Essential Enterprise Architecture

October, 2005

Koehler Consulting, Inc.

Holliston, MA / Wellesley, MA

(508) 429-1589

www.koehlerconsult.com

Scott Koehler, President

Agenda

- Architecture
- Modeling & design
- Business logic
- Service oriented architecture / Web services

About Us

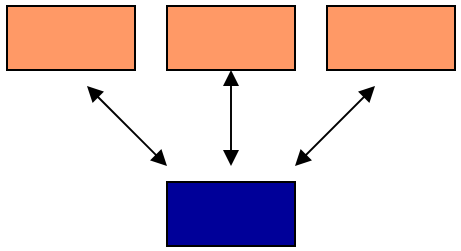
- Information Technology Consulting
- KCI was founded in 1990
- Modern systems development
- Financial services specialization
- website → www.koehlerconsult.com

Services We Offer

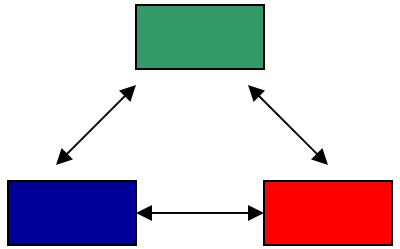
- Architecture & Design
  - Application Architecture / Technical Architecture
  - Enterprise modeling
  - Object Oriented Analysis & Design
  - Enterprise Application Integration
  - User Experience Design / Portal Design
- Technology Services
  - Application development (.NET, C#, Java)
  - Web development
- Software Development Lifecycle
  - Support for all phases of lifecycle including
    - Project Management
    - Business Analysis / System Analysis
    - Testing Strategies

Integration projects we've done

- Reusable business engine serving multiple business systems
  - legacy systems as clients

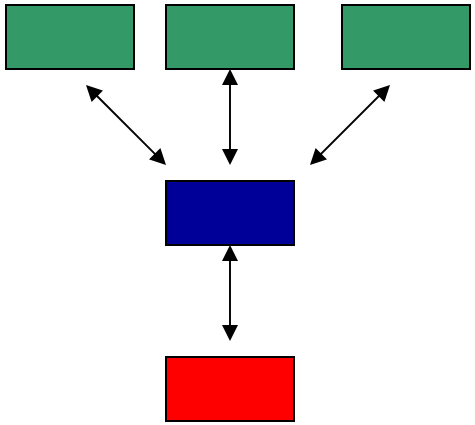


- Web-enabled business engine connected to workflow engine
  - sharing same database

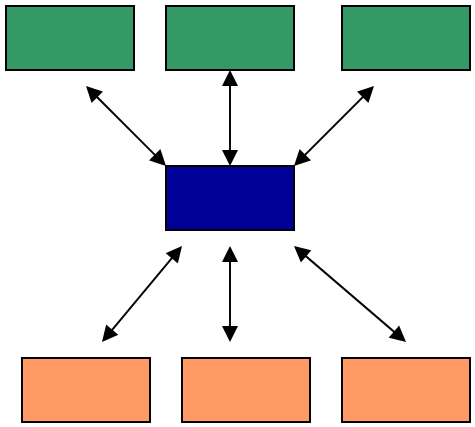


Integration projects we've done

- Integrating multiple user interface systems (call center, web) with backend business engine
  - previously stove pipes



- Integrating multiple product administration systems
  - supporting multiple user interfaces

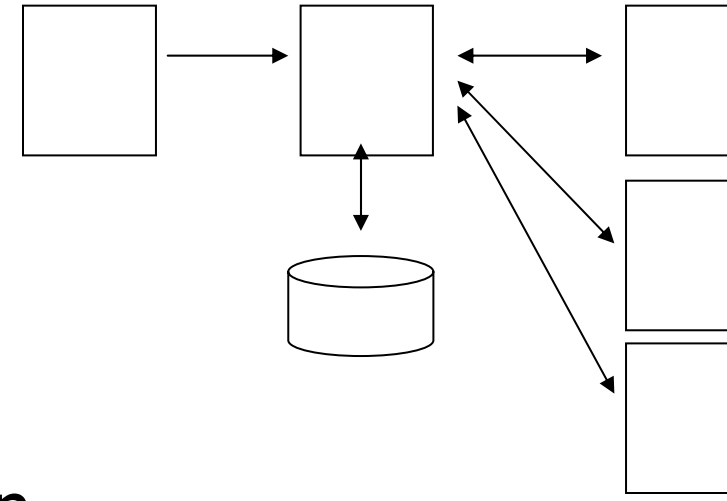


Application Architecture

- Break up business applications into parts
- Document current state
- Document desired future state
  - Architectural Blueprint
- 'Big bang' often not practical
- Use projects to help to move toward desired state
  - challenges in making this happen

