SOA versus
Traditional Architectures

The monolithic mainframe application architecture

- Separate, single-function applications, such as order-entry or billing
- Applications cannot share data or other resources
- Developers must create multiple instances of the same functionality (service).
- Proprietary (user) interfaces

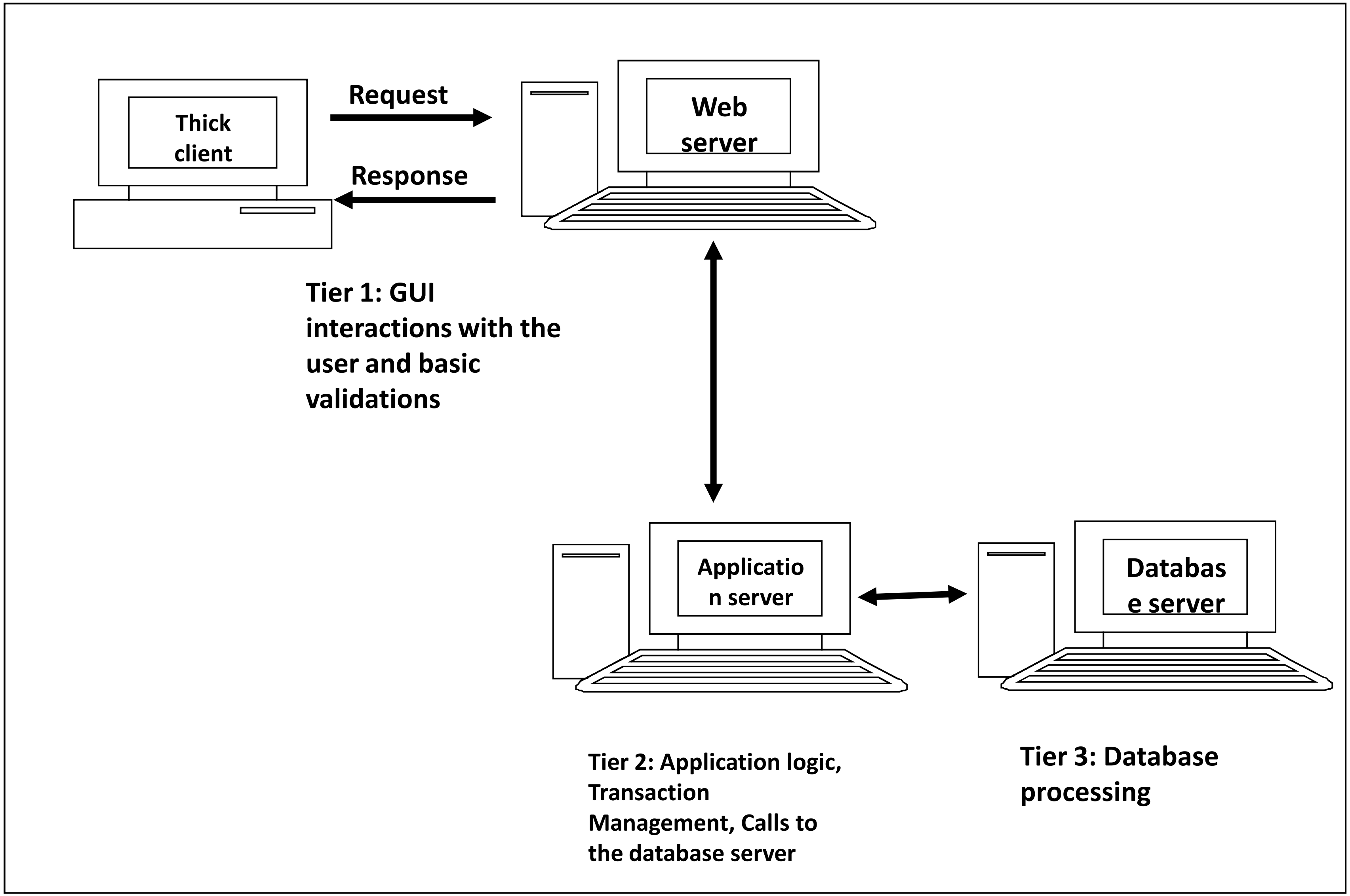
The distributed application architecture

