



Slide 1



Enterprise Architecture

Copyright © Wyzzk, Inc. 2004
Version 5.0

Slide 2




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

Copyright © Wyzzk, Inc. 2004
Version 5.0

Enterprise Architecture - 2

Slide 3



Wyzzk, Inc. Lesson Outline

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

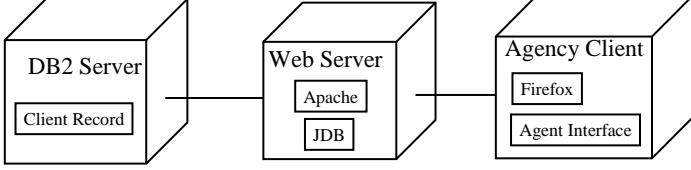
Copyright © Wyzzk, Inc. 2004
Version 5.0

Enterprise Architecture - 3

Slide 4

Wyzask, Inc. 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



The diagram illustrates a three-tier enterprise architecture. It consists of three main components connected in a line by horizontal lines. The first component is a 'DB2 Server' box containing a 'Client Record' box. The second component is a 'Web Server' box containing 'Apache' and 'JDB' boxes. The third component is an 'Agency Client' box containing 'Firefox' and 'Agent Interface' boxes.

Copyright © Wyzask, Inc. 2004
Version 5.0

Enterprise Architecture - 4

It is seldom that an enterprise architecture is planned. More often, it develops over time as systems and software are purchased and connected for individual products. Places like Safeway (retail grocer), banks, insurance companies, and investment companies have been purchasing systems and software for decades, usually without an overall plan. One recent exception is WalMart. This is a relatively new company, founded after computers were in common use in retail. So the enterprise architecture was planned not long after the company was founded.

At some point, a company may decide to document the existing enterprise architecture, then make changes to enforce standards or comply with regulatory requirements. Because this is very expensive to do, the usual approach is to work on it one project at a time – each project team adds to the enterprise architecture the part they are using for their project.

An enterprise is typically a whole company or a whole division of a company. For example, Safeway IT is an enterprise, whereas Highmark Blue Cross might have several different enterprises – perhaps company internal systems, insurance systems, or

marketing systems. One of the things you have to decide for an enterprise architecture is “what is the enterprise I am describing?”


Many companies find it is not worth the time and money to do a project to identify every hardware system, every piece of software, and all of the relationships. Rather, this kind of information is collected when needed. Often a particular project team will find that they have to identify and describe some part of the enterprise architecture in order to do their project. It is rare for a company to have any one person or group of people who actually know the whole enterprise architecture.

The diagram shows a very simplistic enterprise architecture. This is using the UML (Unified Modeling Language) notation. The cubes represent computers. The rectangles represent components or processes on the computers. The lines show which computers can communicate with each other. Things to notice: the client records are on a different computer from the web server; the agency client cannot directly access the client records; there is no database cache on any computer; the company uses apache and JDB for the web server and nothing else. The last bit implies that the web server is running UNIX or Linux, but unless that is noted on the diagram, you do not know that for sure. That would be good information to add to the diagram – the kind of operating system each computer is running.

UML – cube represents a computer, device or execution environment (all these cubes are computers)

- rectangle on this diagram represents a component, artifact, or class (client record is a class, apache, jdb, and firefox are artifacts, agent interface is a component)

Slide 5




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

Copyright © Wyzzyk, Inc. 2004
Version 5.0 Enterprise Architecture - 5

Other information might include protocols or interfaces supported by the system, whether it is behind the firewall or in the DMZ, can it be accessed outside the company, any restrictions on its use, average load on the system, performance of the system, level of fault tolerance, backup schedule, and any mirroring or caching of the data on the system.


Slide 6



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

Copyright © Wyzzyk, Inc. 2004
Version 5.0 Enterprise Architecture - 6



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

Copyright © Wyzask, Inc. 2004
Version 5.0

Enterprise Architecture - 7

Use cases are text descriptions of process from the point of view of the user. An example of an enterprise level use case would be this:

1. The customer service representative receives an order for products.
2. The customer service representative verifies that the order is complete and correct.
3. The customer service representative gives the order to the warehouse manager.
4. The customer service representative gives the payment to the accounting clerk.
5. The warehouse manager creates a pick list from the order.
6. The warehouse clerk uses the pick list to collect the products for the order into a bin.
7. The warehouse clerk gives the bin and the order to the shipping clerk.
8. The accounting clerk deposits the check.
9. The shipping clerk creates a mailing label from the order.
10. The shipping clerk puts the order and the items from the bin into a box.
11. The shipping clerk puts the mailing label on the outside of the box.
12. The shipping clerk contacts UPS to pick up the order.


