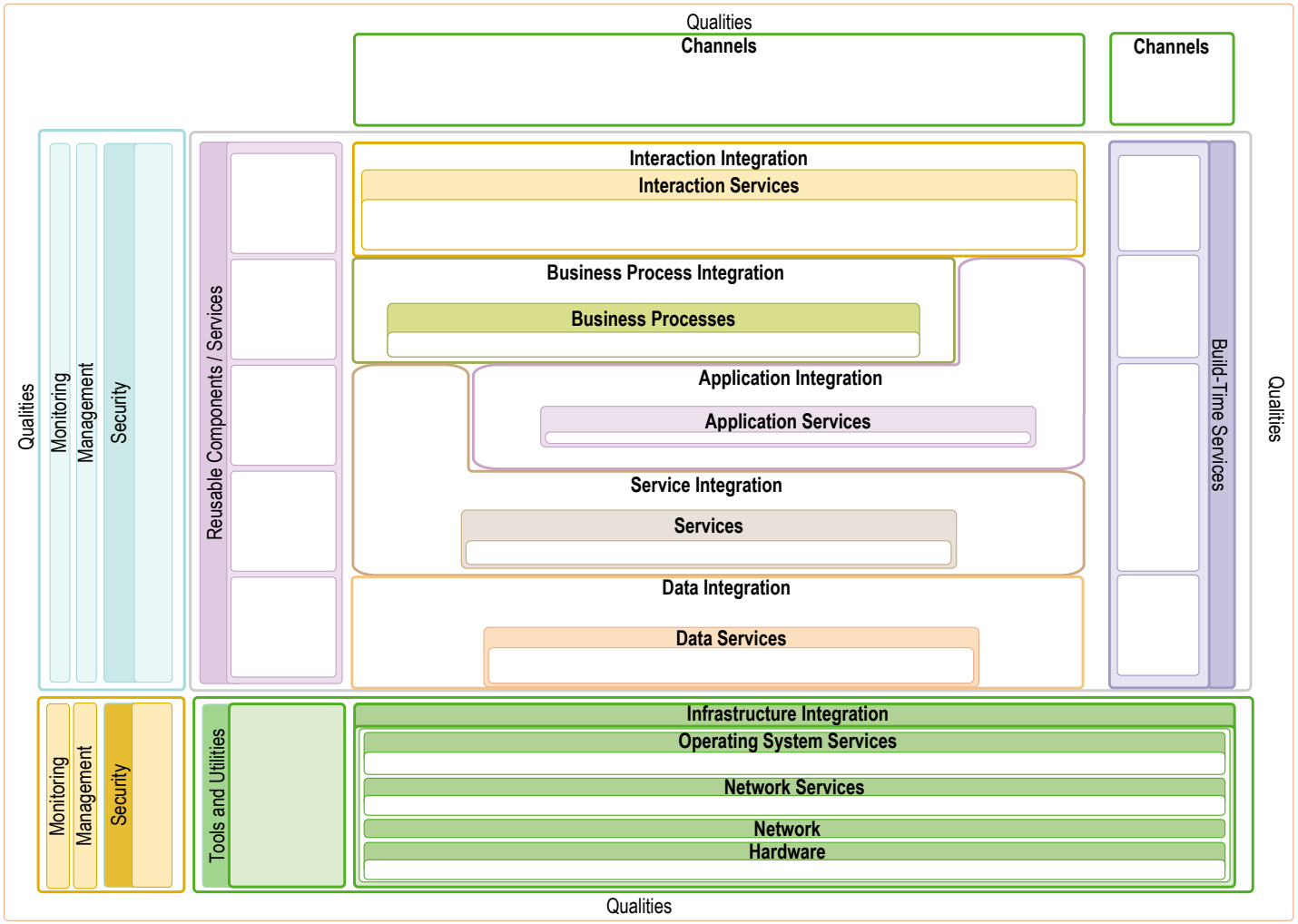
Agenda

1. Introduction
2. High-Level Analysis and Design
3. Architecture Blueprinting
4. Sample Architecture Blueprints
5. Architectural Mapping Process Illustrated
6. Reference Architecture Cataloguing Framework
7. Summary and Conclusion

Mapping Dimensions to Consider

- Levels of abstraction
- Breadth (i.e., architectural domain)
- Depth (i.e., services/facilities needed)
- Specialization (i.e., styles and related pattern)
- Integration of various patterns results in integration variants/hybrids
- Mapping relies on the selection of standards and products that implement that standard
  - e.g., JEE – IBM WebSphere Application Server