- Integrated applications
- Applications can share resources
- A single instance of functionality (service) can be reused.
- Common user interfaces
- Bottom-up approach
- Real world scenario

Web based systems …

- ♣ Client-server model
- ♣ Client side technologies
- ♣ Server side technologies
- ♣ Web client, Web servers
- ♣ Application servers

Basic idea of Tiers

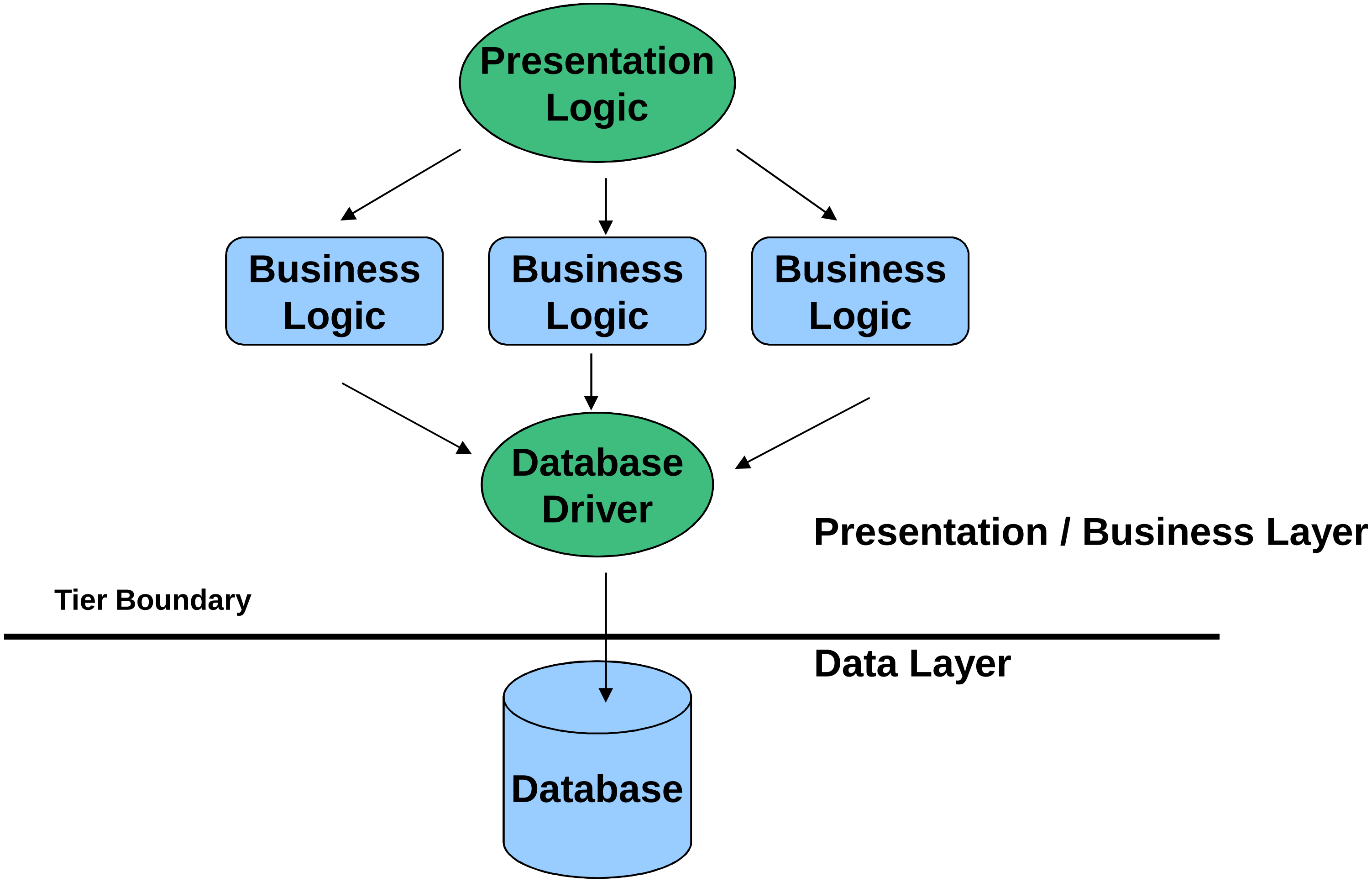


Tier 1: GUI interactions with the user and basic validations