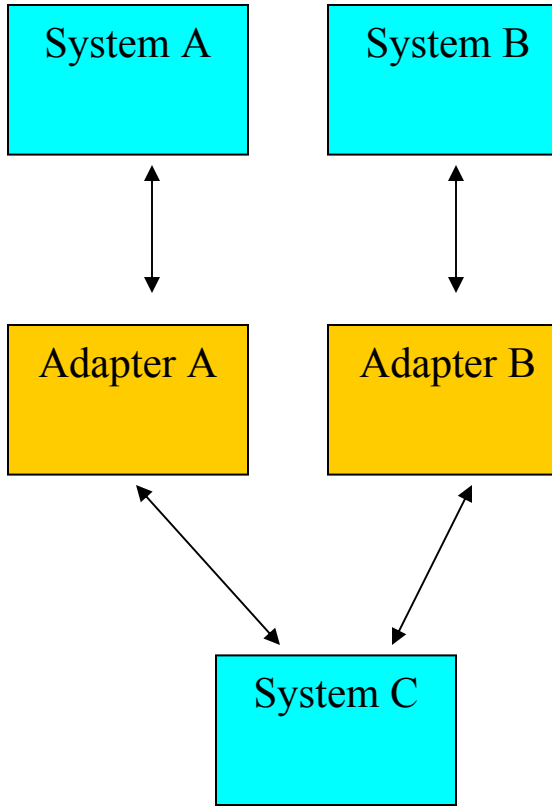
Application Architecture

- Draw a picture
- Define how components will interact (interfaces)
- Document description of each component
  - functionality, API, etc.
- Technology is a separate decision
- Reduce complexity / promote flexibility
- Development team can be organized by components
- Consider future business needs



Adapters / transformers

- Transform data from one system to another's format
- Maps to standard interface
- Analyze transactions
- Granularity differences
  - many → one
  - one → many
- Other complications
  - derivations due to missing data / transactions
- Common mistake → adapter logic in System C



Design Principles

- A written set of stated design objectives to use as a reference during the development phase
- Addresses current and future uses of the framework
- Affects the representation and placement of knowledge
- Influences design decisions

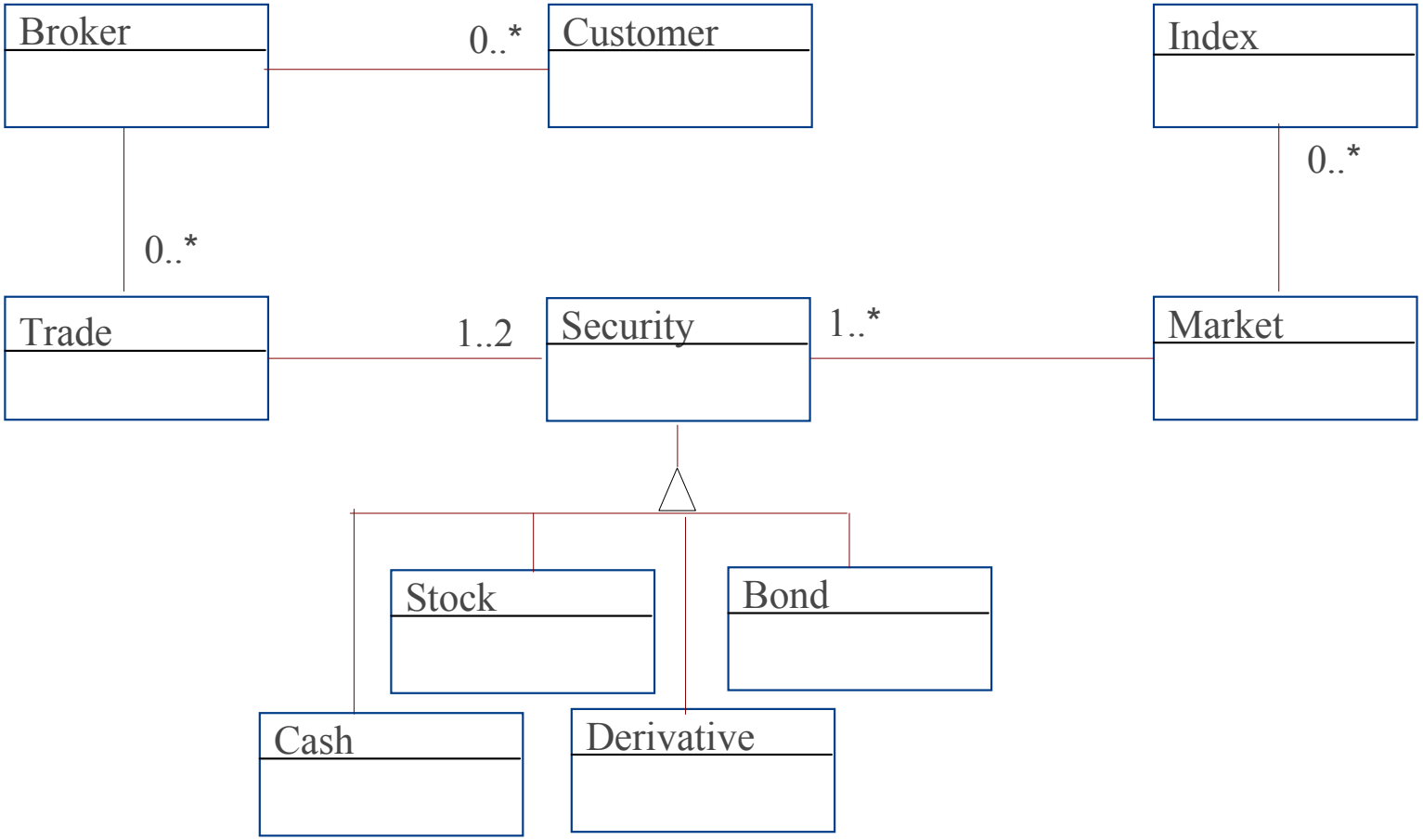
Design Principles

- *"The tax subsystem will contain the knowledge of the tax law with other interfacing subsystems exhibiting little or no knowledge of the tax law."*
- *Business logic will be contained in producing services, not consumers*