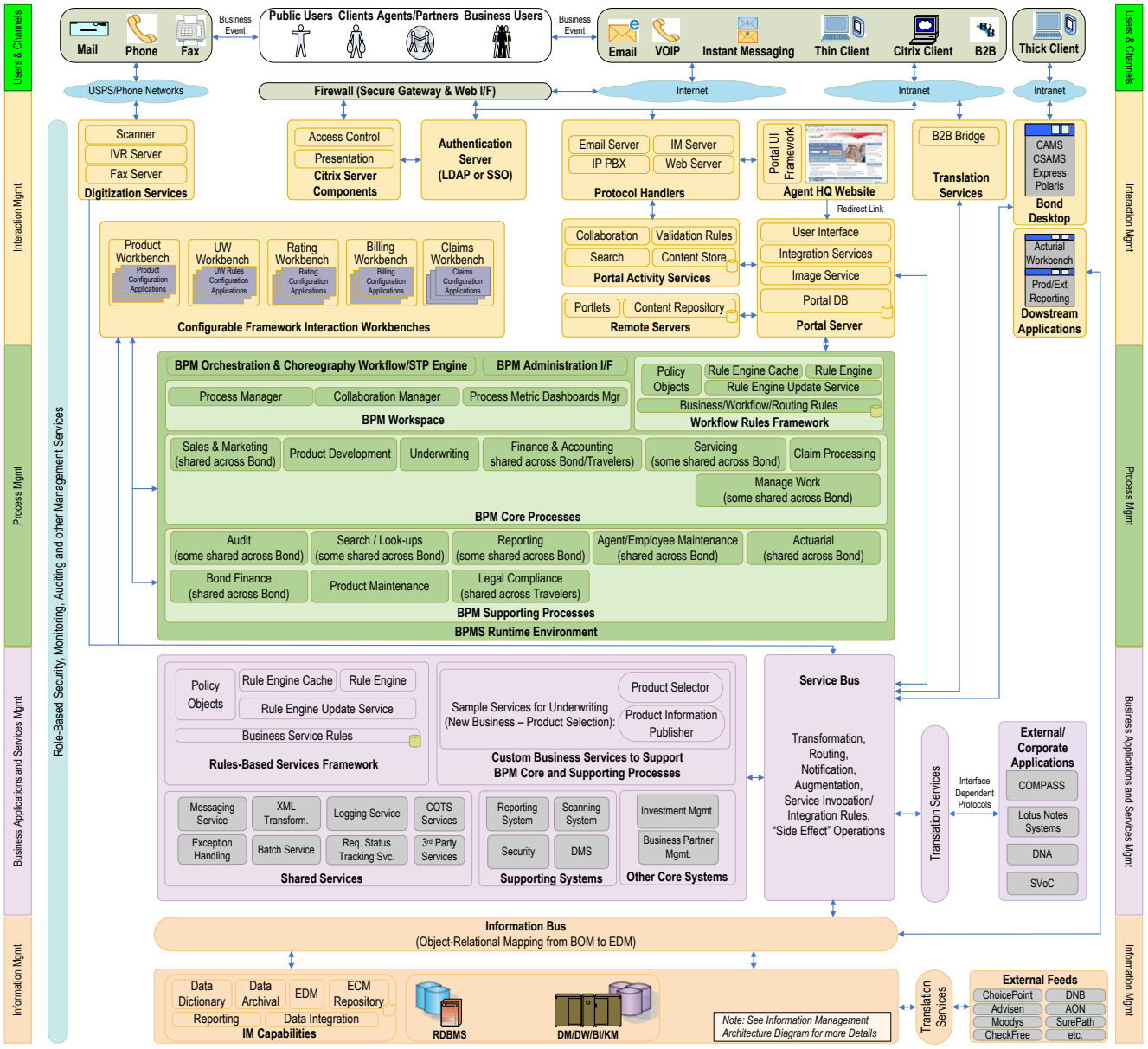
Sample Reference Logical Application Architecture Blueprint (OMA / SOA Hybrid)



Separation of concerns through layering enables high cohesion and low coupling across the application components