The concern at this level is to show the basic responsibilities of each department and the communication between departments. If the process is using software, then you can

reference the use of the software in the business process use case. This will show which systems need to communicate in order to accomplish the business process.

Maintenance type use cases can include add a node, remove a node, take a node off-line for maintenance, bring a node online, backup the systems, release a new application, update an existing application, make an application end-of-life.

Constraints will be discussed later in the lesson. Examples of constraints might be: data cannot be cached; all customer information must be encrypted when stored and routed between systems; system up-time must be 90% for all systems

Slide 8




Wyzzk, Inc. 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

Copyright © Wyzzk, Inc. 2004
Version 5.0 Enterprise Architecture - 8

Slide 9




Break

➤ As a class, review the given Enterprise Architecture.

Copyright © Wyzzyk, Inc. 2004
Version 5.0 Enterprise Architecture - 9

Slide 10




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

Copyright © Wyzzyk, Inc. 2004
Version 5.0 Enterprise Architecture - 10

These are not mutually exclusive. Often you will find these patterns in combination with each other, and recurring through various levels of the architecture (enterprise, program/line-of-business, and project).



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


Copyright © Wyzzak, Inc. 2004
Version 5.0

Enterprise Architecture - 11

Notice that the coarse grained services could be complete use cases, while the fine grained services correspond more to functions.

This is really possible today for many companies. Web services technologies allow a company to integrate services from legacy systems (perhaps in CICS or Cobol) with services in newer technologies (C, C++, Java, .Net). A business application can use web services technology to access the different services by just knowing their interfaces.

A service is described by what it does and how to invoke (call) it.



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

Copyright © Wyzzak, Inc. 2004
Version 5.0

Enterprise Architecture - 12

With existing systems and applications, you may need to write interfaces and wrappers to access the services. After that point, then new applications just use the interfaces and ignore how they are implemented.

It is more likely that a company will implement SOA a bit at a time. A new project team will consider the functionality needed for their project, determine if the functionality already exists in the enterprise, create a web service with an interface for the functionality, the web service will make the necessary calls to existing systems to implement the functionality, the project team registers the service with Web services server, then the project team uses the interface to the functionality. It is more work for the project team than making direct calls to the necessary systems to implement the functionality, but once they have written the service, no one else has to do that work again.



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?

Copyright © Wyzask, Inc. 2004
Version 5.0

Enterprise Architecture - 13

What are the pros and cons of centralized vs decentralized?

centralized – pros: one point of maintenance, easy to update, everyone gets changes at the same time

cons: one point of failure, load too high on the system, server becomes a bottleneck

decentralized – pros: many points of failure, ability to load balance, relatively easy recovery from disaster

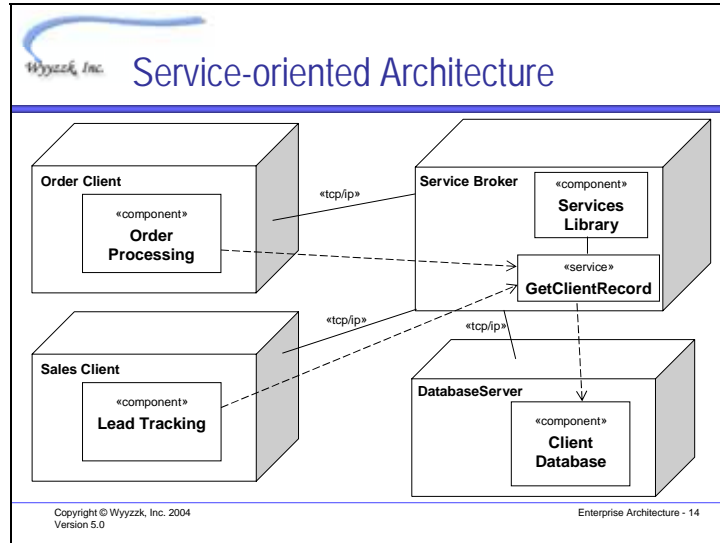
cons: need to set up mirroring to keep all servers synchronized, updates more complex, many points of maintenance, possibility of multiple versions in use at the same time


The service broker is software that has these functions : register a service with the broker, remove a service from the broker, find a service that the broker manages.