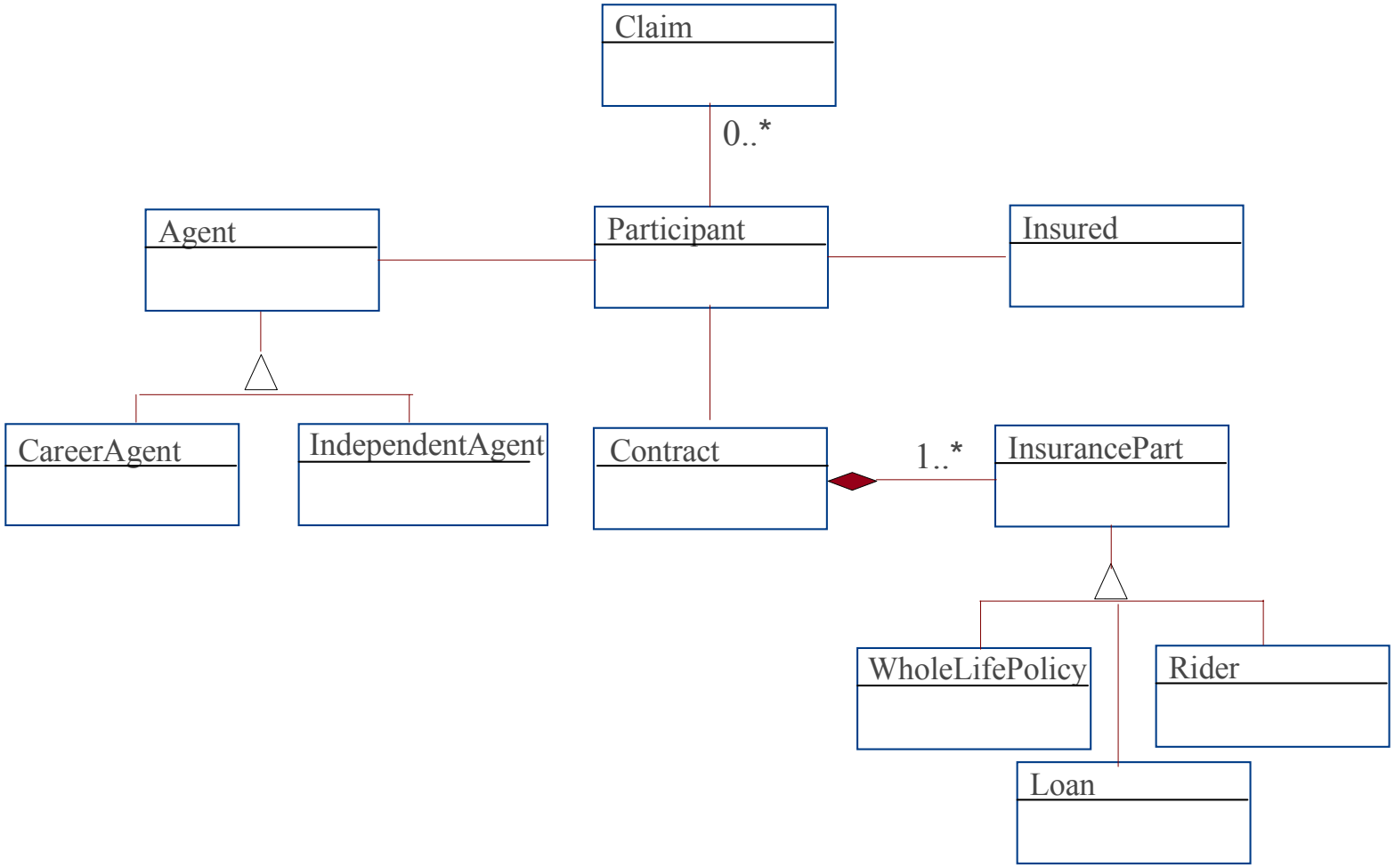
Enterprise Model

- Construct an Enterprise Model for a reference model
- Establishes a common vocabulary
  - the language of integration
- Depicts future state
- Business object model, data model, and XML schemas should be consistent with Enterprise model
- Services emit XML consistent with the model

