Wyyzzk, Inc.

Enterprise Architecture

Version 5.0

Lesson Goal & Objectives

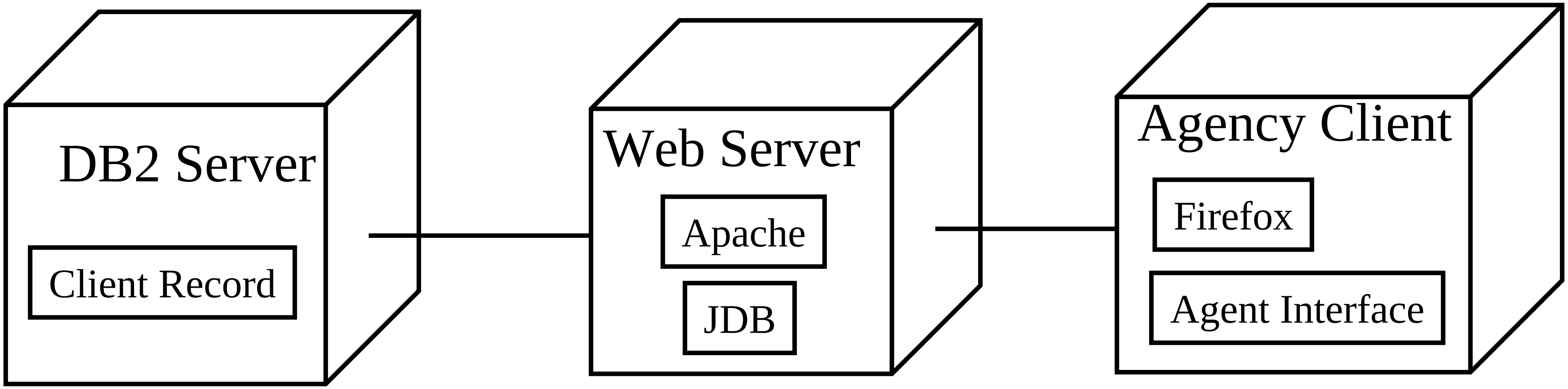
- Understand the purpose and nature of enterprise architecture.
- Upon completion of the lesson, the participant will be able to:
  - Describe architecture patterns at the enterprise level
  - Identify constraints and issues at the enterprise level

Lesson Outline

- Enterprise Architecture
  - What is enterprise architecture?
  - Patterns at the enterprise level
  - Constraints and Issues
  - Impact on Project Architecture

Enterprise Architecture

- An enterprise architecture concerns the hardware and software systems, applications, and data and their relationships across the whole enterprise
- This may or may not be formally described



Enterprise Architecture

- Infrastructure – for each computer
  - Name
  - Hardware and Operating System
  - Other environment information (database, TM)
  - Which systems access it
  - Which systems does it access
  - Where it is on the network, subnet, or cluster
  - Wired or wireless

Enterprise Architecture

- Applications
  - Applications and batch jobs
  - Where they are located
  - Who accesses them and how
- Information
  - Data, where it is kept
  - Who accesses the data and how

Enterprise Architecture

- An enterprise architecture will also typically include the important business processes of the company
  - Typically these are written as use cases
- The enterprise architecture may also include maintenance type use cases
- In addition, the enterprise architecture will include enterprise-wide constraints

Documentation

- Database
  - List of all computers with their characteristics
- UML Deployment Diagram
  - Nodes and connections of at least the primary nodes
  - Location of information and applications on nodes
- UML Use Case Diagram
  - Overview of business processes
- UML Component Diagram
  - Applications and their relationships
- ER Diagrams
  - Information and relationships between information
- MS Word
  - Use Case Specifications
  - Constraints, Regulatory Requirements, Guidelines

Break

- As a class, review the given Enterprise Architecture.

Patterns

- Some software architectural patterns are best applied at the enterprise level:
  - Service-oriented architecture (SOA)
  - Event-driven architecture (EDA)
  - Client/server (C/S)
  - Peer-to-peer (PTP)
  - Publish/Subscribe (pub-sub)
  - Broadcast

Service-oriented Architecture

- Architecture based on business services
  - Coarse grained
    - Process an application
    - Calculate a portfolio balance
  - Fine grained
    - Get customer record
    - Get account balance
    - Get someone’s public contact information