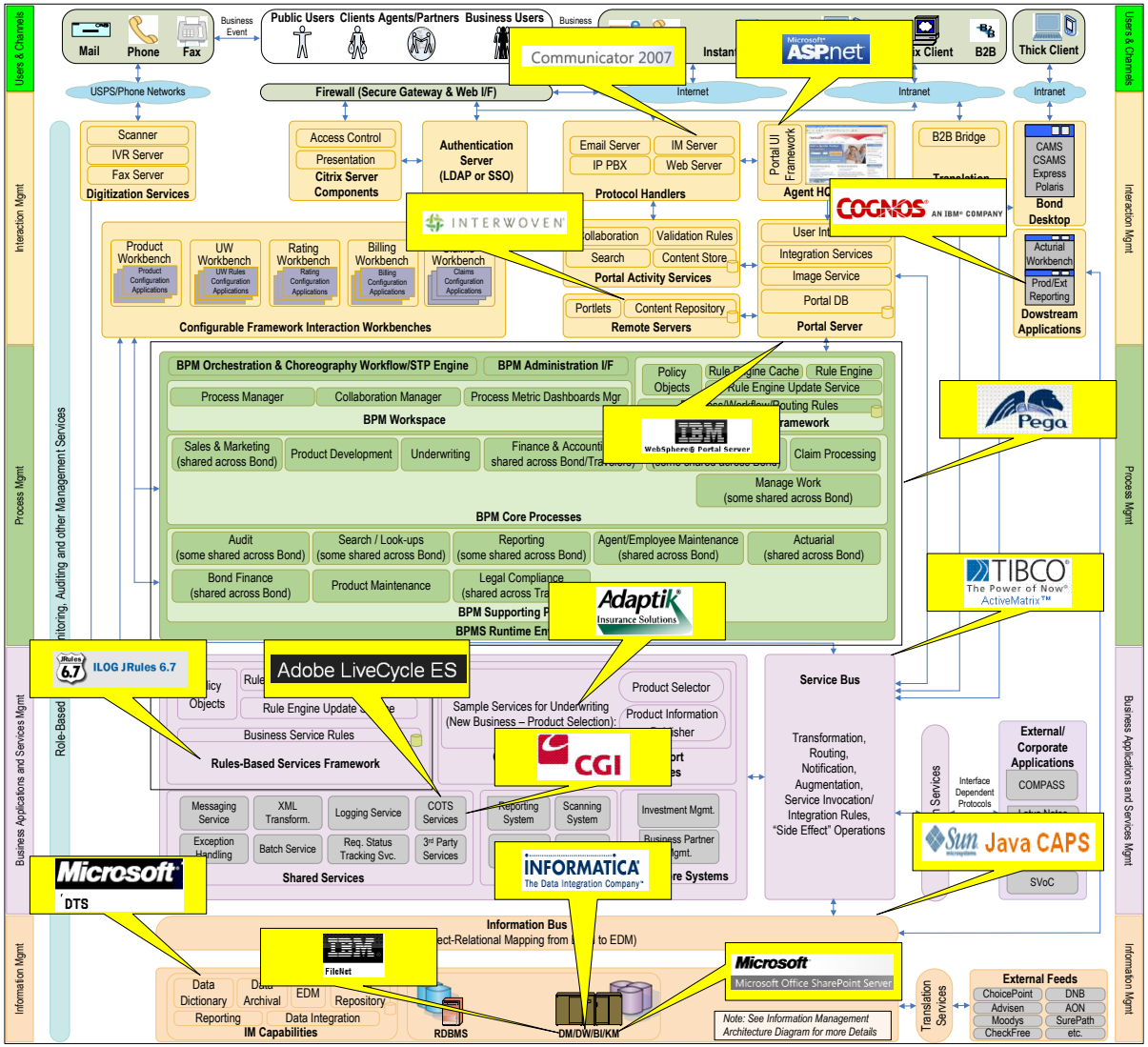
Interaction Integration
Bus. Process Integration
Application Integration
Service Integration
Data Integration
Infrastructure Integration

Sample Reference Application Architecture Implementation Blueprint (OMA/SOA Hybrid)



Technology Products Mapped to the Reference App. Arch. Impl. Blueprint (OMA/SOA Hybrid)

- Sample Product Mapping:
  - JEE Standard
  - IBM WebSphere Product Family
  - Various Third Party Products (as indicated)



Agenda

1. Introduction
2. High-Level Analysis and Design
3. Architecture Blueprinting
4. Sample Architecture Blueprints
5. Architectural Mapping Process Illustrated
6. Reference Architecture Cataloguing Framework
7. Summary and Conclusion

Challenges of an Architecture Continuum Catalog

Scope and Detail of Architectures

- Any Enterprise is likely to have many Solutions and Architectures
- Different Segments of the Enterprise, Product lines or Divisions
- Different Levels of Detail suitable for different purpose and audience
- Time horizon of an Architecture

