IBM Software Group

IBM WebSphere® Technical Conference
Featuring WebSphere Portal, WebSphere MQ, CICS® and Rational®

C03 Architecting e-business Applications: Part 1 Topologies, Configurations, Products

Geoff Sharman

WebSphere software

@business on demand software

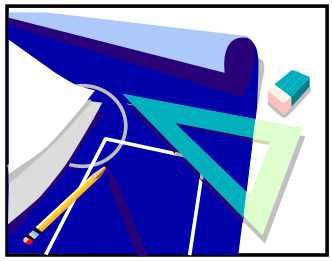
Agenda

Part I

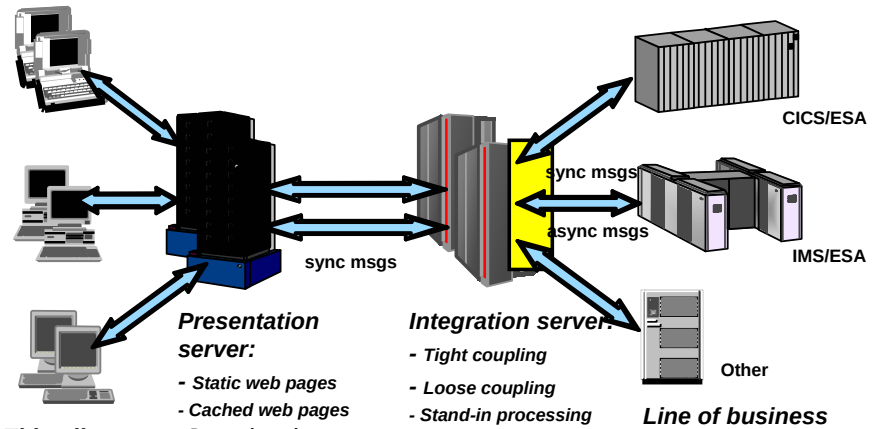
- The general e-business architecture pattern
- How to optimise the pattern

Part II

- Performance principles
- Capacity planning example



e-business Application Architecture



Thin clients

Presentation server:

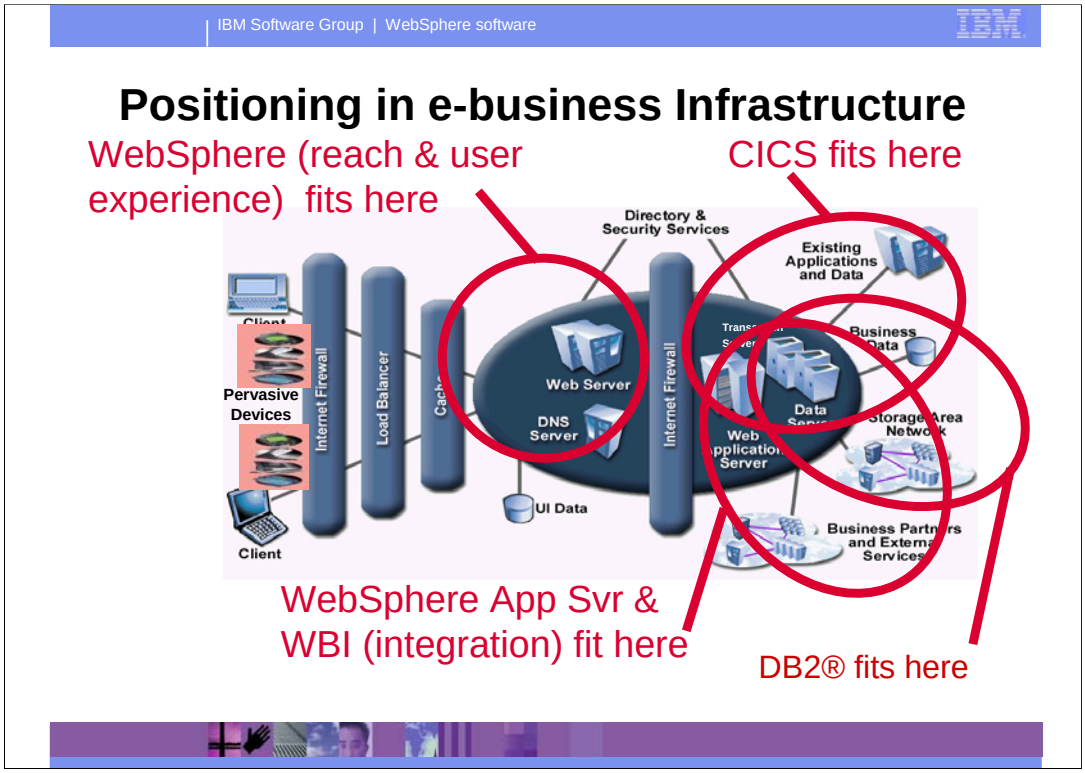
- *Static web pages*
- *Cached web pages*
- *Dynamic web pages*
- *Portal*