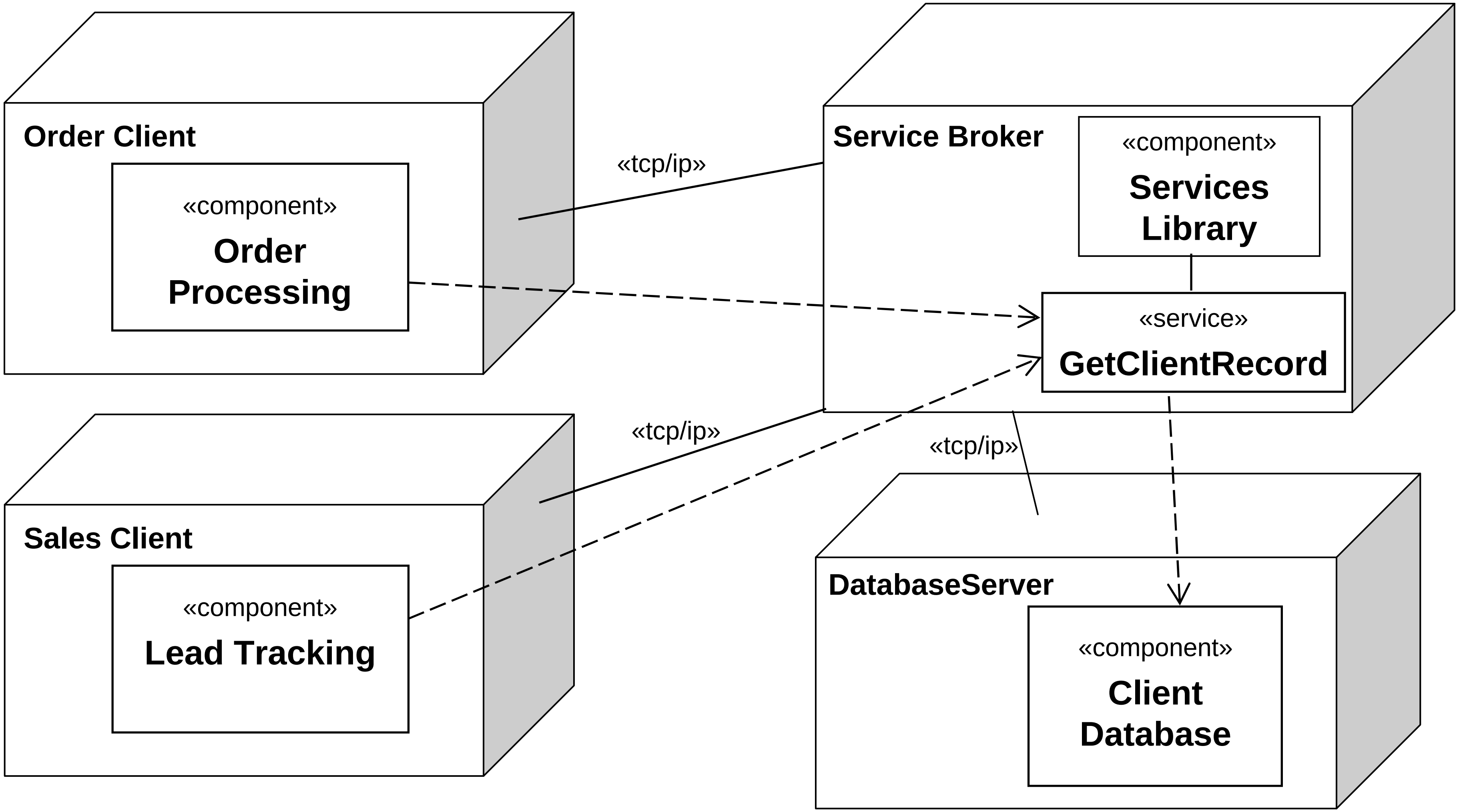
Service-oriented Architecture

- Focus is on what you want to do – business functions rather than business objects
- Inventory the systems of the company with the thought in mind as to what services each system can provide
- Consider the functions that are written over and over in your business applications
  - Try to make such a function a service and let all the applications call the one service

Service-oriented Architecture

- Consider a centralized service broker to store all the enterprise services
- What do you need to provide for development of new services?

Service-oriented Architecture



Event-driven Architecture

- Business events
  - Inventory below the minimum
  - Too many calls in wait queue
  - Receipt of an application
- System responds when an event happens
  - Also need to consider what if event does not happen or an error happens when processing the event

Event-driven Architecture

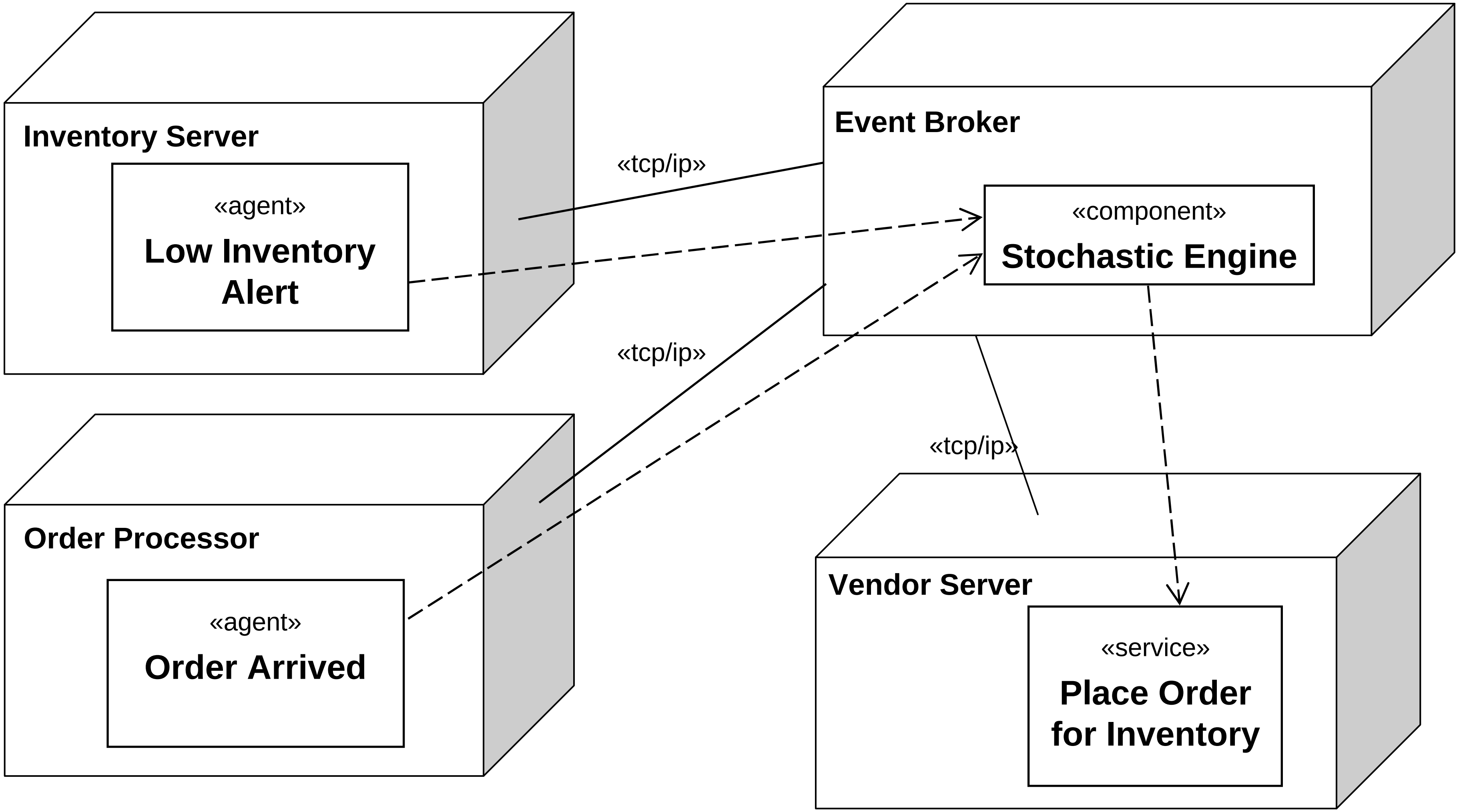
- Focus is on what events or triggers happen in the business – business functions rather than business objects
- Inventory the business processes of the company looking for events or triggers
- Consider the critical events
  - Provide functionality to handle those events

Event-driven Architecture

- Consider a centralized event broker to handle all the enterprise events
- What do you need to provide for development of new event managers?

Wyyzzk, Inc.