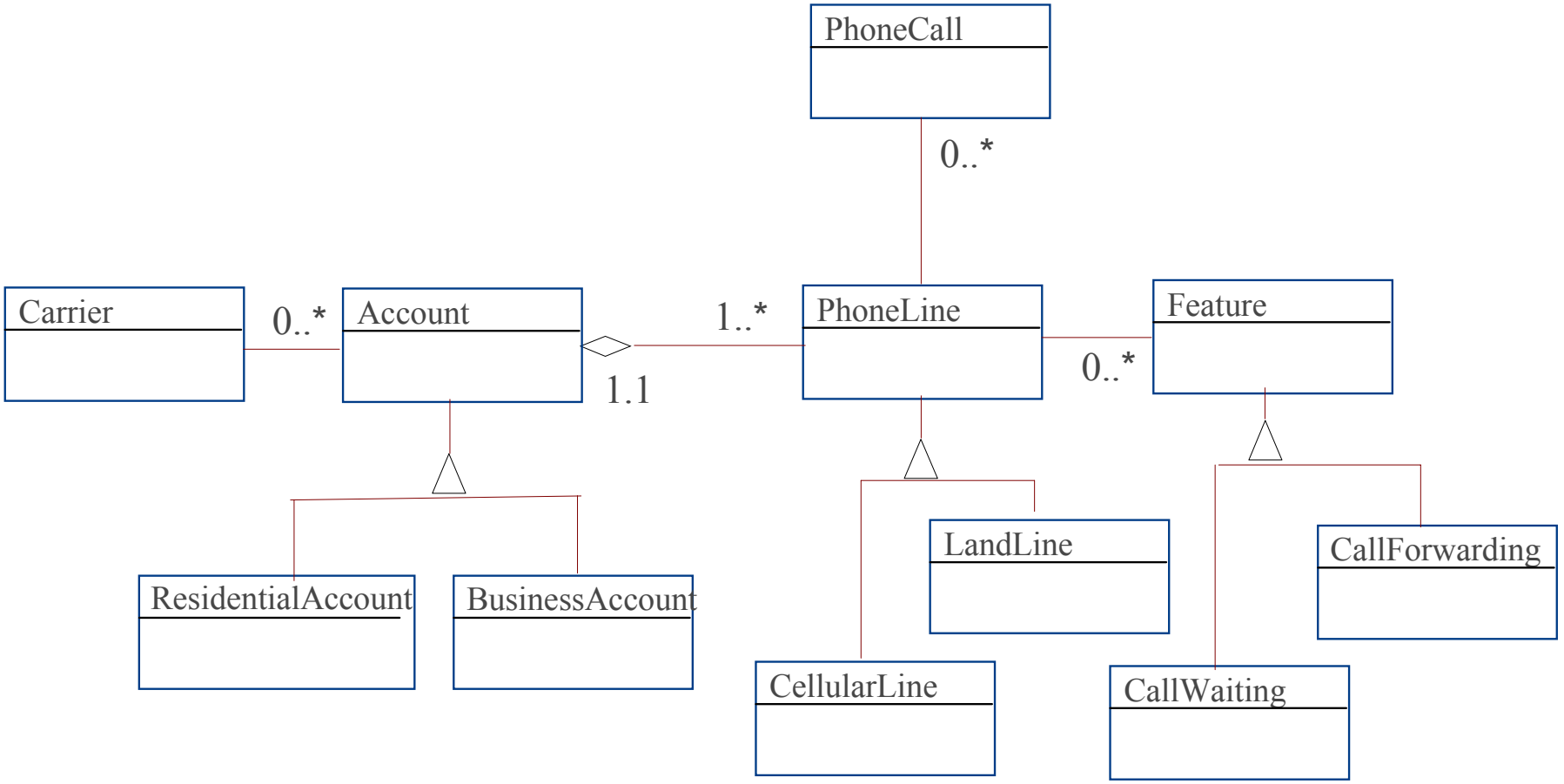
Enterprise Models - Finance



Enterprise Models - Insurance

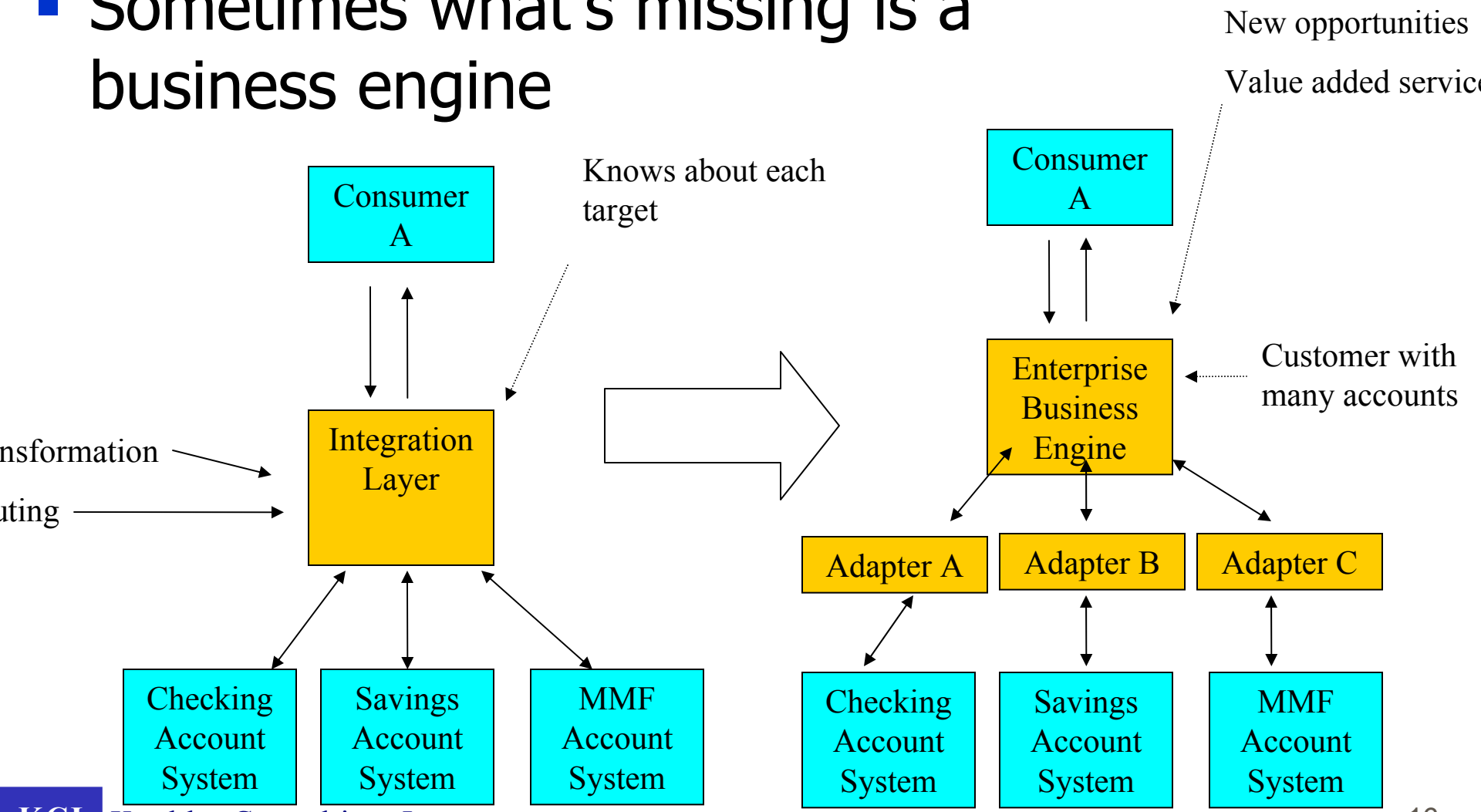


Enterprise Models - Telecom



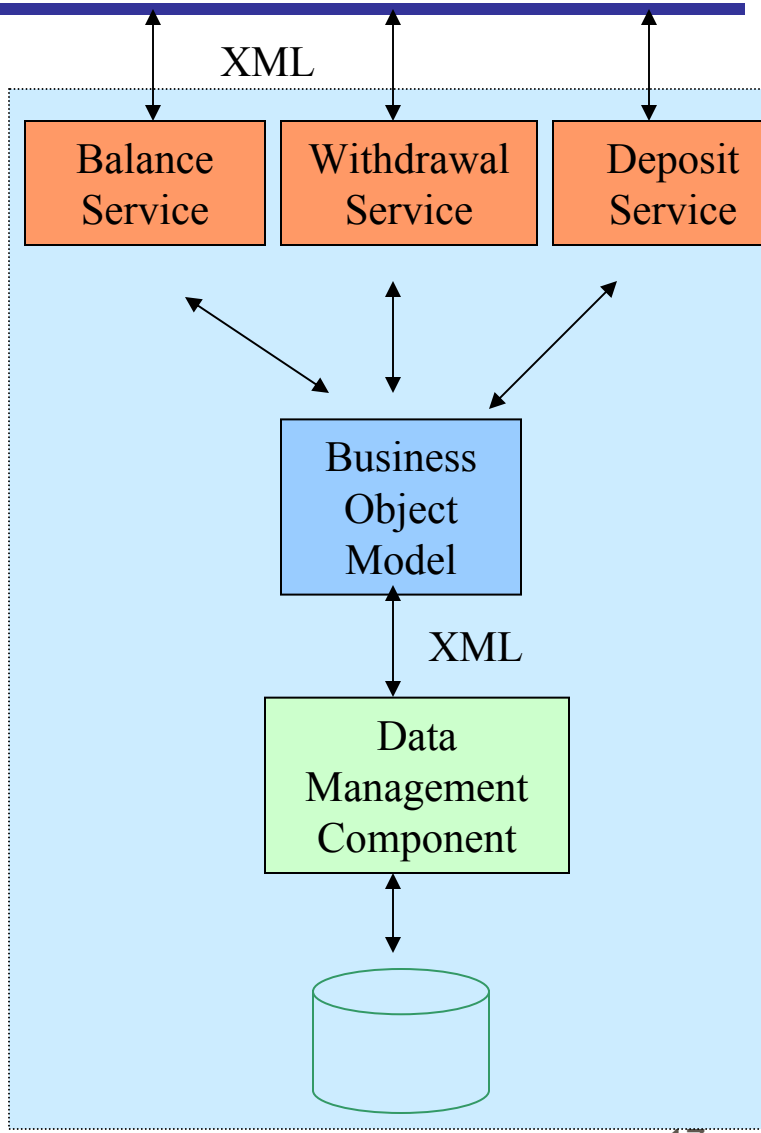
Business Logic

- Sometimes what's missing is a business engine



Business Logic

- Business Engine structure
- Typical design includes
  - Services operating on….
  - a Business Model reading and writing data to…
  - a Data Management component
  - XML for technology boundaries