To develop new services, you need development and test areas that will not interfere with the running of the Enterprise. In addition, you need a roll-out plan for the new services – what QA must happen before release? How will the broker (s) be updated? How will the

Enterprise be notified that the new service is available? Who decides that the service is appropriate? Who decides that the service is ready for release?

Slide 14





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

Copyright © Wyzask, Inc. 2004
Version 5.0

Enterprise Architecture - 15


In event driven architecture, we consider what kinds of events or triggers we need our company to respond to. These may be found at the level of the corporation, division, or a specific application.

A piece of software called an agent watches for the event to happen. This is a sensor agent, because its job is to sense that the event has happened. Each event has its own agent watching for it. The sensor agent does not handle the event, but sends notifications to people or other software to handle the event. The sensor agent exists just to let someone or something know that the event has happened (or possibly did not happen in a certain time period). For example, the sensor agent could send a notification to an event broker, which collects many events before determining the appropriate action. The sensor agent could also send a notification to a processor agent which then determines the correct thing to receive the event. The processor agent forwards a message to a responder agent (person, device, software) which actually does the work in response to the event.

A sensor agent can be programmed to expect certain behavior, and notify the processor agent when that behavior does not occur. Another kind of sensor watches for something to happen and notifies the processor when that something does happen.

Also consider that a processor agent may be able to change the behavior of a sensor agent by sending it commands (for example, a clock sensor that sends a message once an hour may be changed to send a message once a minute).

Slide 16




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

Copyright © Wyzzak, Inc. 2004
Version 5.0

Enterprise Architecture - 16

It is more likely that a company will implement event-driven architecture a bit at a time. A new project team will consider the events that their application needs to handle, then create an event manager to handle each event. Notice that the code to handle the event could include calls to services. Event-driven and SOA can both be part of the same Enterprise Architecture.



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?

Copyright © Wyzzk, Inc. 2004
Version 5.0

Enterprise Architecture - 17

What are the pros and cons of centralized vs decentralized?

centralized – pros: one point of maintenance, easy to update, everyone gets changes at the same time

cons: one point of failure, load too high on the system, server becomes a bottleneck

decentralized – pros: many points of failure, ability to load balance, relatively easy recovery from disaster

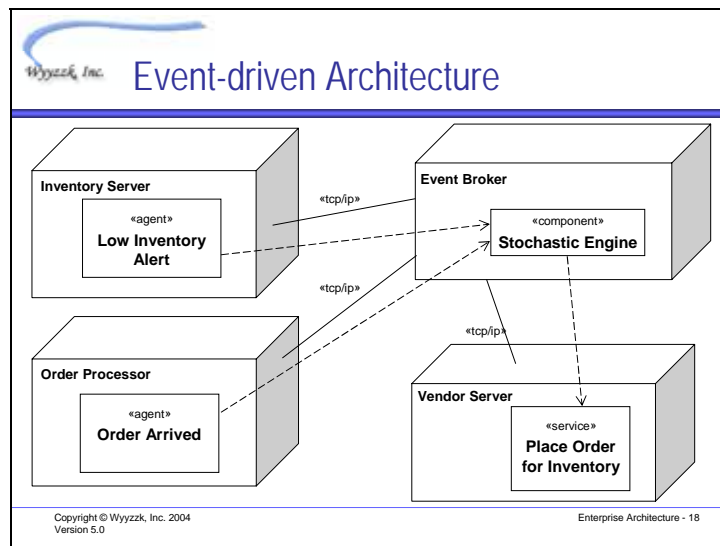
cons: need to set up mirroring to keep all servers synchronized, updates more complex, many points of maintenance, possibility of multiple versions in use at the same time

Notice that the event broker is notified when an event happens. Then logic in the event broker reviews the current events and makes a decision about the action to take. The action to take can include calling one or more event managers. The event broker can become quite complex over time, possibly requiring fuzzy logic, stochastic models, or artificial intelligence to determine the best course of action.

You could have another scheme where the centralized system is a collection of processor agents which get input streams from sensor agents and send output streams to responder agents.

To develop new events managers, you need development and test areas that will not interfere with the running of the Enterprise. In addition, you need a roll-out plan for the new event managers – what QA must happen before release? How will the server (s) be updated? How will the Enterprise be notified that the new event manager is available? Who decides that the event manager is appropriate? Who decides that the event manager is ready for release? How will the event manager be integrated with the event broker? What other events or event managers need to be involved when handling this event?