Event-driven Architecture

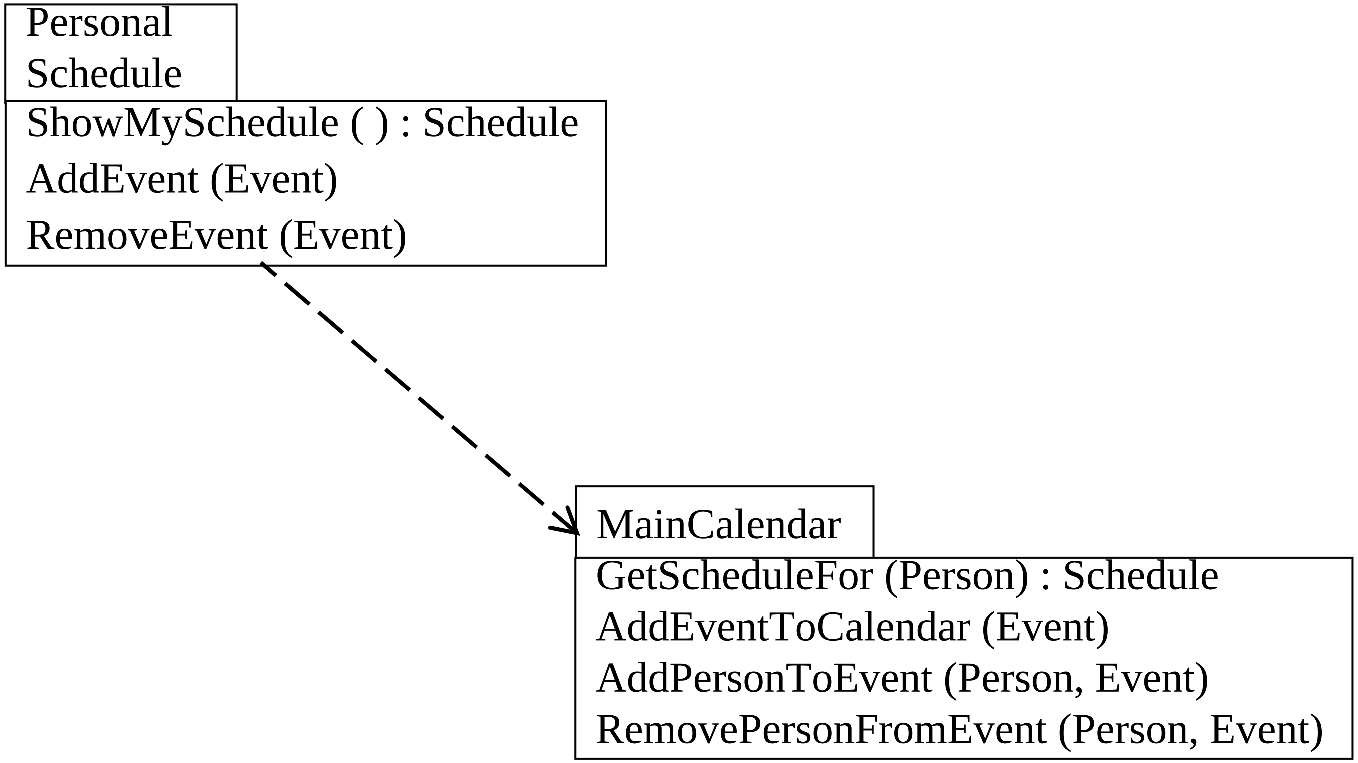


Client/Server

- Client/server refers to either or both of hardware and software
- In this pattern, one thing is a client which makes requests to another thing which is a server
- Requests go from client to server
- Responses go from server to client

Wyyzzk, Inc.

Client/Server Software Example



Personal Schedule

ShowMySchedule ( ) : Schedule

AddEvent (Event)

RemoveEvent (Event)

MainCalendar

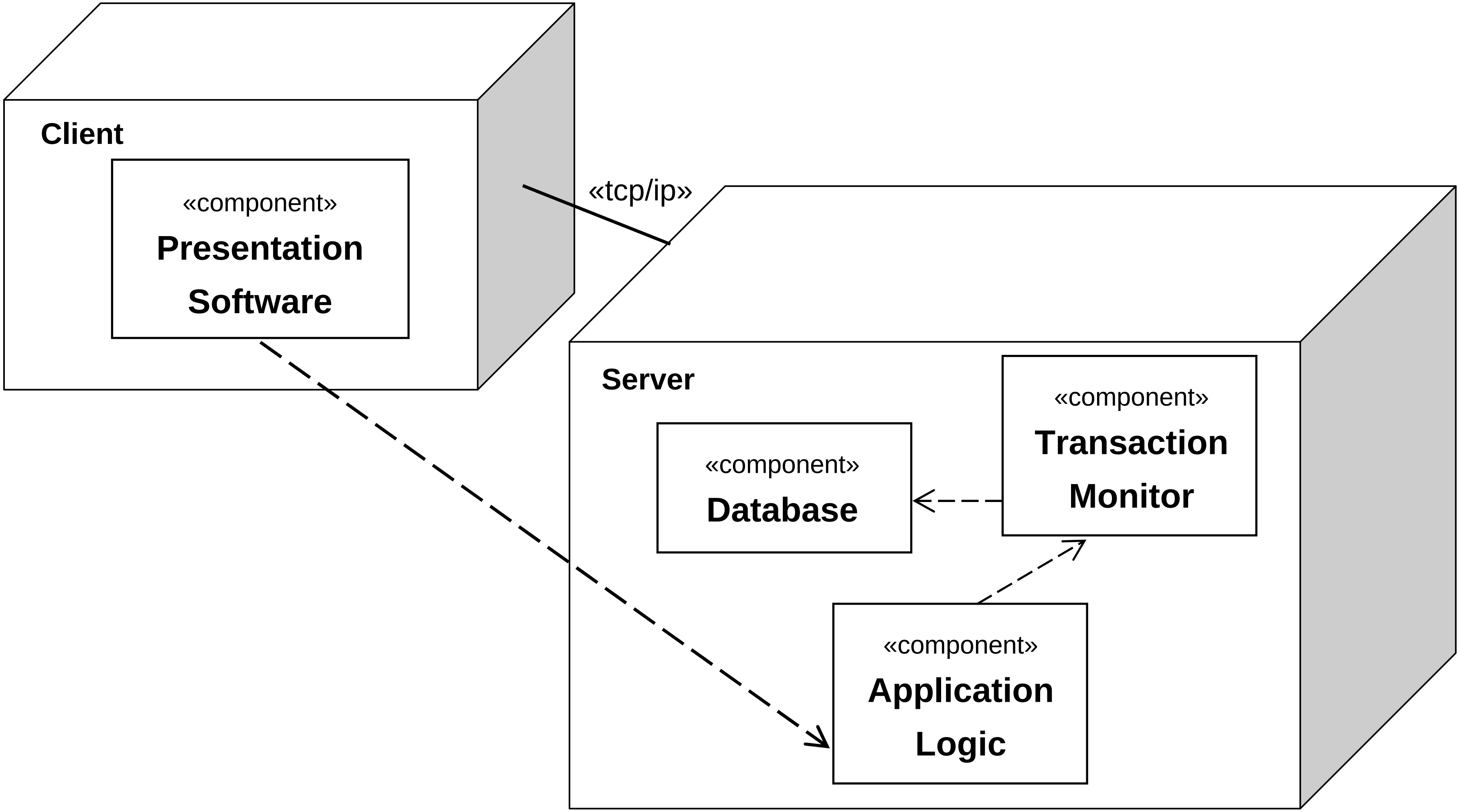
GetScheduleFor (Person) : Schedule

AddEventToCalendar (Event)