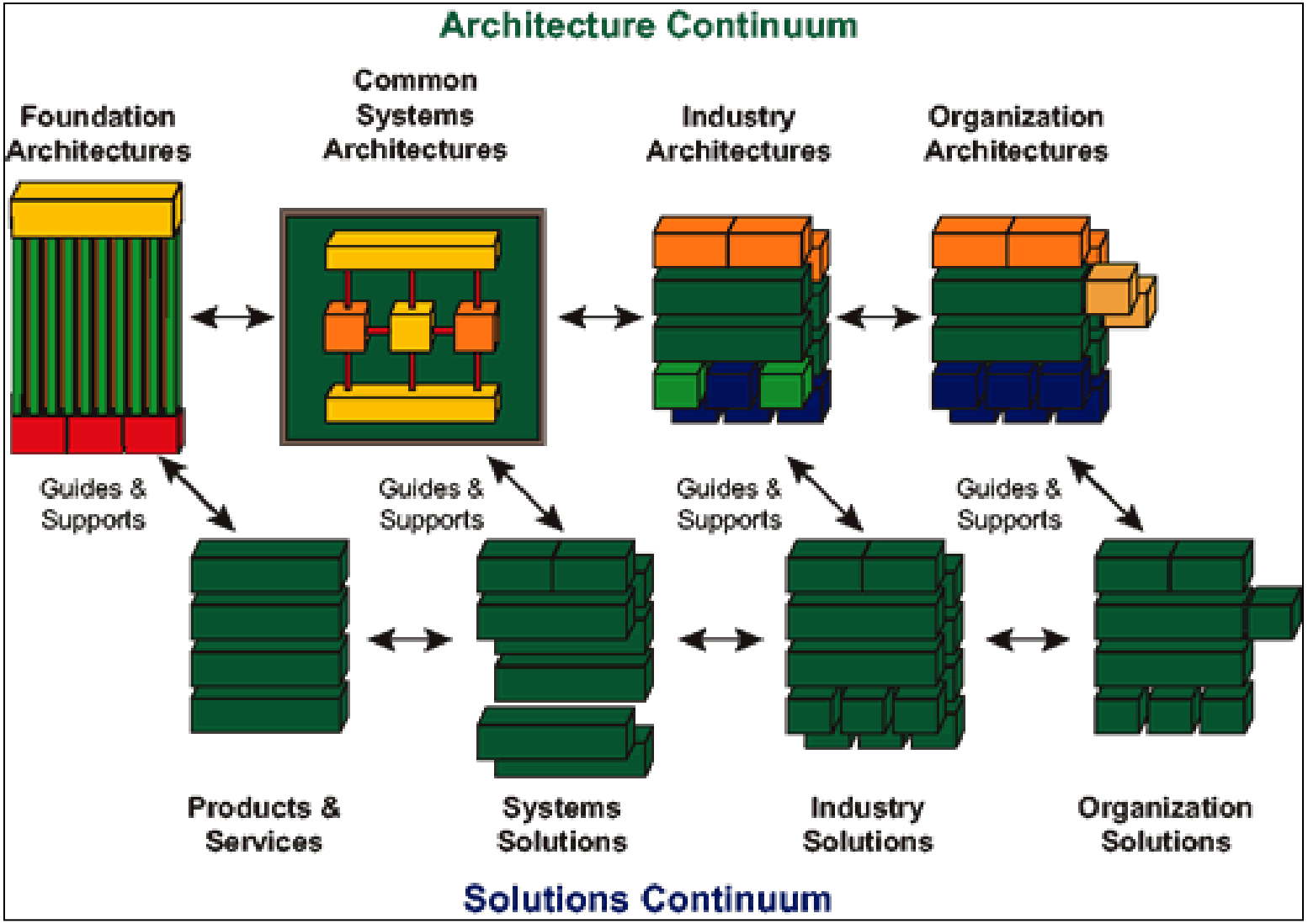
Specialization Hierarchy of Architectures

- Generic Architectures common to all industries
- Industry Specific Architectures
- Reference Architecture of a particular Organization
- …

Domains and Views of an Architecture

- Business Domain, Information Domain, Application Domain, Infrastructure Domain
- A Master Architecture can cover all these domains at a high level
- Each Domain can have a single comprehensive view of an Architecture
- Each Domain can have multiple views to cover an Architecture
- Specialized views for specific purposes within each domain