Integration server:

- *Tight coupling*
- *Loose coupling*
- *Stand-in processing*
- *Flow ctrl/compensation*
- *Connectors*

Line of business Transaction servers

Note: connectors are shown in yellow



We (IBM) understand the evolving ebusiness environment

We are optimizing our products in support of this environment

There are many parts to the infrastructure:
Attach and support of pervasive devices and clients

The support for Directory and Security Services

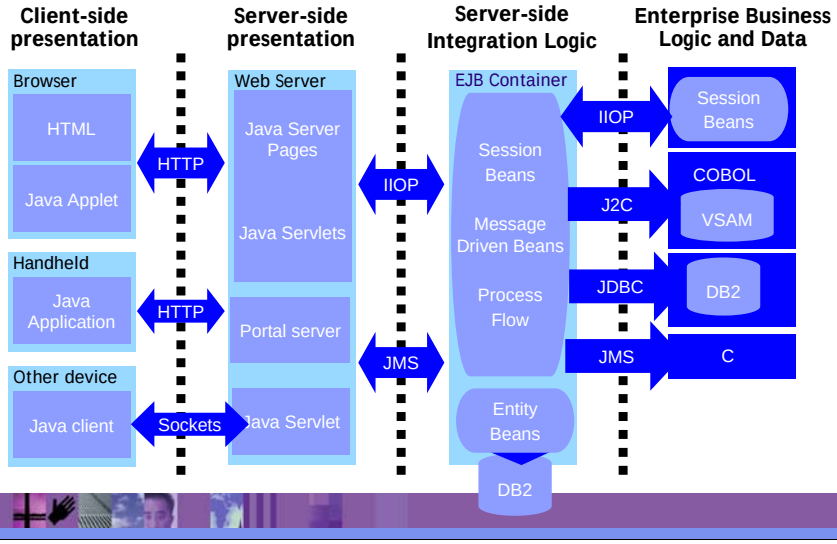
The use of Web Servers

The connection to existing systems and processes
But at the heart of this infrastructure are the:
Transaction Servers, Data Servers, Application servers. Like CICS, IMS, DB2 nad WebSphere

These are the products that are coping with the explosion of transactions caused by e-business.