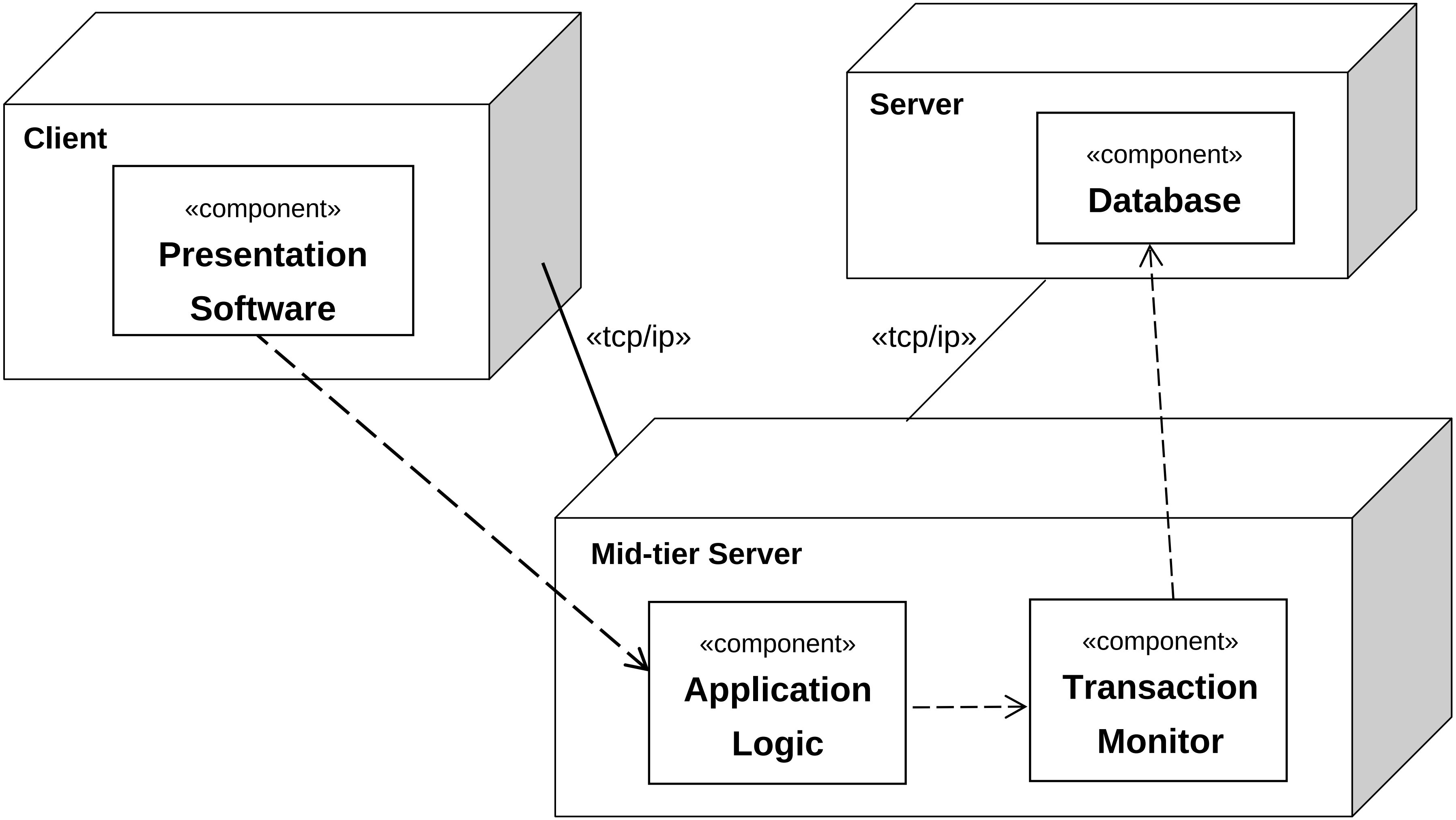
AddPersonToEvent (Person, Event)

RemovePersonFromEvent (Person, Event)

Client/Server Physical Example



N-tier Physical Example



Client/Server

- Business applications
- Client, Server
  - Client - the user interface, validation and verification rules for data
  - Server - application software and database
- The Client is typically distributed across a network from the Server

Strengths and Weaknesses of Client/Server Architecture

- Strengths
  - Leads to a consistent user interface across the whole application
  - Easy to understand
  - Relatively easy to program
  - Changes are relatively easy to make
- Weaknesses
  - Application software is closely tied to the database
  - Limits the processing power to two systems – client and server
  - Puts more load on the database server because it also handles application processing

n-Tier Architecture

- Business applications
- Presentation, Application, Database
  - Presentation - the user interface
  - Application - the business rules
  - Database - storage and retrieval of persistent data
- The Presentation tier only interacts with the Application tier
- The Application tier only interacts with the database

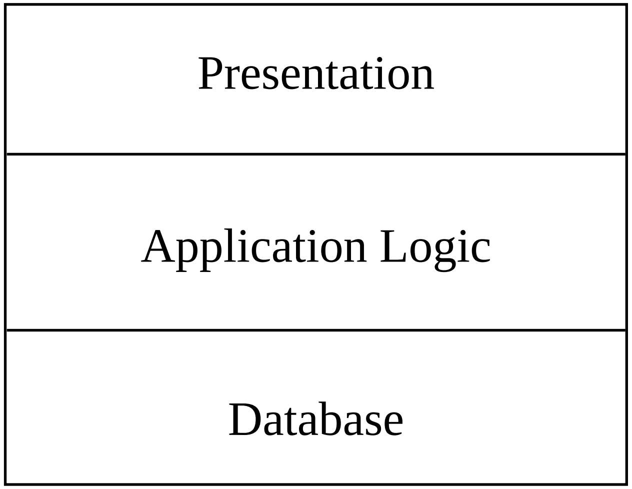
Strengths and Weaknesses of n-Tier Architecture