Tier 2: Application logic, Transaction Management, Calls to the database server

Tier 3: Database processing

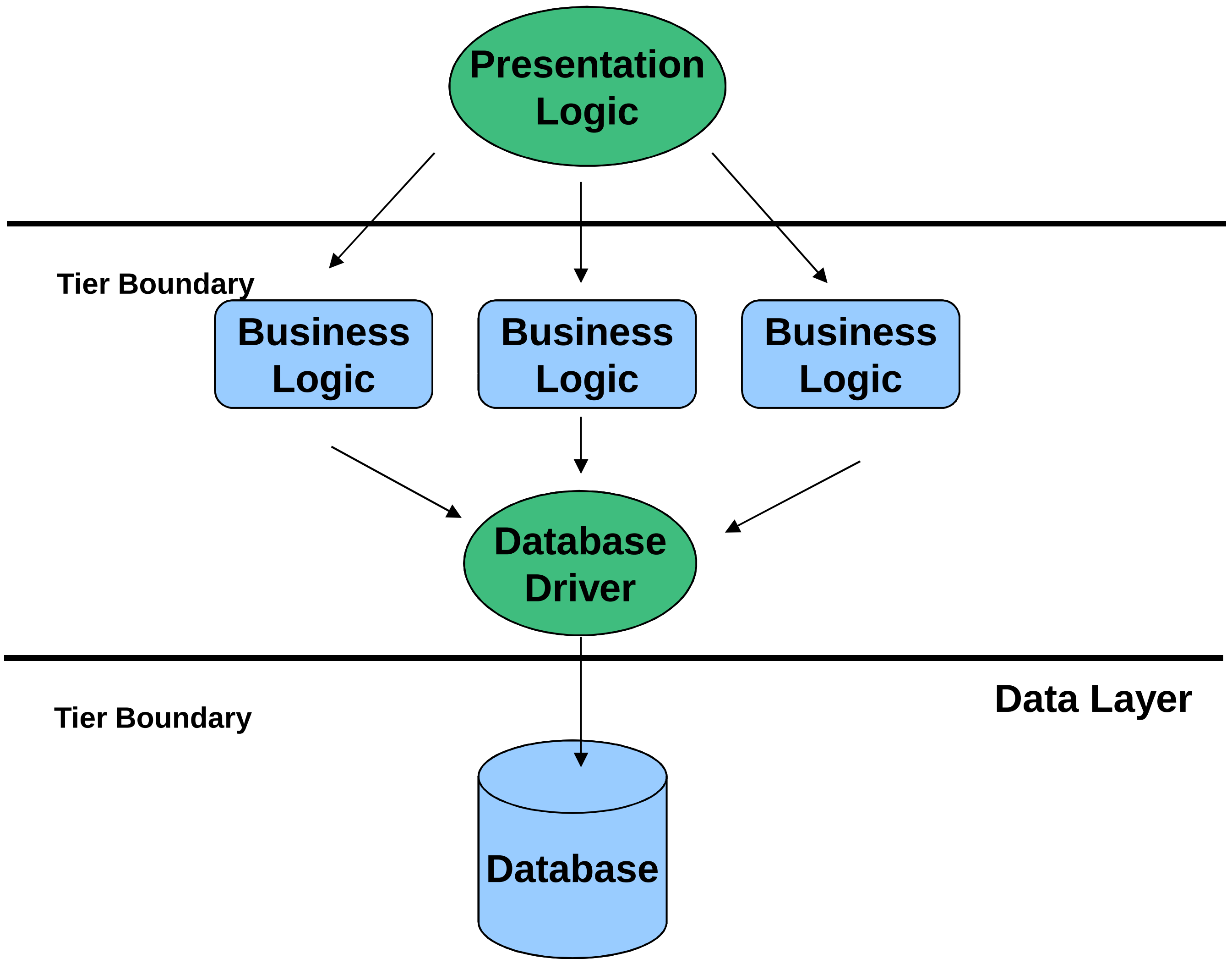
2-tier architecture



Two tier architecture

- **Deployment costs are high**
- **Database driver switching costs are high**
- **Business logic migration costs are high**
- **The client has to recompile if the BL is changed**
- **Network performance suffers**

N-Tier architecture



N-Tier architecture

- **Deployment costs are low**