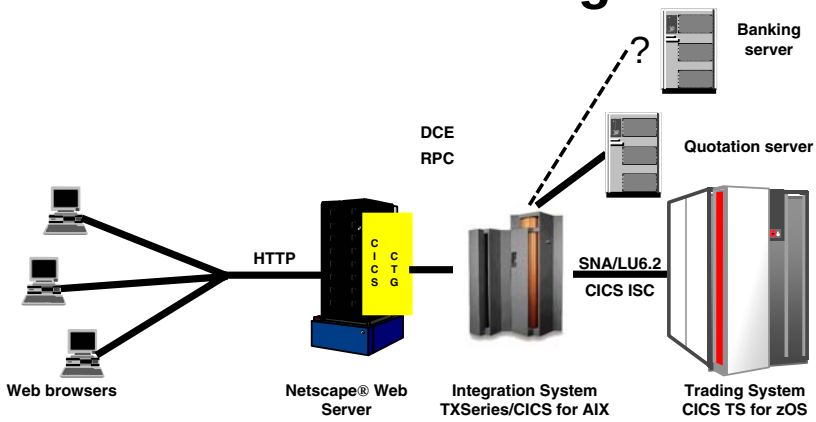
Software Architecture for e-business

An end-to-end architecture ... based on Java™ 2 Enterprise Edition



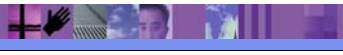
Securities Company #2 - Project B

Internet Stock Trading Service

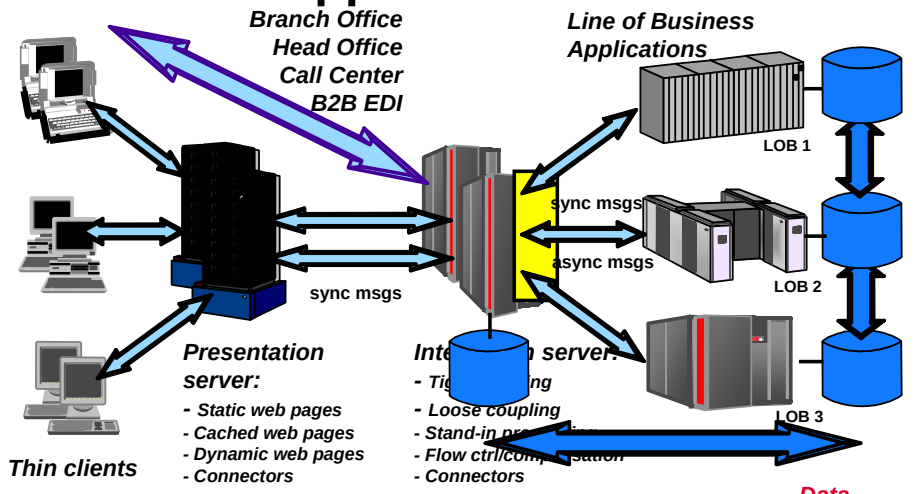


- Over 2 million Internet based self service customers
- >400 nodes running Netscape Web server + WebSphere
- >100 nodes running TXSeries®/CICS for AIX® Server
- ~15 million Internet transactions/day into backend trading system

Generalised Architecture Patterns



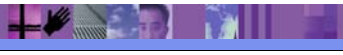
e-business Application Architecture



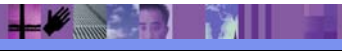
Note: connectors are shown in yellow

Forms of Integration

- Why is integration vital?
  - Because most enterprises have a collection of heterogeneous systems, for historical reasons
- What is Service Integration?
  - Decomposition of a realtime service request into sub-requests on different systems, and recomposition of sub-responses into an overall response
  - Usually synchronous or nearly-synchronous
  - May involve atomic transactions using the 2PC paradigm
- What is Data Integration?
  - Exchange of "common" data, eg. customer name/address, between different systems in order to maintain consistency
  - Usually asynchronous, delay time may be hours

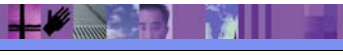


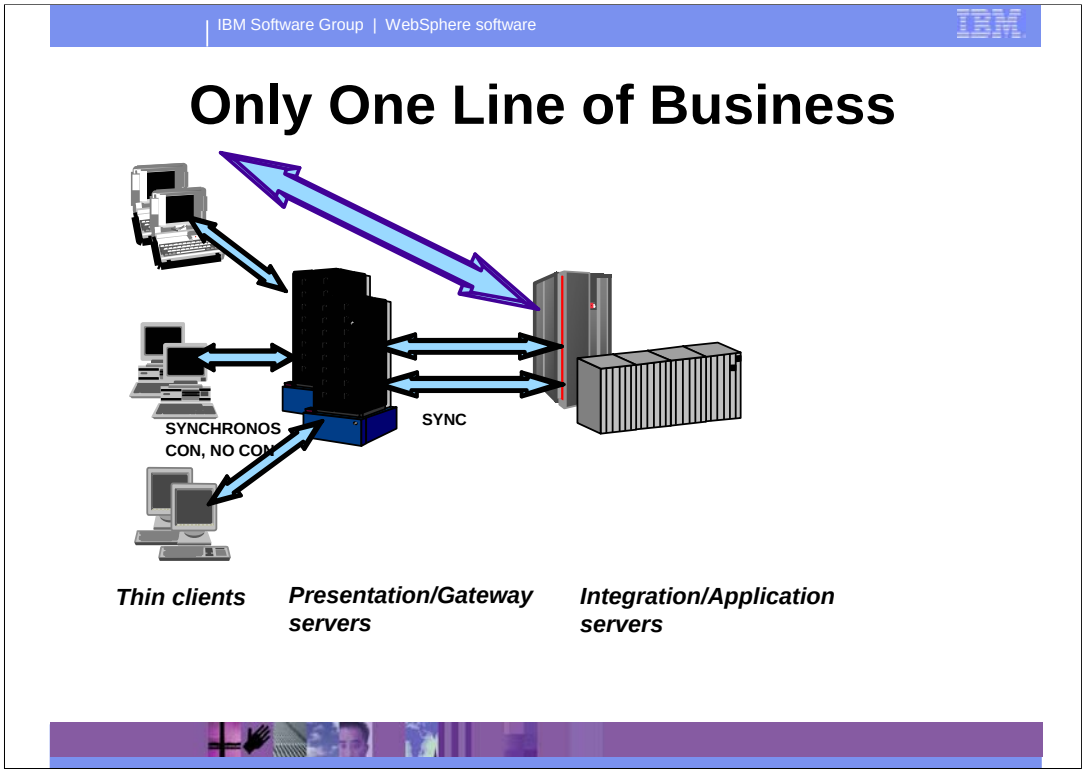
Service Integration Patterns



Service Integration: Special Cases

- When there is only one line of business server
  - integration logic and business logic may co-reside
  - backend application server used for both
- When there is only one service delivery channel
  - presentation and integration logic may co-reside
  - frontend web application server used for both
- Both offer opportunities for optimisation
  - less networking = more performance/availability