- Strengths
  - Since the presentation layer is separated from the database by the application tier it is easy to change the look and feel or the database with relatively minor effects on the rest of the application
  - Leads to a consistent user interface across the whole application

- Weaknesses
  - Usually requires some kind of transaction management service to track transactions from presentation tier to database
  - Changes in functionality typically require changes to all 3 tiers of the architecture

Applications

- Most business applications are built with a 3 tier (or more) software architecture
  - The presentation tier allows the user to view results and / or the information in specific business objects
  - The application tier contains the rules for manipulating the business objects
  - The database tier contains the business objects



Presentation

Application Logic

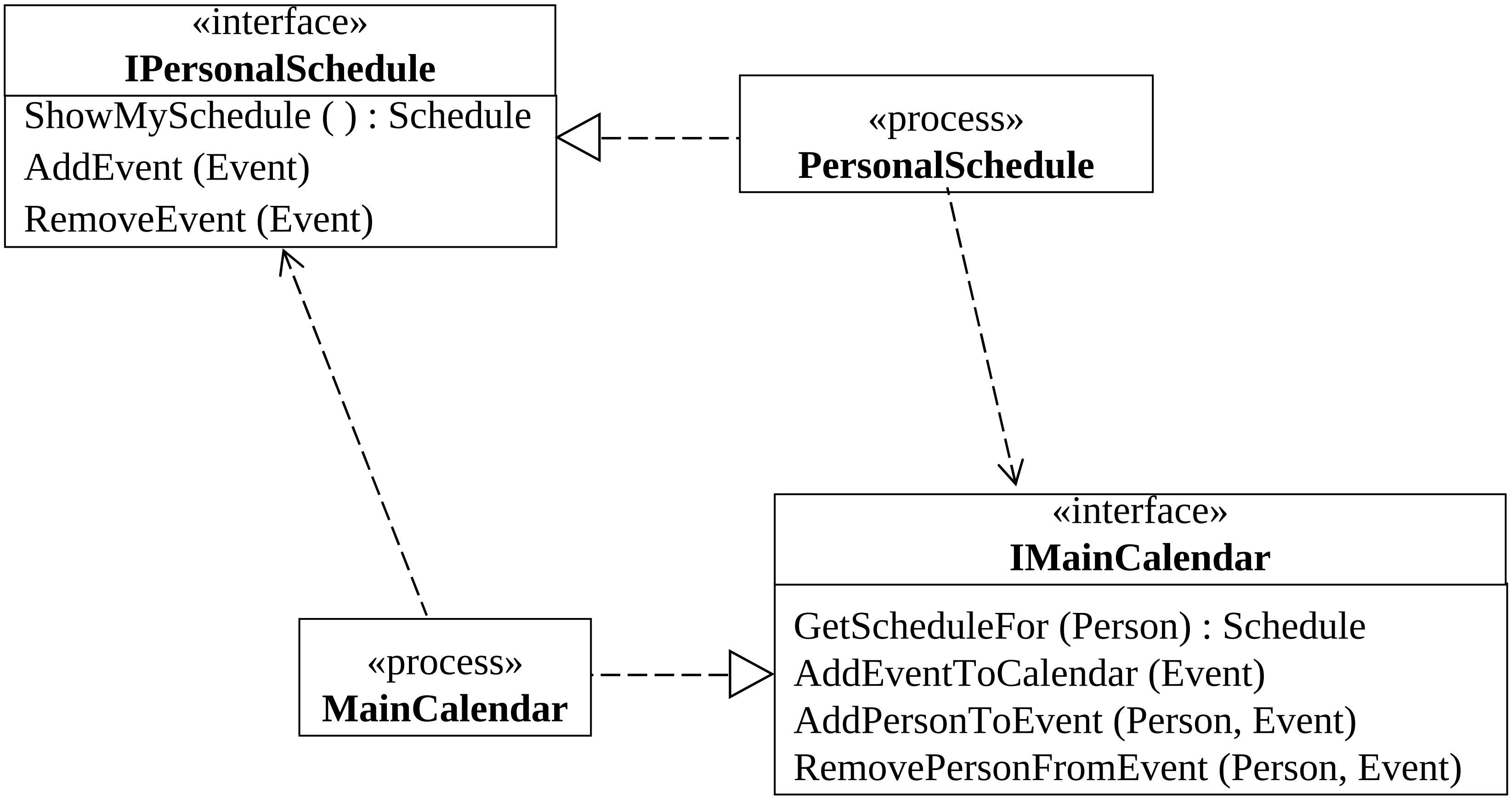
Database

Peer-to-peer

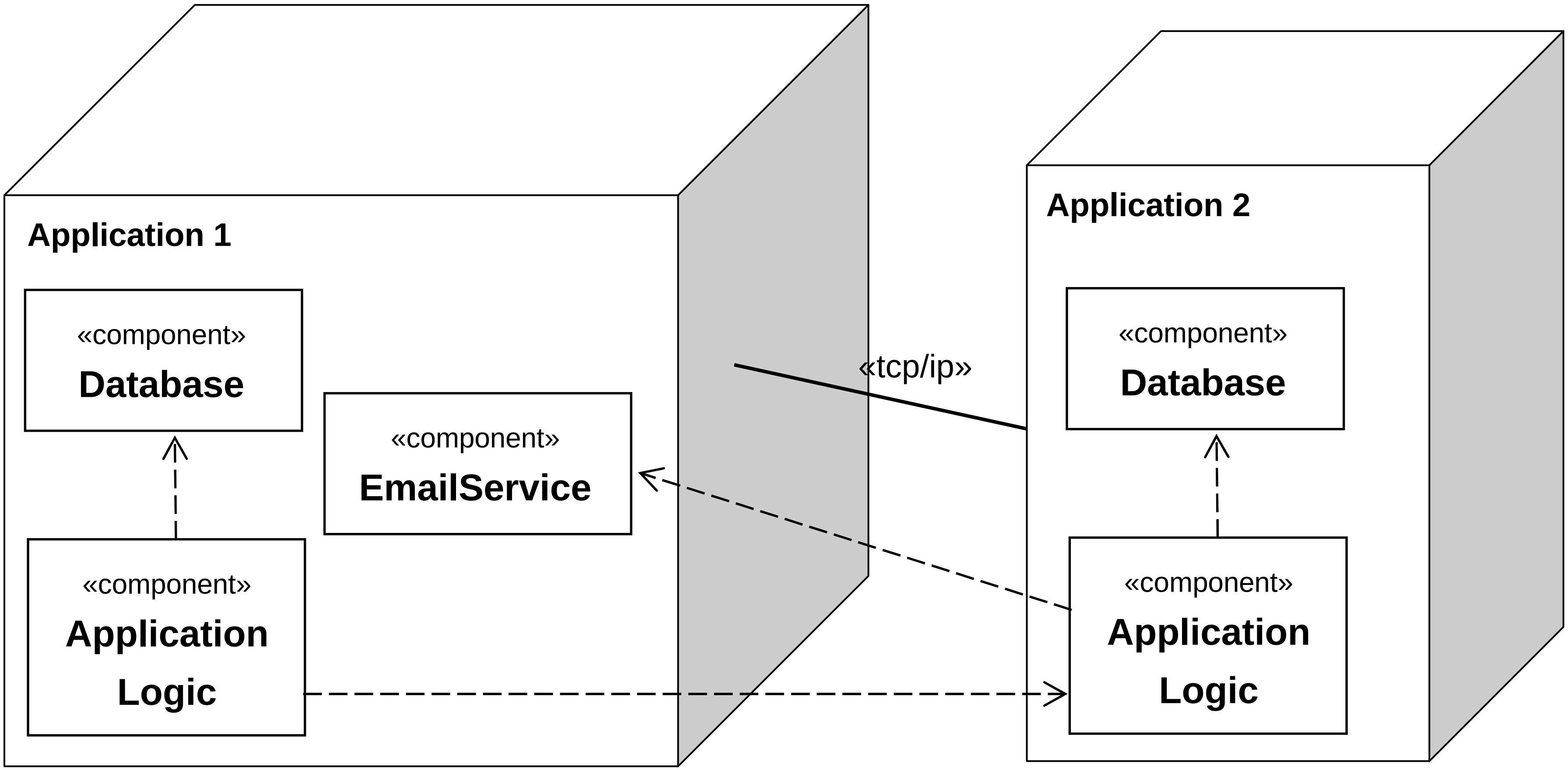