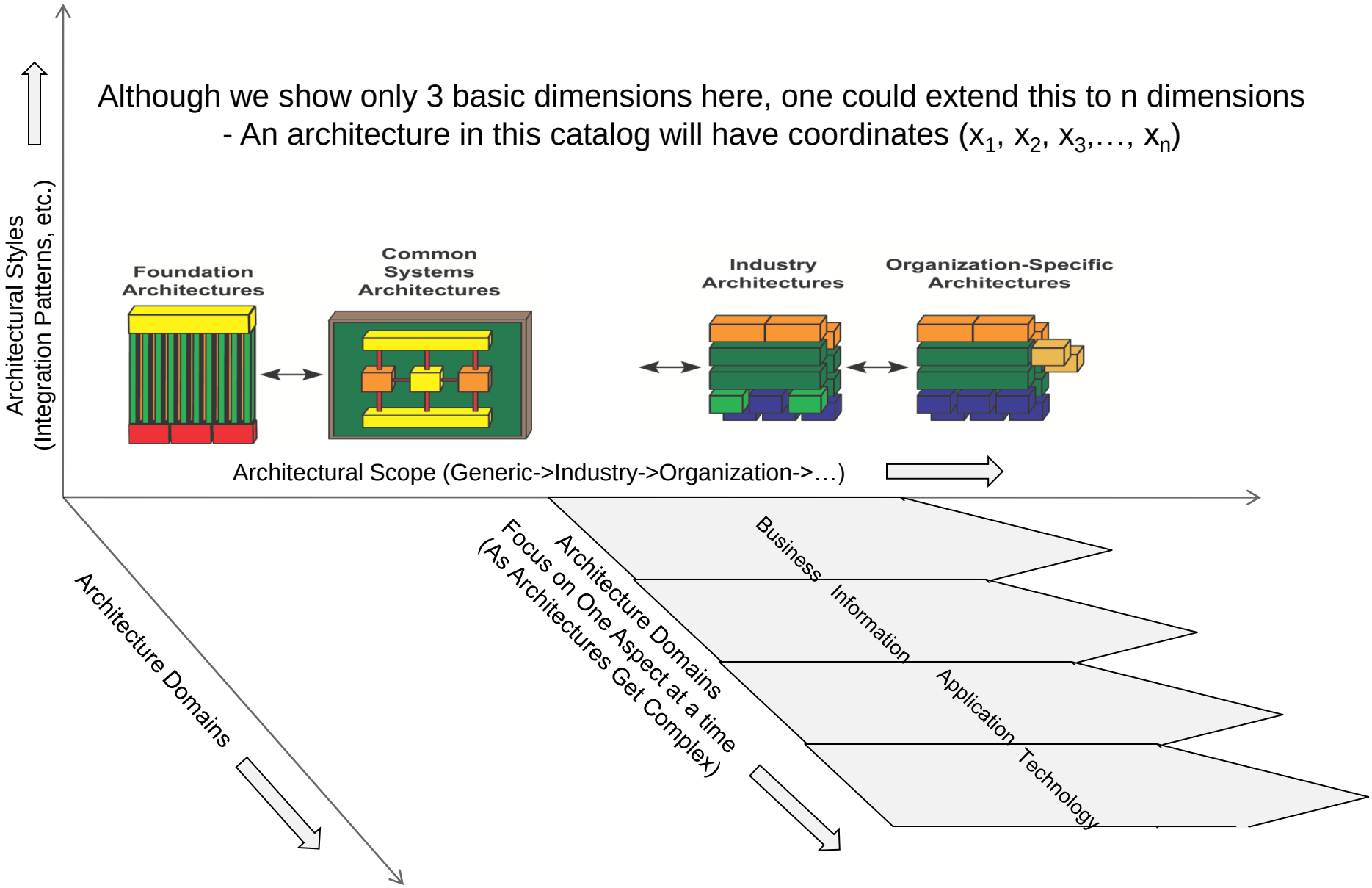
Specialization Hierarchy for Architectures (TOGAF-Centric)



Reference Architecture Cataloguing Framework

- Objectives:
  - *A catalog* with a scope comprehensive enough to hold all reference architectures blueprints in a meaningful and well-understood structure (i.e. be able to accommodate different types of reference architectures blueprints)
  - *A set of processes* to access and maintain the catalog as well as the reference architectures in the catalog, so the reference architecture blueprints could be easily preserved and reused, providing the following functions:
    - Retrieve a specific reference architecture at any time, given the dimension specifications
    - Searchable given any dimension specifications
    - Allow adding variants to any existing reference architecture
    - Extendable in terms of new options (attributes) in each dimensions