Business Logic

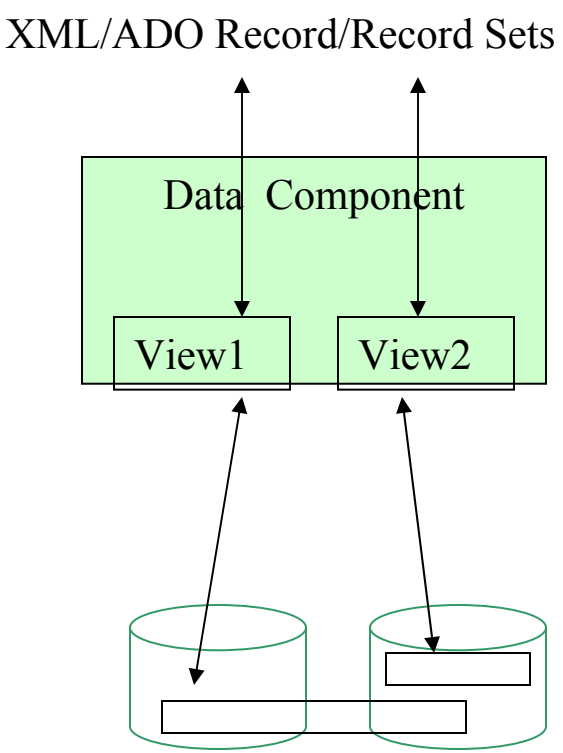
- Design techniques
  - Smart objects v. XML
    - Tradeoffs
      - serialization / deserialization
      - manage state
      - lazy initialization
    - proxied data
  - moving or transforming data versus business logic

Business Logic

- Design patterns
  - Reusable design techniques
  - Recurring techniques found in well designed OO software
  - Provides a name and description of the technique
  - Gamma, et al describe 23 patterns
- Ones we use frequently
  - Composite, Façade, State, Proxy, …

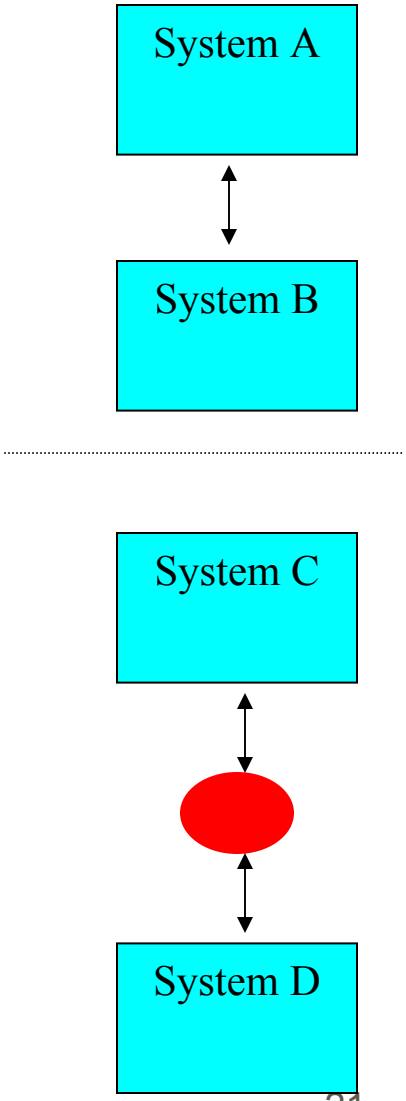
Data Management

- Data component to provide Views of data
  - insulates database(s) from application logic
  - no business logic
  - promotes flexibility
- Structured v. unstructured data
  - Permanent data → Structured
  - Transcient data → Structured or unstructured



Messaging Architecture

- Enables heterogeneous communication
  - opened up the mainframe
- Good for coarse grain payloads
- Features
  - Queues
  - Guaranteed delivery
  - Point to point
  - Publish & subscribe
  - Synchronous / asynchronous
  - Request / response
  - Message broker
  - Store and forward



Service Oriented Architecture

- Service: well defined, self-contained business function
  - Reusable across multiple business processes
  - Well defined interface
  - Promotes a loosely coupled architecture
  - e.g. credit check service, check history service
  - in the past, these were "components"
- Services know the process to be performed
- Entry into middle tier

Web Services

- Web service: a service accessed via XML over HTTP
  - expose services across network, across enterprise, and beyond enterprise boundaries
- Types of services
  - single step services → inquiry, update
  - multi-stepped (wizard like) services
    - Issues for state management
      - session state persisted to highly available store → database, file
      - session state retained in memory
      - session state emitted with response
- SOAP / WSDL