- Similar to client server in that two systems communicate directly with each other
- Both systems act as both client and server
  - Both systems make requests
  - Both systems send responses
- Like client/server, peer-to-peer can be hardware, software or both

Wyyzzk, Inc.

Peer-to-peer Software Example


«interface»
IPersonalSchedule
ShowMySchedule ( ) : Schedule
AddEvent (Event)
RemoveEvent (Event)
«process»
PersonalSchedule
«interface»
IMainCalendar
GetScheduleFor (Person) : Schedule
AddEventToCalendar (Event)
AddPersonToEvent (Person, Event)
RemovePersonFromEvent (Person, Event)
«process»
MainCalendar

Peer-to-Peer Physical Example



Peer-to-Peer

- Community/Personal networking applications
- Wireless networks
- The peers are typically distributed across a network from each other

Strengths and Weaknesses of Peer-to-peer Architecture

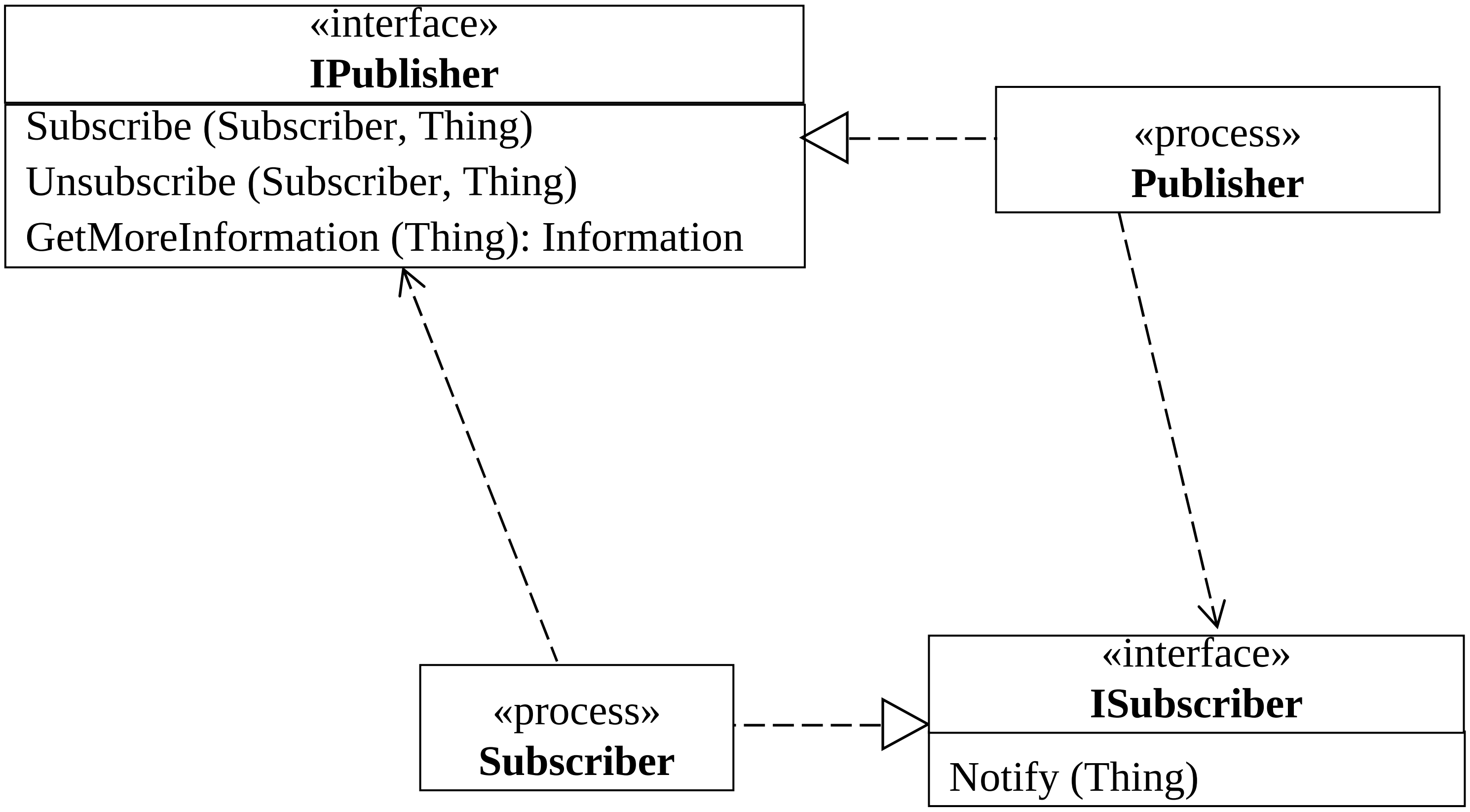
- Strengths
  - A lot of flexibility for designing both hardware and software architectures
  - Easy to understand
  - Good architecture match for wireless applications