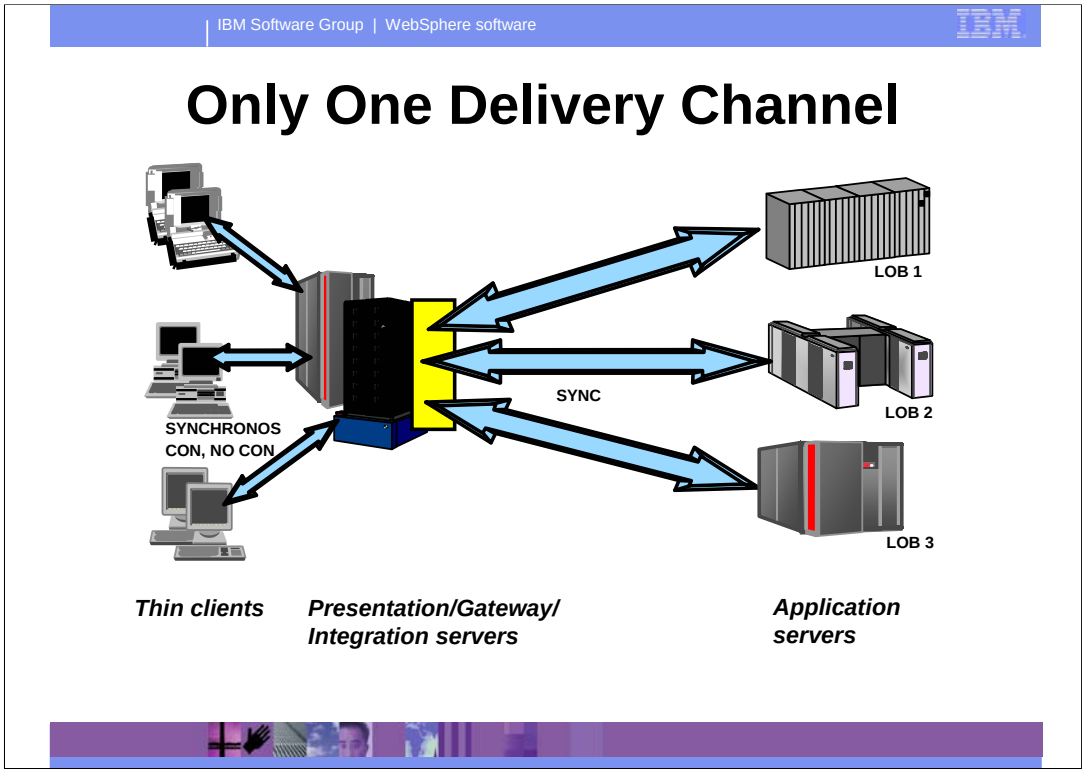




Integration logic and application logic both reside on the same backend application server - separate integration server isn't needed

One network link can be eliminated, giving:

- less comms cycles = better throughput
- less comms latency = better response time
- less comms failures = better availability



Presentation logic and integration logic can both reside on the same frontend web application server - separate integration server isn't needed

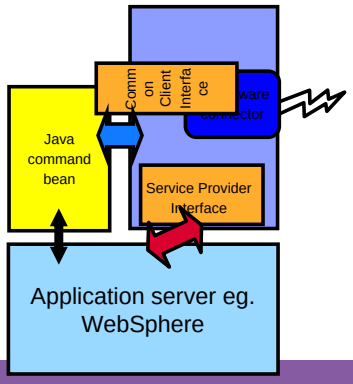
One network link can be eliminated, giving:

- less comms cycles = better throughput
- less comms latency = better response time
- less comms failures = better availability