Slide 18



Slide 19

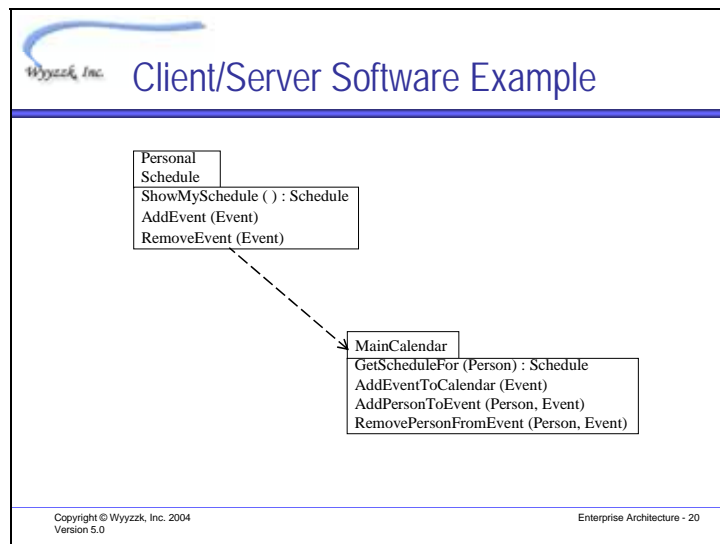
Wyzzk, Inc. Client/Server

- 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

Copyright © Wyzzk, Inc. 2004
Version 5.0

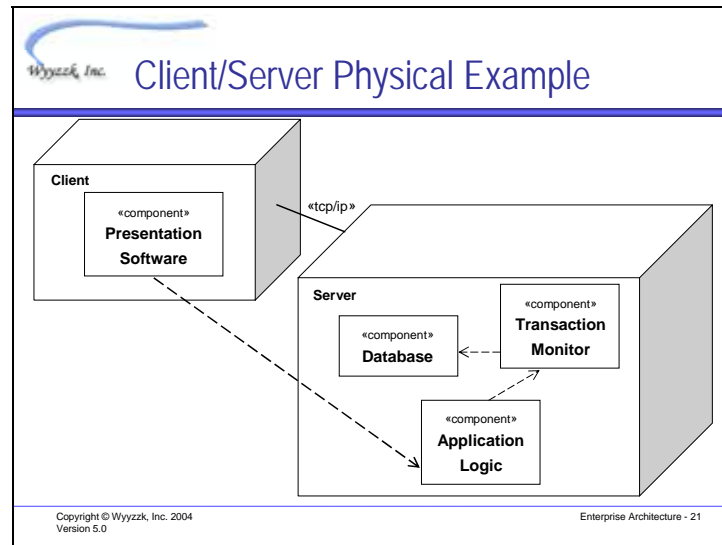
Enterprise Architecture - 19

Slide 20



Tabbed folders represent subsystems or packages in UML. In this case, the tabbed folders are subsystems.

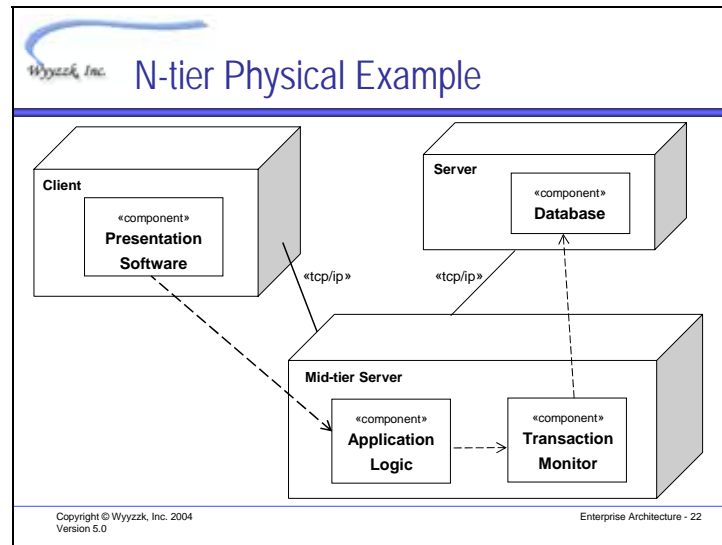
Slide 21



Here we are showing a physical client/server architecture. One computer is the client, the other is the server. In this case they are using tcp/ip as the communication mechanism.

Notice that the software architecture is multi-tier. Presentation software communicates with application logic which communicates with transaction monitor which communicates with database.

Slide 22




3 computers, so a 3 tier physical architecture. Notice that the software architecture has not changed from the previous slide.