- **Database switching costs are low**
- **Business migration costs are low**
- **Each tier can vary independently**
- **Communication performance suffers**
- **Maintenance costs are high**

Presentation tier technologies

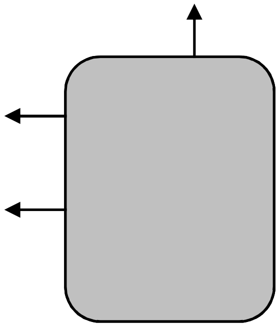
At client or server?	Property	Microsoft Technology	Sun Technology
Client	HTTP (Web) based	HTML browser (Internet Explorer)	HTML browser (Netscape Navigator)
		ActiveX Controls	Java Applets
	Non-HTTP based	COM clients	CORBA clients
	Communication Protocol between client and server	DCOM	RMI, IIOP
Server	For creating dynamic Web pages	ISAPI, ASP	NSAPI, Servlets, JSP
	Other pages	HTML, XML	HTML, XML

Business tier technologies

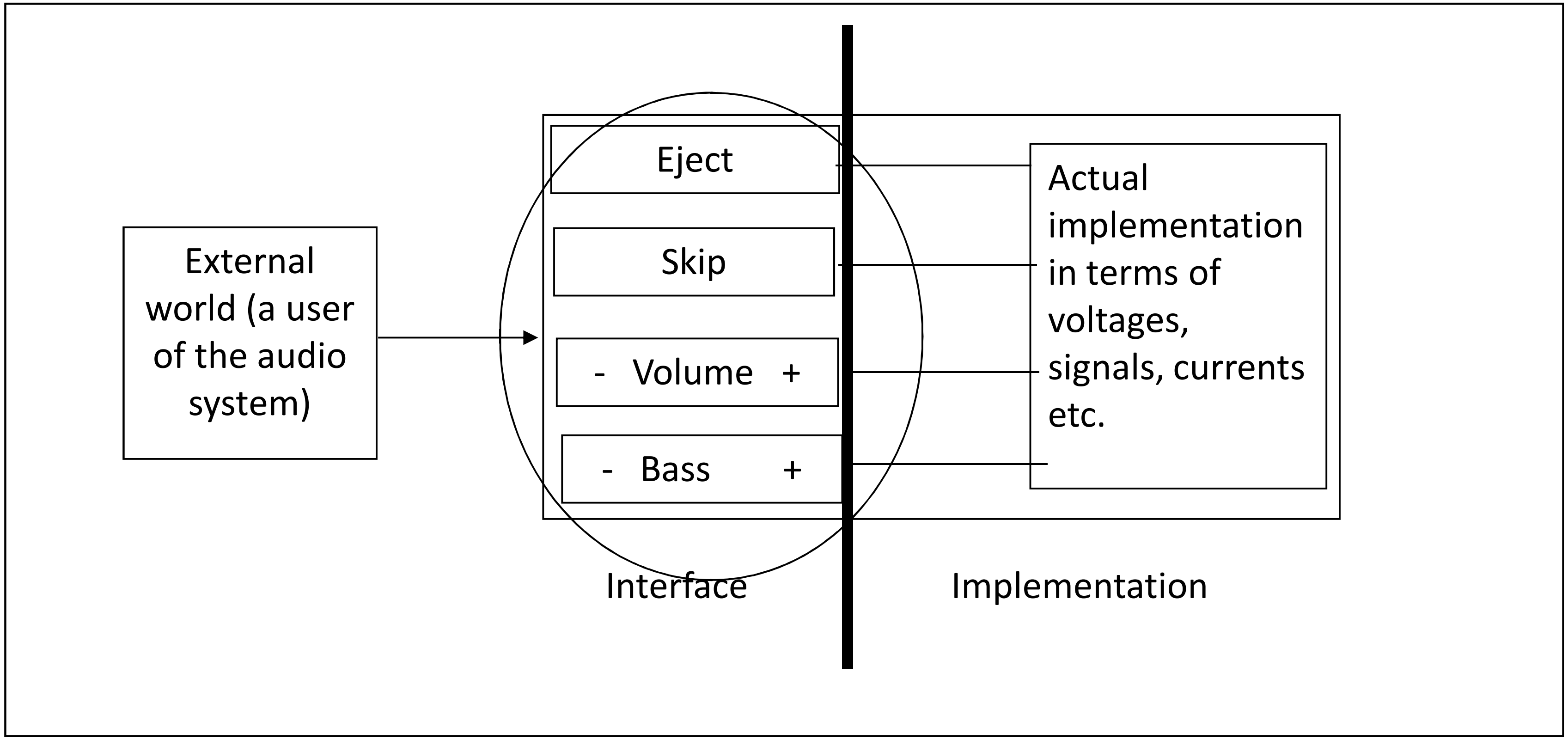
Purpose	Microsoft Technology	Sun Technology
Transaction handing, Business Objects	COM, MTS	EJB (Session Beans)
Queuing and Messaging	MSMQ	IBM’s MQSeries, Java Messaging Service (JMS)
Database access	ADO, OLE, ODBC	JDBC, J/SQL (via Entity Beans)