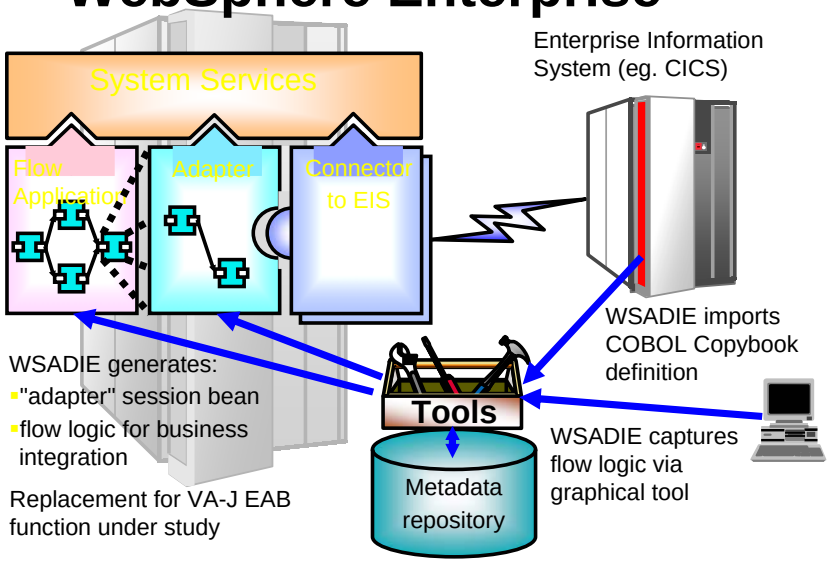
What do JCA Connectors do?

- ***Provide Common Client Interface for all back end servers***
  - standard call interface hides connection details from Java applications
  - each back end server still requires unique data format
- ***Enable Service Provider Interface for Application Server***
  - operate in "managed" or "unmanaged" environments
  - managed environment enables connection pooling, transactions & security
- ***Leverage tool technology:***
  - Connectivity to specific backend may be encapsulated in an "adapter" bean
  - WSAD/IE automates construction of interactivity/navigation & data format logic

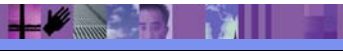
- A ***Connector*** is generic runtime code, such as a J2EE architected connector, that transforms one calling interface into another
- An ***Adapter*** is runtime code, ***possibly generated by a tool***, that converts one data format to another (e.g. converts a bean format into a CICS COMMAREA)
- Many solutions will use ***both*** connectors and adapters



JCA Tooling with WSADIE and WebSphere Enterprise



Data Integration Patterns



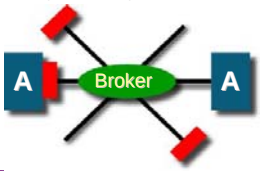
Forms of Data Integration

The Business Integration Evolution

1 *Application to Application*

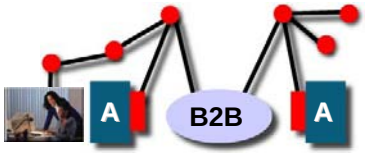
2 *Adapters for speed & simplification*

3 *Using a broker - routing between many to many*



4 *Business Process Management within an enterprise*

5 *Business Process Management within and across enterprises*



Company A *Company B*

Multi-server Messaging Pattern (i)

Servers exchange messages in *arbitrarily complex patterns*

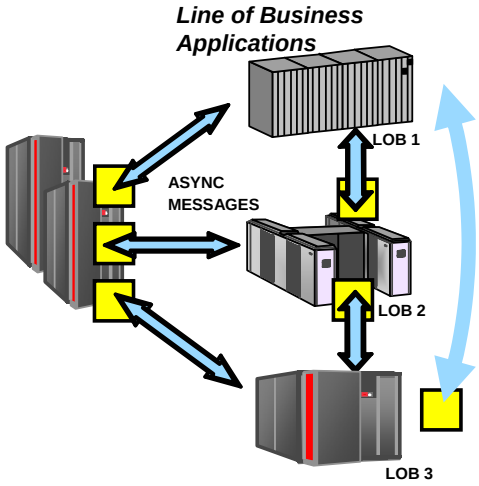
- for n servers, there are n(n-1)/2 possible pairs
- for heterogeous servers, each pair may use a different message format
- complexity grows rapidly with number of servers, eg.

n=4 gives 6 pairs

n=5 gives 10 pairs

n=6 gives 15 pairs

n=10 gives 45 pairs

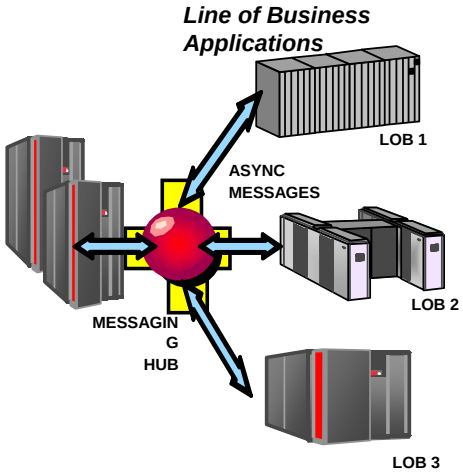


Geometric growth of complexity!