Reference Architecture Cataloguing Framework Dimensions

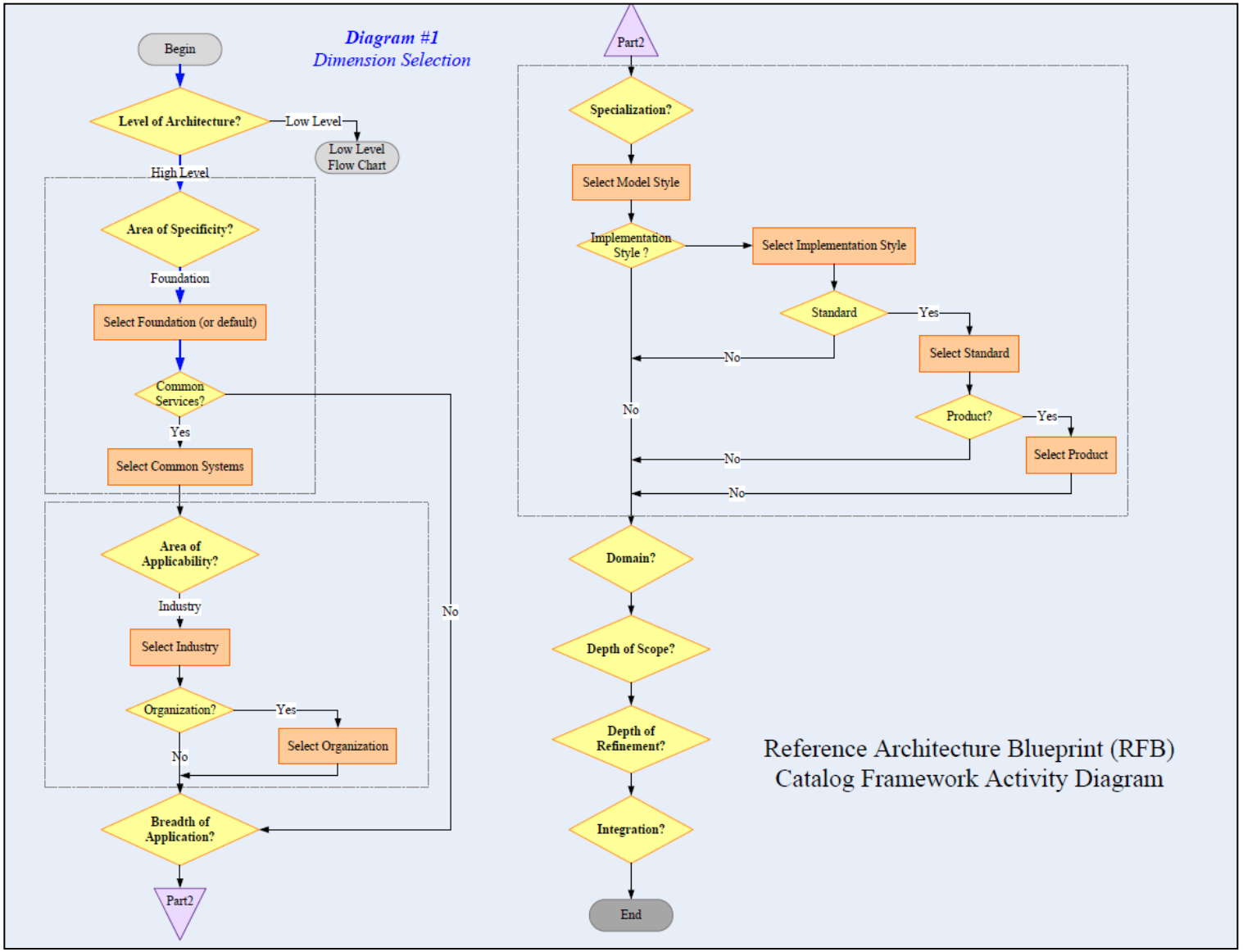


Reference Architecture Cataloguing Framework Dimensions

D1	D2		D3			D4	
Level of Architecture	Area of Specificity		Area of applicability			Breath of Applicability	
High	Foundation	common services	Industry	Finance	Organization	Enterprise	
Low	Generic	security			Mogan Stanley	Portfolio	
		management			Citi	Project	
		monitoring			…		
		etc		Telecom	AT&T		
	TOGAF	…			Verizon		