Slide 23

-
- The slide, titled "Client/Server" by Wyzzk, Inc., lists the following points:
- 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
- The footer includes "Copyright © Wyzzk, Inc. 2004 Version 5.0" and "Enterprise Architecture - 23".

Slide 24




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

Copyright © Wyzack, Inc. 2004
Version 5.0 Enterprise Architecture - 24

Slide 25




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

Copyright © Wyzack, Inc. 2004
Version 5.0 Enterprise Architecture - 25

3-tiers is a typical starting place, though often one or more of the tiers will be further subdivided.

Slide 26

 **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

Copyright © Wyzzyk, Inc. 2004
Version 5.0 Enterprise Architecture - 26

Slide 27

 **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

Copyright © Wyzzyk, Inc. 2004
Version 5.0 Enterprise Architecture - 27

Slide 28

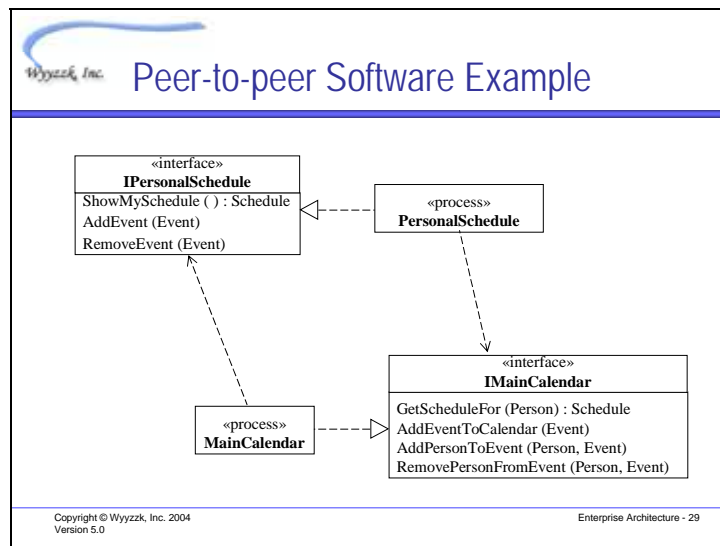
Wyzzyk, Inc. Peer-to-peer

- 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

Copyright © Wyzzyk, Inc. 2004 Version 5.0 Enterprise Architecture - 28

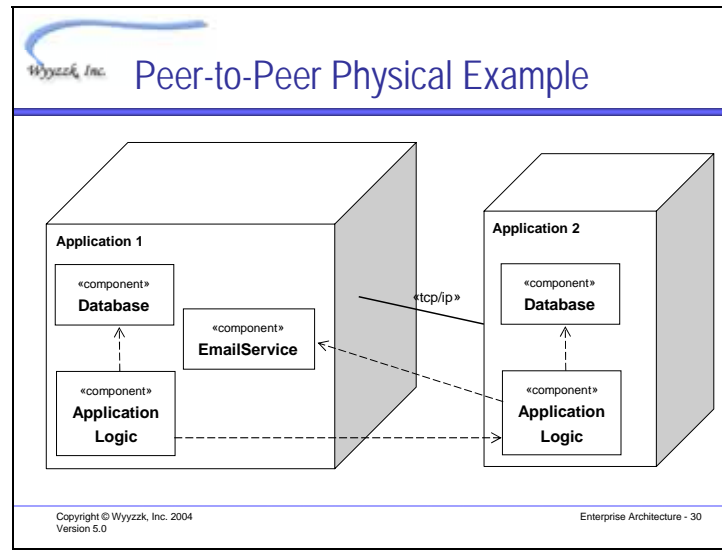
Peer-to-peer is essentially two-way client-server.

Slide 29



Tabbed folders represent subsystems or packages in UML. In this case, the tabbed folders are subsystems.

Slide 30




Here we are showing a physical peer-to-peer. Communication between the computers is 2-way. Notice that the software architecture is not peer-to-peer.

Slide 31

-
- The slide, titled "Peer-to-Peer", lists three characteristics of peer-to-peer systems:
- Community/Personal networking applications
 - Wireless networks
 - The peers are typically distributed across a network from each other
- The slide is from Wyzzak, Inc. and is part of Enterprise Architecture - 31, Version 5.0.

Think of blogging. You and I can each have a blog on our servers. When you read my blog, you are the client and my blog is the server. When I read your blog, I am the client and your blog is the server. Our systems are peer-to-peer. File-sharing applications are another example of peer-to-peer.