Multi-server Messaging Pattern (ii)

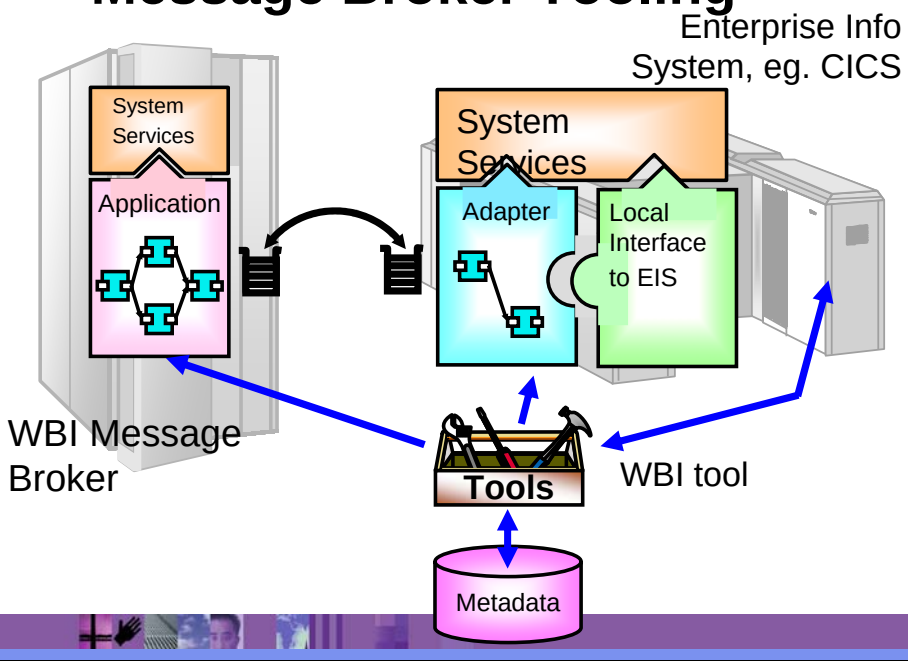
Servers exchange messages via a *messaging hub/broker*

- for n servers, there are n possible connections to hub
- for heterogeous servers, each connection may use a different message format
- complexity grows with number of servers, but hub looks after transformations (adapters)
- cost of adding (n+1)th server is adding n transformations



Linear growth of complexity!

Message Broker Tooling

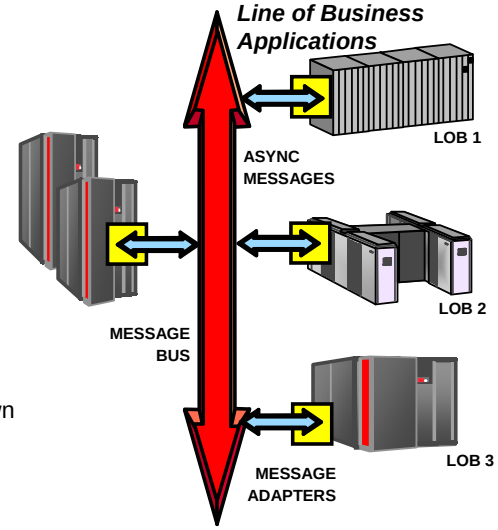


Multi-server Messaging Pattern (iii)

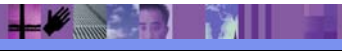
Servers exchange standard messages via a ***message bus*** *(usually consisting of 1 or more message brokers)*

- for n servers, there are n possible connections to bus
- all messages on bus use same standard format
- for heterogeous servers, each server transforms to standard message format, using adapter
- complexity independent of number of servers, each has own message adapter
- cost of adding (n+1)th server is cost of adding 1 adapter