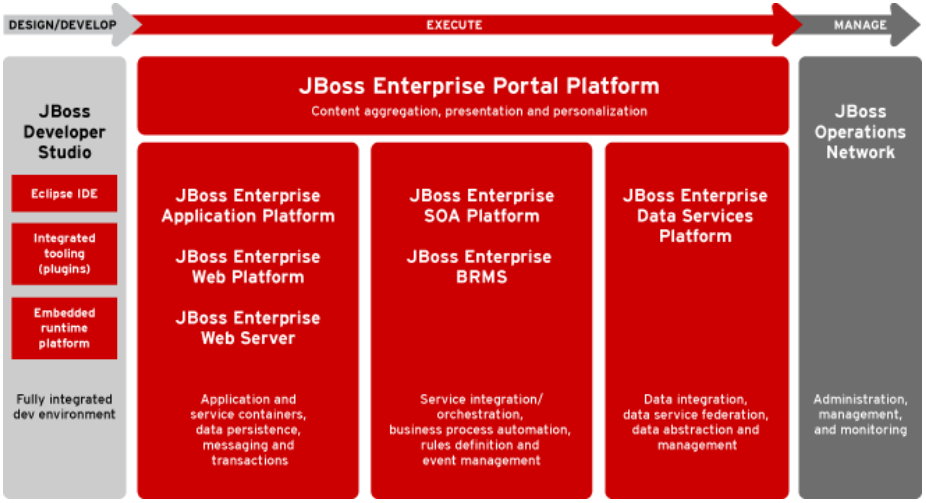
D5	D6	D7		D8		D9	D10
Domain	Depth of Scope	Specialization		Product Mapping		Level of Abstraction	Combination type
Business	services	Model Style	Implementation style	Standard	Product	presentation	integration
Application	facilities	Generic	Generic	N/A	N/A	conceptual	package
Technology		integration	Hybrid	Mix	Mix	logical	
Information		package	Package	COM+	MS DNA	physical	
Views		OMA	CORBA	JEE	IBM Websphere		
		SOA	ESB	CORBA2	VisiBroker		
		etc	etc	WS-I	.NET SOA		
				etc	etc		

Reference Architecture Catalogue Processes
Dimensions Selection

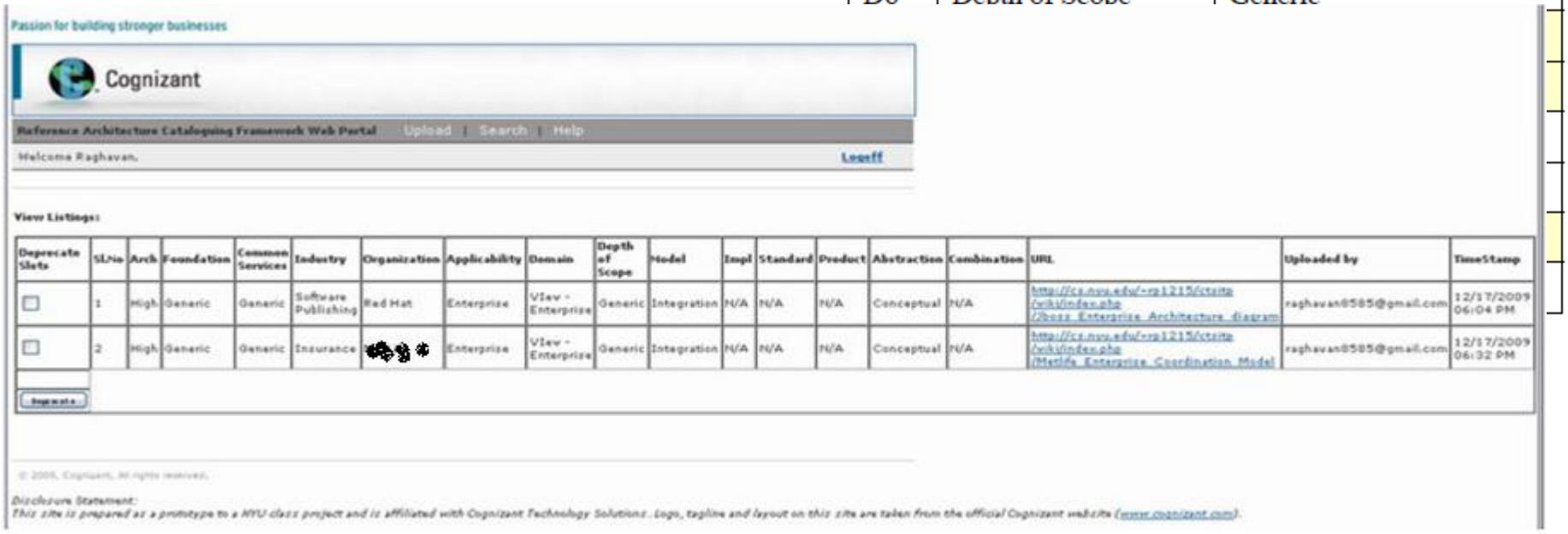


Reference Architecture Cataloguing Framework at Work

Sample from Joint NYU-CTS ITP Project (Fall 2009)