Component World …

- ♣ Justification for component
- ♣ Interface
- ♣ Implementation
- ♣ Reusability
- ♣ standards



Interface and Implementation


External world (a user of the audio system)
Eject
Skip
- Volume +
- Bass +
Actual implementation in terms of voltages, signals, currents etc.
Interface
Implementation

Technologies for implementing components

- RMI / EJB
- CORBA
- COM, DCOM, COM+
- Limitations
- Web services (XML based standards)

Limitations of Components

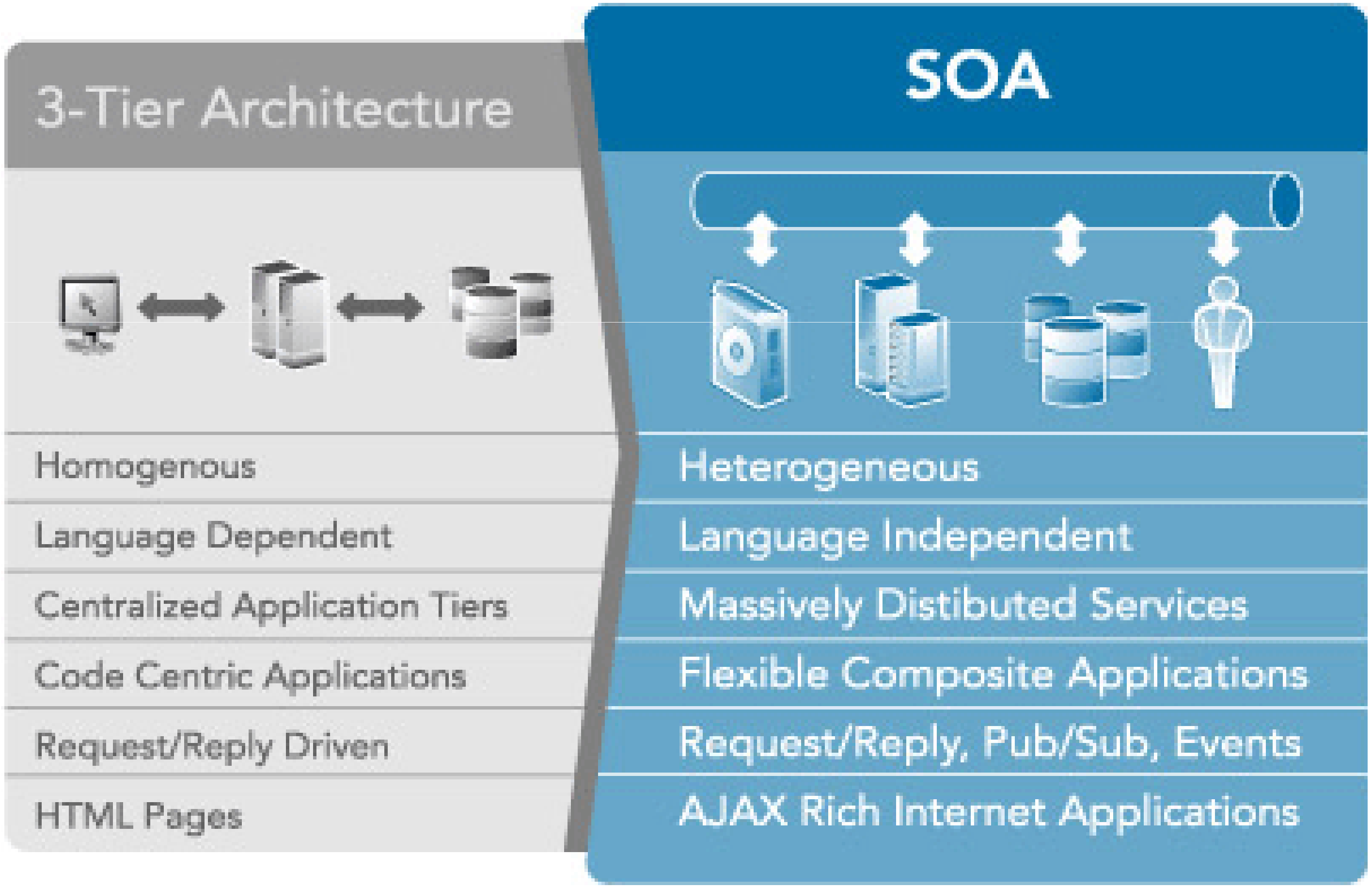
- Tightly coupled
- Cross language/ platform issues
- Interoperability issues
- Maintenance and management
- Security issues

Why SOA?

- Heterogeneous cross-platform
- Reusability at the macro (service) level rather than micro(object) level
- Interconnection to - and usage of - existing IT (legacy) assets
- Granularity, modularity, composability, componentization
- Compliance with industry standards