- Weaknesses
  - Coordination of communication can be difficult
  - Prone to blocking timeouts
  - Harder to load balance since any machine can be a client or a server or both at once

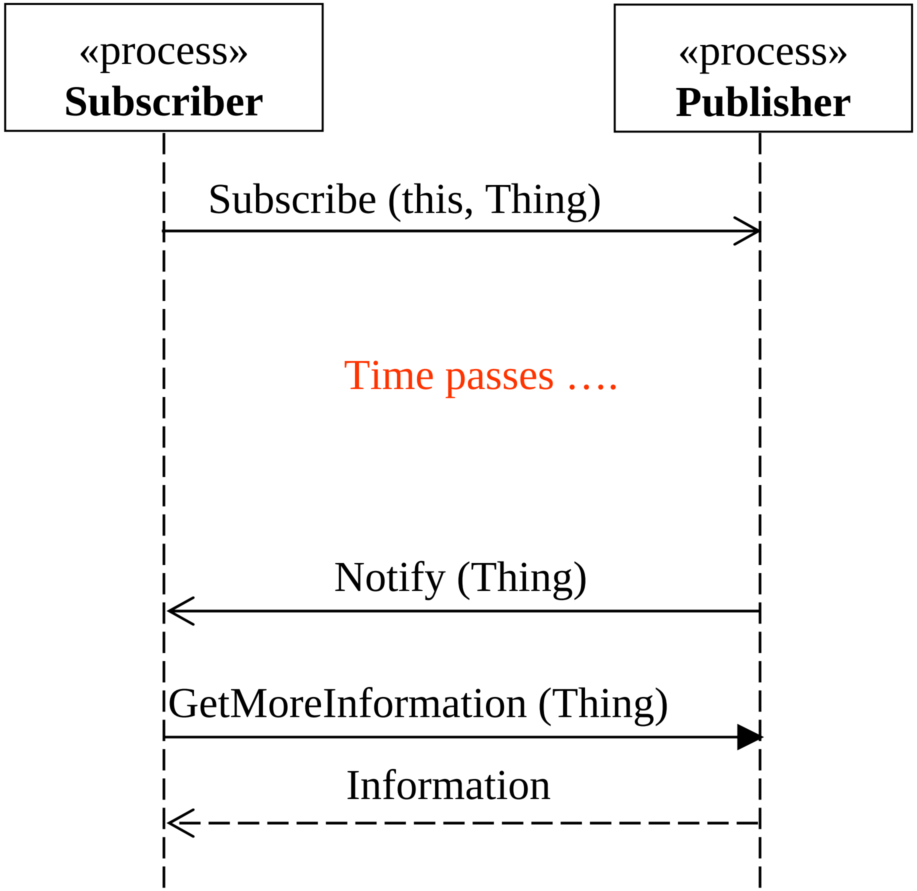
Publish/Subscribe

- One system is a publisher of information; any number of systems can subscribe to the information
- Publisher/Subscriber
  - Publisher – provides a means for other systems to subscribe to something of interest, notifies subscribers when that something of interest occurs
  - Subscriber – is interested in some information on another system
- Publish/Subscribe is typically a software architecture

Publish/Subscribe Software Example


«interface»
IPublisher
Subscribe (Subscriber, Thing)
Unsubscribe (Subscriber, Thing)
GetMoreInformation (Thing): Information
«process»
Publisher
«process»
Subscriber
«interface»
ISubscriber
Notify (Thing)

Publish/Subscribe Software Example


«process»
Subscriber

«process»
Publisher

Subscribe (this, Thing)

Time passes ….

Notify (Thing)

GetMoreInformation (Thing)

Information

Publish/Subscribe

- Business applications
- Centralized datastore or event broker
- The different systems are typically distributed across a network from each other

Strengths and Weaknesses of Publish/Subscribe Architecture

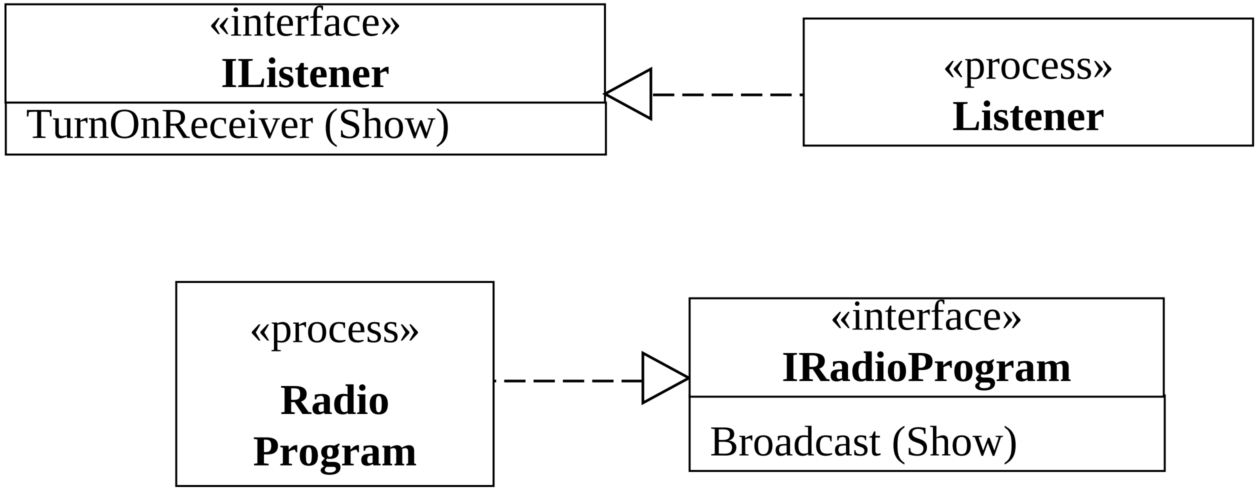
- Strengths
  - One point of maintenance for information
  - Consistency in accessing information
  - More efficient use of resources than polling schemes

- Weaknesses
  - One point of failure – the publisher
  - All subscribers are notified of a change of state of information and have to query to find out if it is actually of interest