D1	Level of Architecture	High	
D2	Area of Specificity	Foundation	Generic
		Common Services	Generic
D3	Area of applicability	Industry	Software
		Organization	Red Hat
D4	Breath of Applicability	Enterprise	
D5	Domain	View-Enterprise	
D6	Depth of Scope	Generic	



Agenda

1. Introduction
2. High-Level Analysis and Design
3. Architecture Blueprinting
4. Sample Architecture Blueprints
5. Architectural Mapping Process Illustrated
6. Summary and Conclusion

Summary – Key High-Level Analysis and Design Objectives

- Enable Rapid Development of Business and Technical Solutions
- Quickly produce a high-level model that reflects the current understanding of the future state architecture
- Put together a high-level program/project estimate and provide a view of the future state that can be used as a starting point
- Leverage blueprinting notations/process and blueprints that have been standardized within the Enterprise working on the high-level analysis and design
- Follow a top-down process to document the various facets of the future state architecture starting from the Enterprise level and going through the business and technology architectures
- Conduct technology architecture blueprinting in parallel at the application, data, and technical levels

Course Assignments

- Individual Assignments
  - Reports based on case studies / class presentations
- Project-Related Assignments
  - All assignments (other than the individual assessments) will correspond to milestones in the team project.
  - As the course progresses, students will be applying various methodologies to a project of their choice. The project and related software system should relate to a real-world scenario chosen by each team. The project will consist of inter-related deliverables which are due on a (bi-) weekly basis.
  - There will be only one submission per team per deliverable and all teams must demonstrate their projects to the course instructor.
  - A sample project description and additional details will be available under handouts on the course Web site

Team Project

- Project Logistics
  - Teams will pick their own projects, within certain constraints: for instance, all projects should involve multiple distributed subsystems (e.g., web-based electronic services projects including client, application server, and database tiers). Students will need to come up to speed on whatever programming languages and/or software technologies they choose for their projects - which will not necessarily be covered in class.
  - Students will be required to form themselves into "pairs" of exactly two (2) members each; if there is an odd number of students in the class, then one (1) team of three (3) members will be permitted. There may not be any "pairs" of only one member! The instructor and TA(s) will then assist the pairs in forming "teams", ideally each consisting of two (2) "pairs", possibly three (3) pairs if necessary due to enrollment, but students are encouraged to form their own 2-pair teams in advance. If some students drop the course, any remaining pair or team members may be arbitrarily reassigned to other pairs/teams at the discretion of the instructor (but are strongly encouraged to reform pairs/teams on their own). Students will develop and test their project code together with the other member of their programming pair.

Team Project Approach - Overall

- Document Transformation methodology driven approach
  - Strategy Alignment Elicitation
    - Equivalent to strategic planning
      - i.e., planning at the level of a project set
  - Strategy Alignment Execution
    - Equivalent to project planning + SDLC
      - i.e., planning a the level of individual projects + project implementation
- Build a methodology Wiki & partially implement the enablers
- Apply transformation methodology approach to a sample problem domain for which a business solution must be found
- Final product is a wiki/report that focuses on
  - Methodology / methodology implementation / sample business-driven problem solution

Team Project Approach – Initial Step

- Document sample problem domain and business-driven problem of interest
  - Problem description
  - High-level specification details
  - High-level implementation details
  - Proposed high-level timeline

Assignments & Readings

- Readings
  - Slides and Handouts posted on the course web site
  -  Textbook: Part Two-Chapter 5
- Individual Assignment (due)
  - See Session 3 Handout: “Assignment #1”
- Individual Assignment (assigned)
  - Sess Session 5 Handout: “Assignment #2”
- Team Project #1 (ongoing)
  - Team Project proposal (format TBD in class)
  - See Session 2 Handout: “Team Project Specification” (Part 1)
- Team Exercise #1 (ongoing)
  - Presentation topic proposal (format TBD in class)
- Project Frameworks Setup (ongoing)
  - As per reference provided on the course Web site

Any Questions?

Next Session: Detailed-Level Analysis and Design