Slide 32




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

Copyright © Wyzzyk, Inc. 2004
Version 5.0 Enterprise Architecture - 32

Slide 33

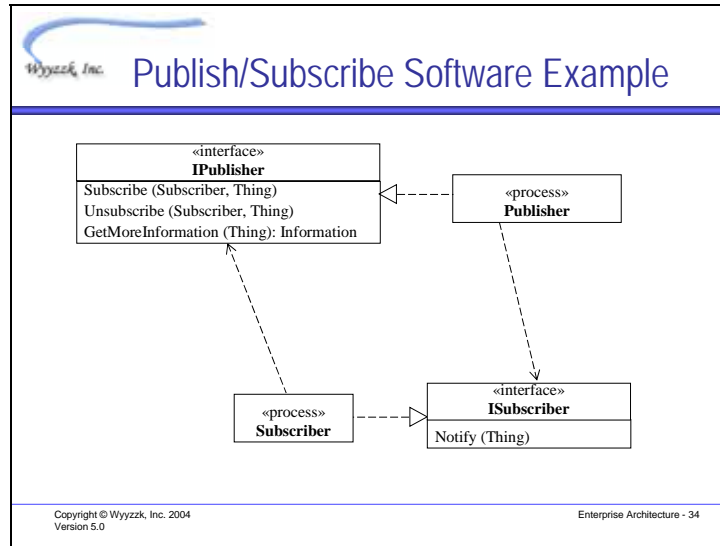


Publish/Subscribe

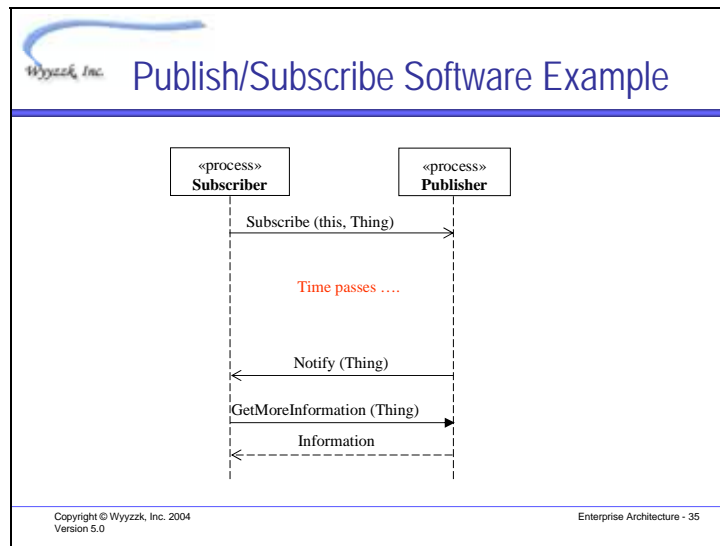
- One system is a publisher of information; any number of systems can subscribe to the information
- Publisher/Subscriber
 - Publisher – provides a means for other systems to subscribe to something of interest, notifies subscribers when that something of interest occurs
 - Subscriber – is interested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

Copyright © Wyzzyk, Inc. 2004
Version 5.0 Enterprise Architecture - 33


Slide 34



Slide 35



Slide 36




Publish/Subscribe

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

Copyright © Wyzask, Inc. 2004
Version 5.0 Enterprise Architecture - 36

Publish/subscribe is used when you have some kind of information in a centralized location and a number of other systems need to know that information.

Slide 37



Strengths and Weaknesses of Publish/Subscribe Architecture

<ul style="list-style-type: none">➤ Strengths<ul style="list-style-type: none">▪ One point of maintenance for information▪ Consistency in accessing information▪ More efficient use of resources than polling schemes	<ul style="list-style-type: none">➤ Weaknesses<ul style="list-style-type: none">▪ 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
---	---

Copyright © Wyzask, Inc. 2004
Version 5.0 Enterprise Architecture - 37

In polling, the subscriber would periodically ask the publisher if there is anything interesting about some thing that the publisher is managing.