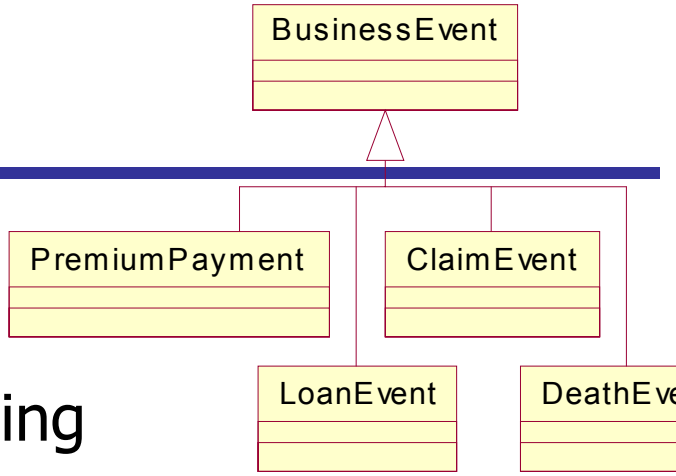
XML

- Useful for information exchange between components across major component boundaries
- Great for representing coarse grain information
- Raises the granularity of the interaction
  - "Business Forms" to deliver business object data
  - Business events

XML

- Hides internals of services
- Less sub-system dependency
- Accelerates / eases integration by providing stubs that return pre-fabricated XML
- Testing
  - create pre-fabricated test cases
  - compare actual result to expected result

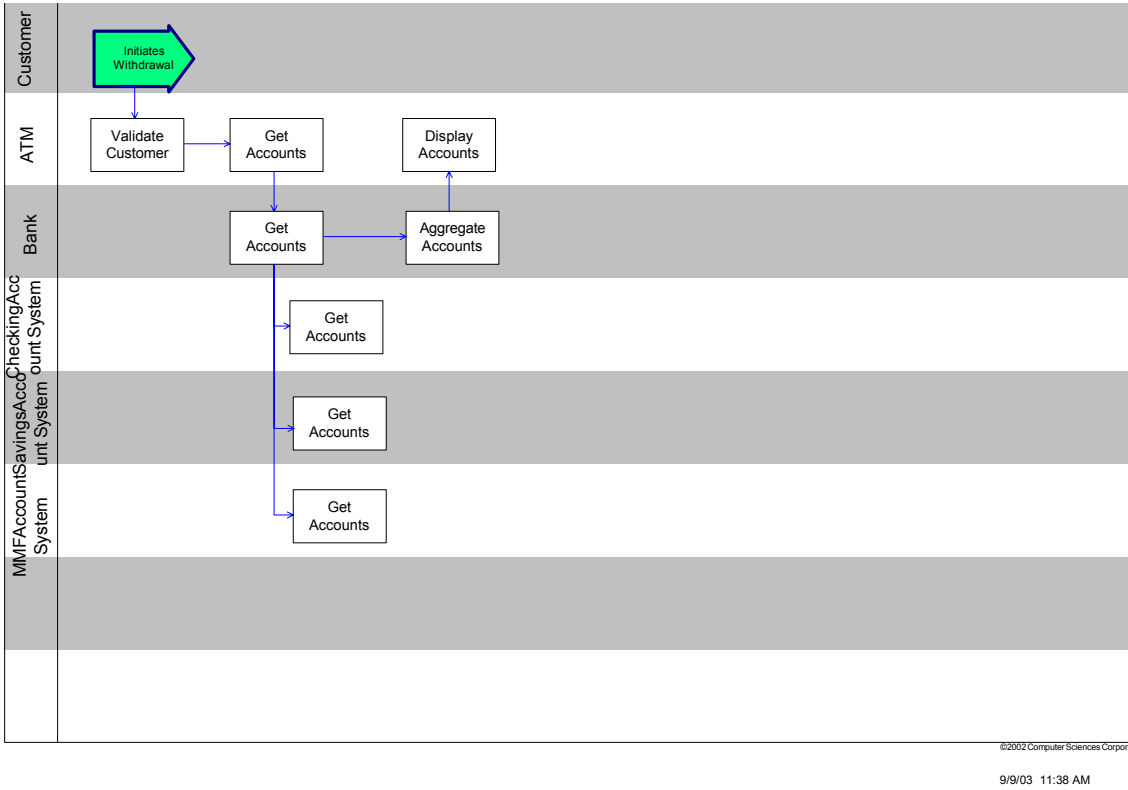
Business Events



- What's a business event?
  - a stimulus resulting from an external or internal occurrence that has meaning to the business
    - e.g. Deposit, Withdrawal, Marriage, Birth of child
  - carries a payload of information that defines the event
- Simple event, complex event
- Events trigger processes and other events
- Duration of an event can be short-lived or long-lived
- Events have a status (incomplete, complete, etc)
- Services can process events

Business Process Management

- Modeling the business process
- Business process map diagrams
  - Swim lanes
- Enterprise or integration lane may be missing or undefined
  - Common issue: draw them the way they exist now