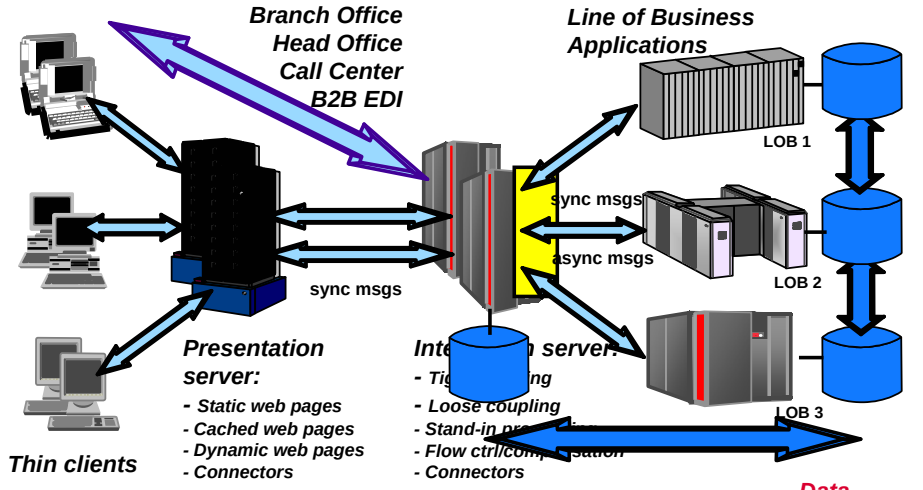
No growth of complexity!



Bringing it all Together

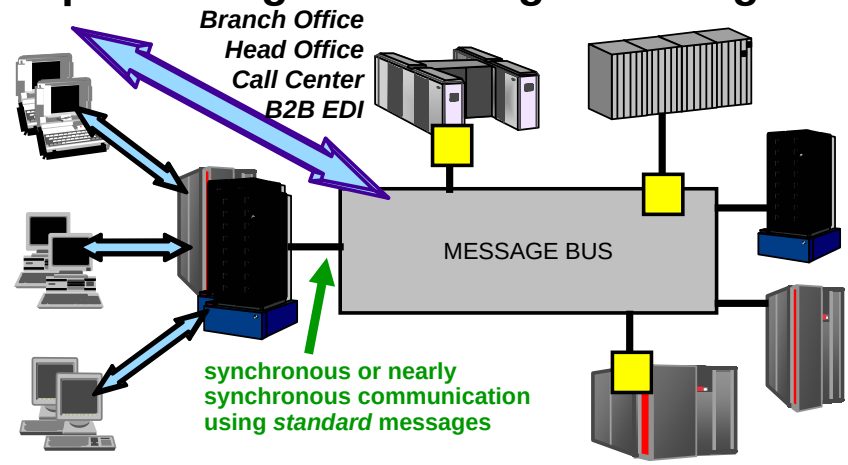


e-business Application Architecture



Note: connectors are shown in yellow

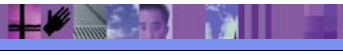
Enterprise Integration using a Message Bus



NB. may still use connectors and adapters (shown in yellow)

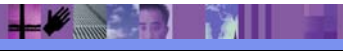
Isn't this like Web Services?

- "Web Services" is a set of standard architectures which:
  - define how systems interoperate using ***standard*** messages
  - are intended for interoperation ***between*** enterprises
  - may also be used for interoperation ***within*** enterprises, eg. between channel servers and business servers
- Message architecture based on SOAP (Simple Object Access Protocol) defines:
  - message envelope protocol for routing
  - message header protocols for:
    - security
    - transactions
  - message payload/body defined by agreement between communicating applications

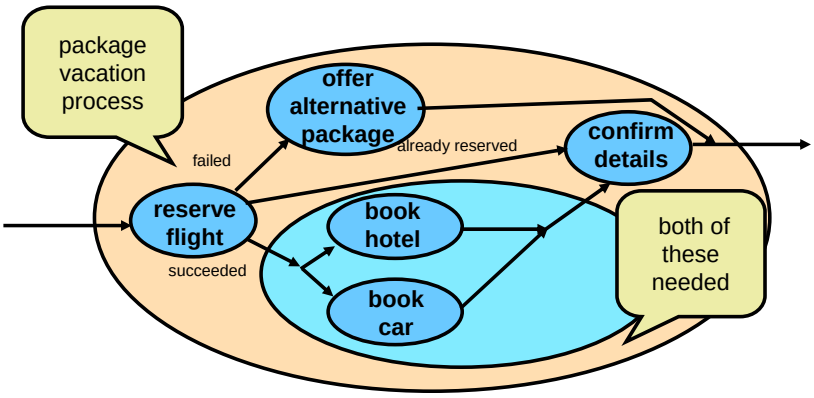


What is a Message Bus?