Slide 38

Wyzzk, Inc. **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

Copyright © Wyzzk, Inc. 2004
Version 5.0 Enterprise Architecture - 38

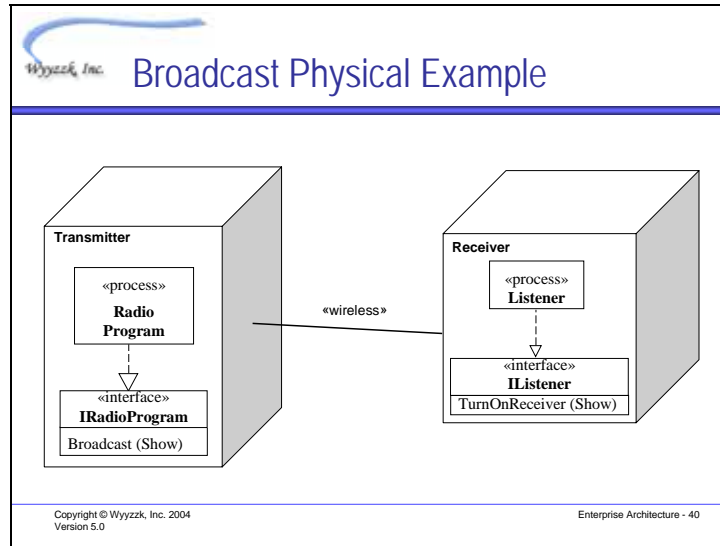
Slide 39

Wyzzk, Inc. **Broadcast Software Example**

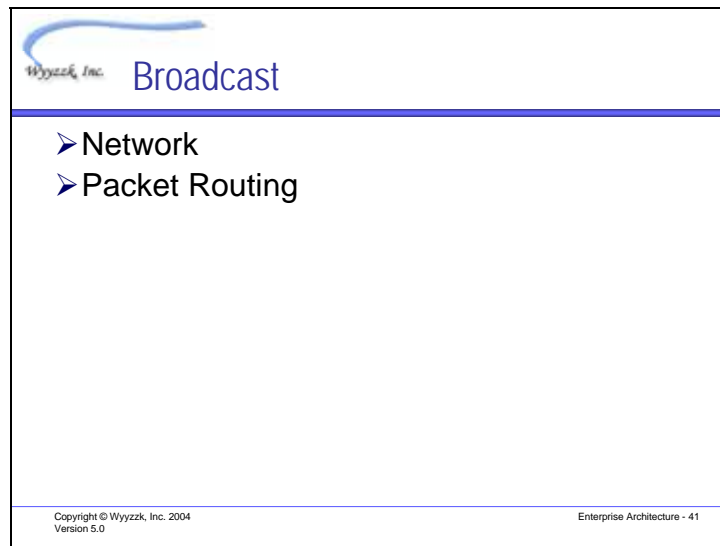
```
classDiagram
    class IListener {
        <<interface>>
        TurnOnReceiver (Show)
    }
    class Listener {
        <<process>>
    }
    class IRadioProgram {
        <<interface>>
        Broadcast (Show)
    }
    class RadioProgram {
        <<process>>
    }
    IListener <|.. Listener
    IRadioProgram <|.. RadioProgram
```

Copyright © Wyzzk, Inc. 2004
Version 5.0 Enterprise Architecture - 39

Slide 40



Slide 41

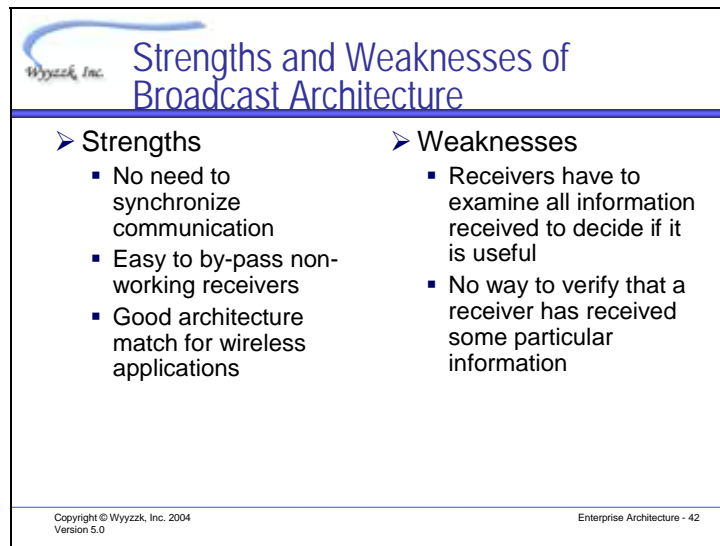



If your machine is looking for another machine with a specific IP address, your machine will send an ARP command to the local area network. This is a broadcast command to all machines on the network asking “who has this IP address?” If the IP address does not exist on your LAN, then your router will do a similar query of the internet.

A similar thing happens when you configure a TCP/IP printer on your Windows system. Windows broadcasts a query “Any printers out there, tell me who you are.” That populates the list of printers you see in the setup screens.