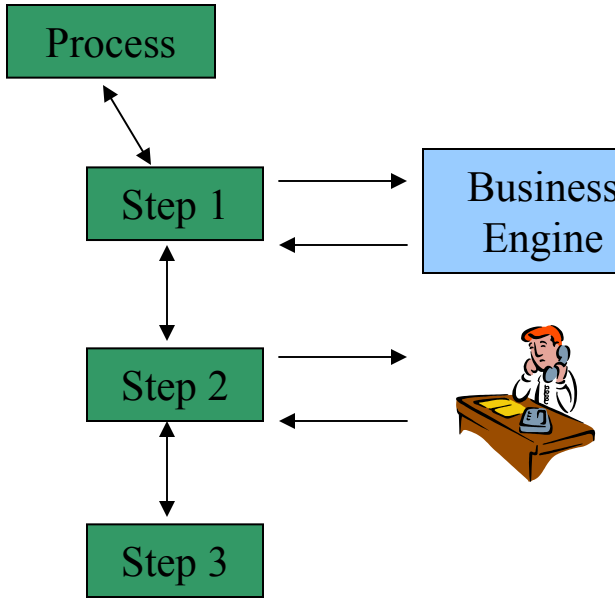


Business Process Management

- Tool capabilities
  - Visio templates
  - More advanced
    - modeling
    - meta data repository
    - simulation
    - export to workflow engine
    - monitoring

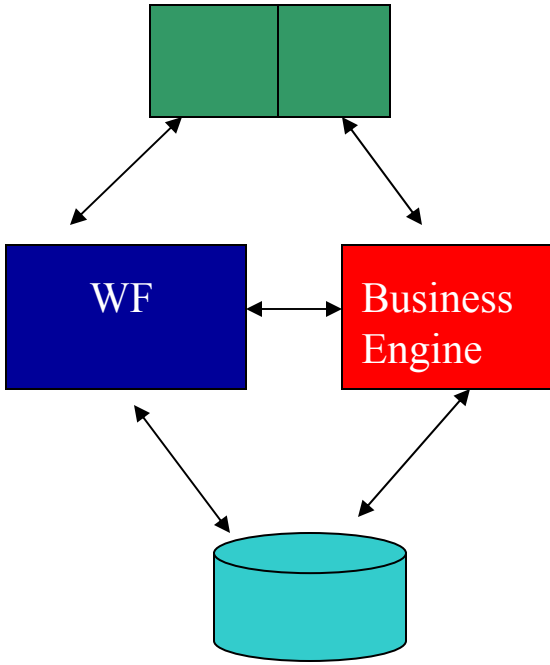
Integration with Workflow

- Human workflow
  - Inbox ToDo items
  - Insurance Claims
- Systematic workflow
  - WF engine "Orchestrates" the process
  - Order processing .... check inventory, credit rating, ...
- Workflow execution engine has control
- Executes steps in a process where each step could be a human interaction or system interaction
  - e.g. calling a web service



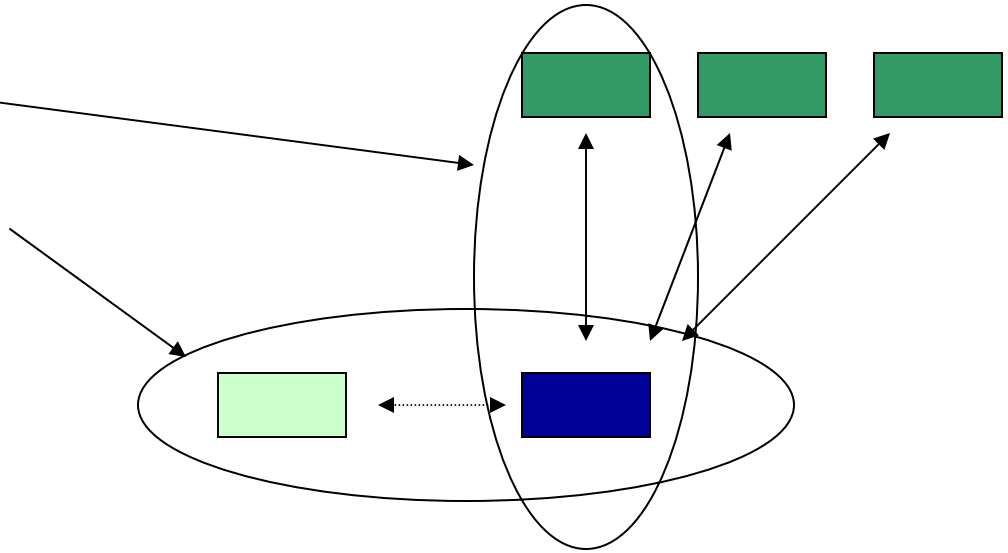
Integration with Workflow

- WF Tips
  - Include business policy in WF
    - e.g. if payment is > 10,000 route to manager for approval
  - Avoid complex business logic
  - Example
- Steps not = workflow engine
  - e.g. user interface interactions (steps)



Testing Strategies

- System Testing
- Component testing
- Automated testing



Enterprise Integration Artifacts

- We find these useful
  - Business process maps (swim lanes)
  - Use Cases
  - UML
    - Class diagrams
    - Instance diagrams
    - Sequence diagrams
    - an occasional state transition diagram
  - Data model
  - XML schemas

Integration Tools

- Capabilities
  - Enterprise Serviced Bus (ESB)
    - Transformation
    - Routing
    - Guaranteed message delivery
    - Logging, failover, load balancing
  - Business process execution (BPM)
  - Business activity monitoring (BAM)

Obstacles to adoption

- Reuse isn't free
- Some issues:
  - legacy thinking
  - dependent on another group for delivery
  - versioning
  - NIH

Questions / Comments