- A network of message routing servers ...
  - usually built from message brokers or application servers
- ... which supports service integration
  - makes all the backend pieces look like "one service"
  - may coordinate actions on line of business servers
  - masks unavailability/failure of a given service
- ... ***and*** data integration
  - asynchronous interchange of versioned data
- ... also enables definition of process flow
  - "scripting" of existing assets to implement new business functionality
  - includes sequential/parallel steps, compensating actions
  - may involve multiple interactions over an extended period



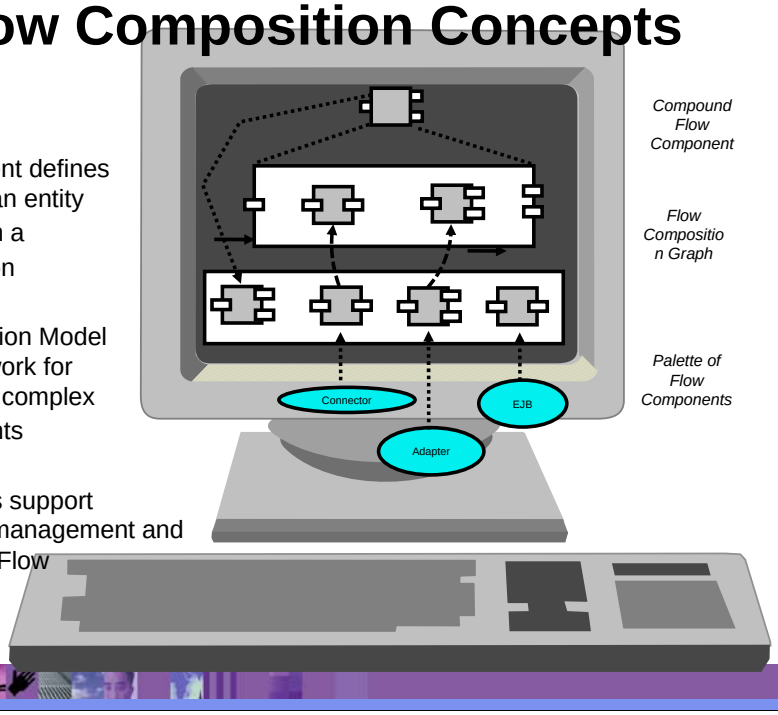
Example of Process Flow Logic



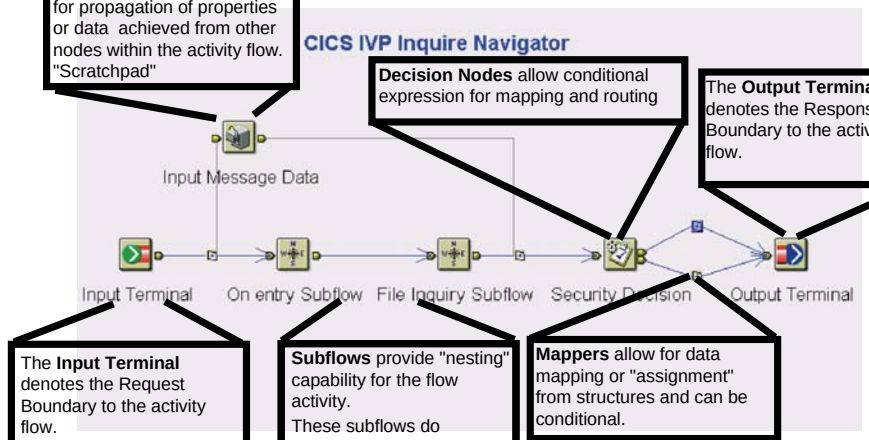
- *each integration step may use a different backend application*
- *composes sequential/parallel steps in process*
- *handles failure of any step; defines compensating actions*
- *represents stateful interactions in longlived business process*

Flow Composition Concepts

- Flow Component defines abstraction of an entity that can perform a business function
- Flow Composition Model defines framework for construction of complex Flow Components
- Common Tools support development, management and deployment of Flow Components



Example: A Customer Inquiry Flow



Data Context Node allows for propagation of properties or data achieved from other nodes within the activity flow. "Scratchpad"

Decision Nodes allow conditional expression for mapping and routing

The **Output Terminal** denotes the Response Boundary to the activity flow.

The **Input Terminal** denotes the Request Boundary to the activity flow.

Subflows provide "nesting" capability for the flow activity.
These subflows do "On Entry" housekeeping function and the actual Customer Inquiry.

Mappers allow for data mapping or "assignment" from structures and can be conditional.

Summarising the Patterns ...

- Presentation server(s) support channel(s)
  - typically a web server/web application server
  - may use portal technology
  - devices may be mobile phone, handheld, TV, etc.
- Message/service bus supports integration
  - enables integration of heterogeneous backend systems
  - supports both service integration *and* data integration
  - isolates end user from failure of any one service
  - determines end user response time and service availability
  - may use process flow to capture business process and drive many backend servers in parallel
- Enterprise servers support business applications
  - as they have always done - most are already in place