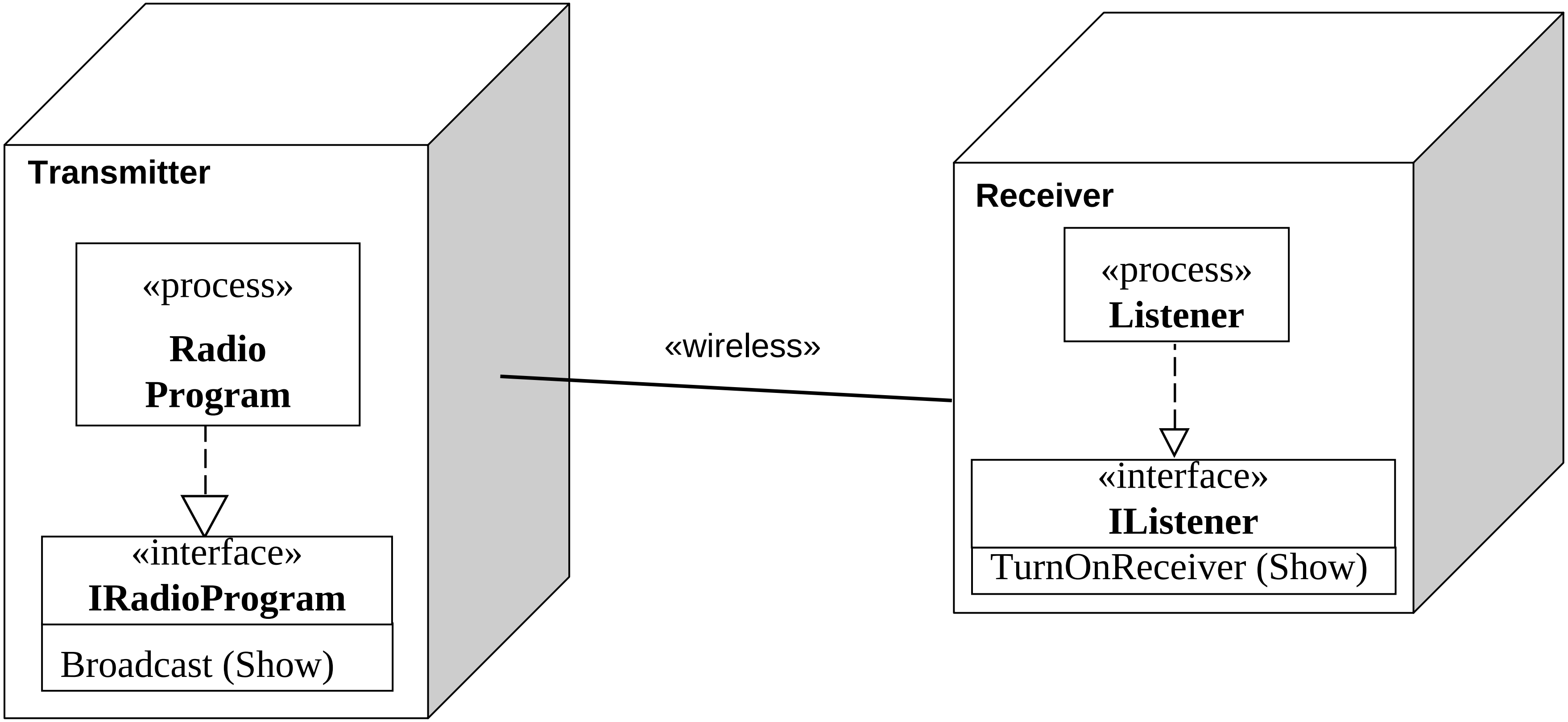
Broadcast

- Similar to publish and subscribe since there is a publisher of information and other systems are interested in that information
- Transmitter, Receiver
  - Transmitter – sends out information
  - Receiver – decides whether or not to listen, decides what to do with information received
- This is often used for communication at the network (packet) level
- It tends to be a hardware architecture, but may also appear in software

Broadcast Software Example



Broadcast Physical Example



Broadcast

- Network
- Packet Routing

Strengths and Weaknesses of Broadcast Architecture

- Strengths
  - No need to synchronize communication
  - Easy to by-pass non-working receivers
  - Good architecture match for wireless applications

- Weaknesses
  - Receivers have to examine all information received to decide if it is useful
  - No way to verify that a receiver has received some particular information

Break

Constraints and Issues

- Regulatory
  - Sarbanes/Oxley
  - HIPPA
  - Privacy laws
- Security
  - Limited access – physical, electronic
  - Encryption – data, communications
- Policy
  - Required performance
  - Required uptime
  - Fault tolerance
  - Budget for new hardware or software
  - Technical support of systems
  - Centralized vs distributed
  - Black-out periods for new releases

Constraints and Issues

- Licensing
  - Servers
  - Usage compliance
- Maintenance
  - Status monitoring
  - Mirroring & backups
  - Scheduled down-time for hardware/software updates
  - Run-time updates
  - Allocation of maintenance costs
- Technical Support
  - User queries
  - Problem resolution (possibly over many systems)

Enterprise Architecture Group

- Responsible for setting, documenting, and enforcing all enterprise level policies for the corporate systems
- Oversee project teams to verify that project architecture does not violate enterprise architecture
- Determine when enterprise architecture needs to change and how to change it
- Oversee all changes to enterprise architecture

Impact on project architecture

- Requirements of enterprise architecture often conflict with project requirements
- Need to negotiate problem areas with many groups
  - Corporate Security
  - Technical support (for users)
  - Maintenance group
  - Accounting
  - Regulatory groups
  - Enterprise architecture group
  - Database architecture group

Impact on Project Architecture

- Patterns selected at the Enterprise level have to be followed at the project level
  - For example, if the Enterprise uses SOA, then your project will be designed around services
- Constraints and regulatory requirements set at the Enterprise level have to be followed at the project level
  - For example, privacy laws require personal information to be encrypted. If your project in any way uses personal information, you will have to deal with decryption and encryption, and possibly only certain people on your team will have access to that data.

Impact on Project Architecture

- Because of the need to have more people involved with your project, your schedule will be longer
  - Enterprise architects, security people, regulatory agencies, and so on
- In the defense industry, you can add the need for clearances to the process
  - It is possible that team members have different clearances and need to know, and that you will have to put processes in place to ensure everyone has the appropriate access

Summary

- An enterprise architecture concerns the hardware and software systems, applications, and data and their relationships across the whole enterprise
- Patterns such as service-oriented architecture, event-driven architecture, client/server, publish/subscribe, peer-to-peer and broadcast can provide an overall structure to an enterprise architecture
- Constraints, issues, and policies set at the Enterprise level will effect your project architecture
- You may find that many groups will be involved in your project. For example, enterprise architecture, corporate security, regulatory agencies, technical support, maintenance, and database architecture.