At a low level of message passing, your message is divided into packets which are sent across the net. The specific route for each packet is determined by querying routers (could be a broadcast request) for who is available to pass the message towards its destination.

Slide 42

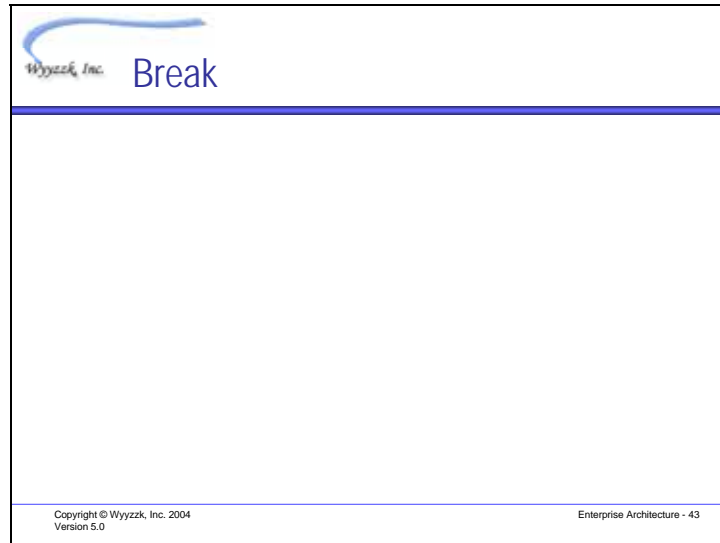


 **Strengths and Weaknesses of Broadcast Architecture**

➤ Strengths	➤ Weaknesses
<ul style="list-style-type: none">▪ No need to synchronize communication▪ Easy to by-pass non-working receivers▪ Good architecture match for wireless applications	<ul style="list-style-type: none">▪ 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

Copyright © Wyzzik, Inc. 2004
Version 5.0 Enterprise Architecture - 42

Slide 43

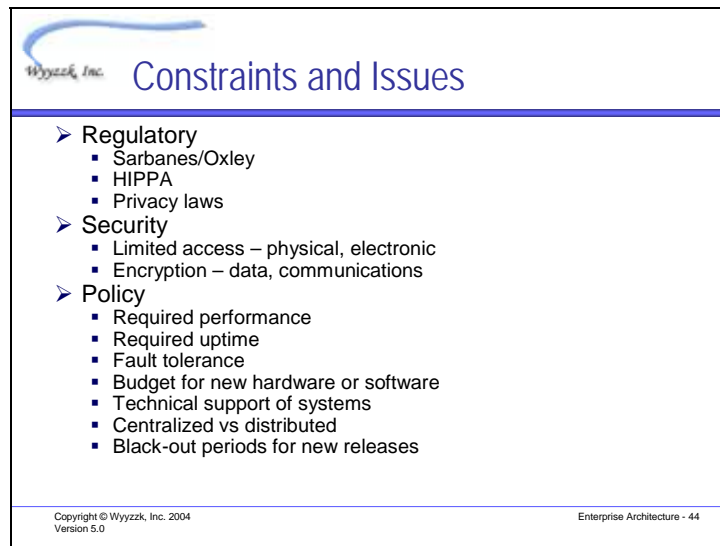


Wyzask, Inc. Break

Copyright © Wyzask, Inc. 2004
Version 5.0

Enterprise Architecture - 43

Slide 44



Wyzask, Inc. 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

Copyright © Wyzask, Inc. 2004
Version 5.0

Enterprise Architecture - 44

Black-out period – for example Visa will not allow the release of most corporate software from end of September until end of December. There is too great a risk of problems during the holiday rush and they don't want to risk it (and neither do their merchants!)

Slide 45




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)

Copyright © Wyzzk, Inc. 2004
Version 5.0 Enterprise Architecture - 45

Slide 46




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

Copyright © Wyzzk, Inc. 2004
Version 5.0 Enterprise Architecture - 46

Slide 47




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

Copyright © Wyzask, Inc. 2004
Version 5.0 Enterprise Architecture - 47

Slide 48




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.

Copyright © Wyzask, Inc. 2004
Version 5.0 Enterprise Architecture - 48

Slide 49




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

Copyright © Wyzzk, Inc. 2004
Version 5.0 Enterprise Architecture - 49

Slide 50



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.

Copyright © Wyzzk, Inc. 2004
Version 5.0 Enterprise Architecture - 50