SOA is an evolutionary step

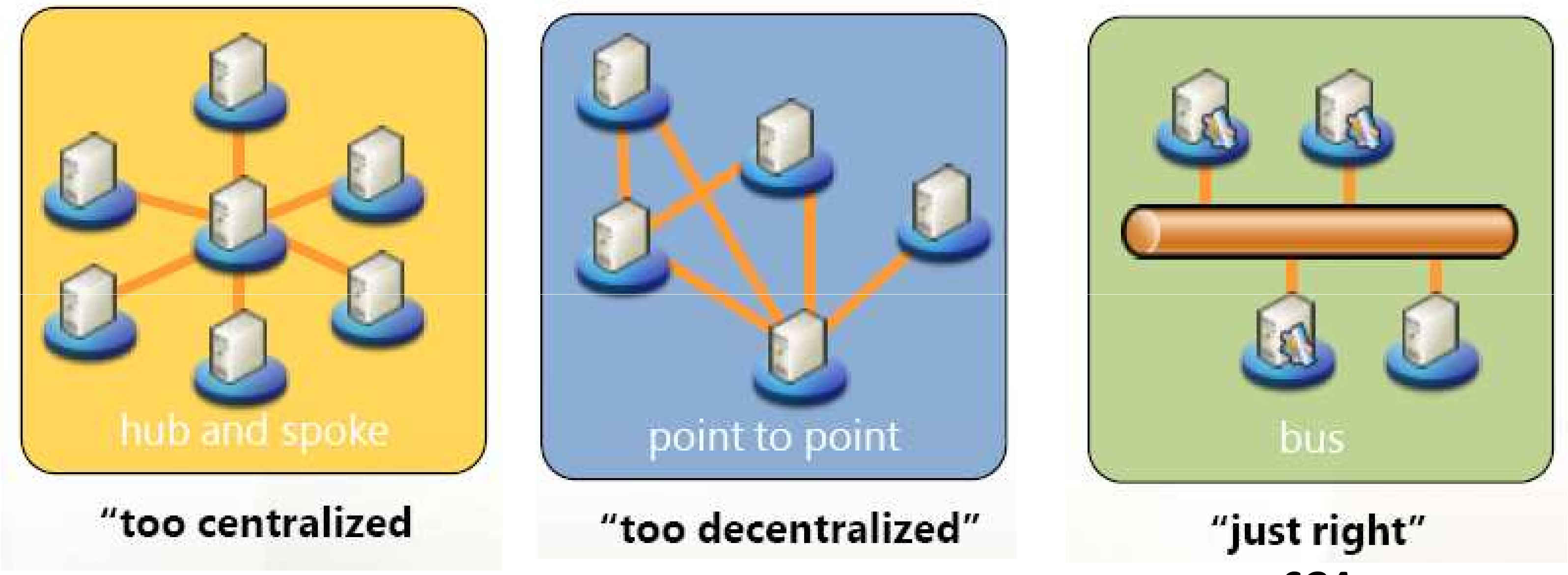
- for architecture



3-Tier Architecture	SOA
Homogenous	Heterogeneous
Language Dependent	Language Independent
Centralized Application Tiers	Massively Distibuted Services
Code Centric Applications	Flexible Composite Applications
Request/Reply Driven	Request/Reply, Pub/Sub, Events
HTML Pages	AJAX Rich Internet Applications

SOA is an evolutionary step

in distributed communications



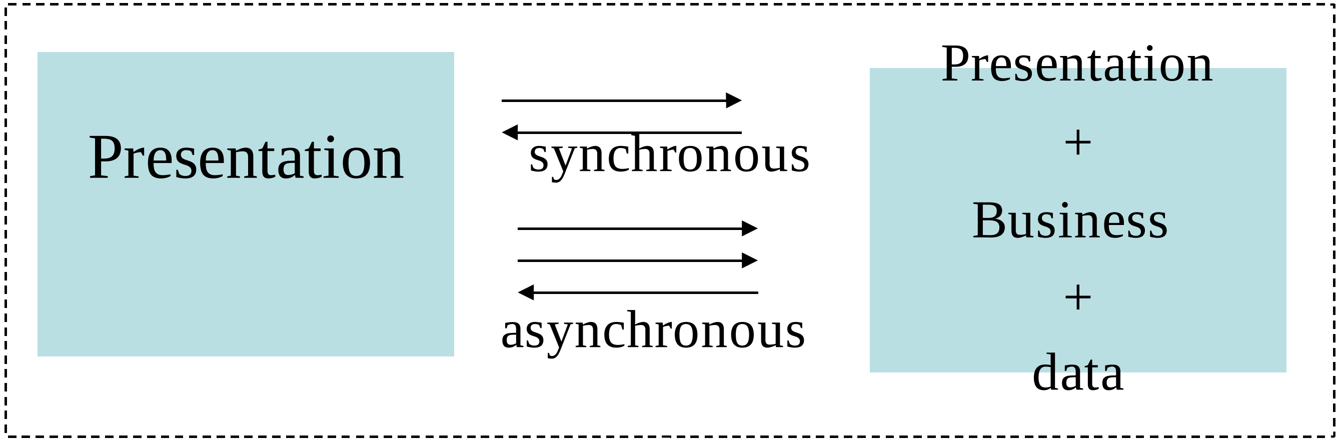
EAI

Project-ware

SOA

source:Sam Gentile

Single-tier client-server Architecture



Presentation

synchronous

asynchronous

Presentation

+

Business

+

data

Thin (dumb) client

No application Logic

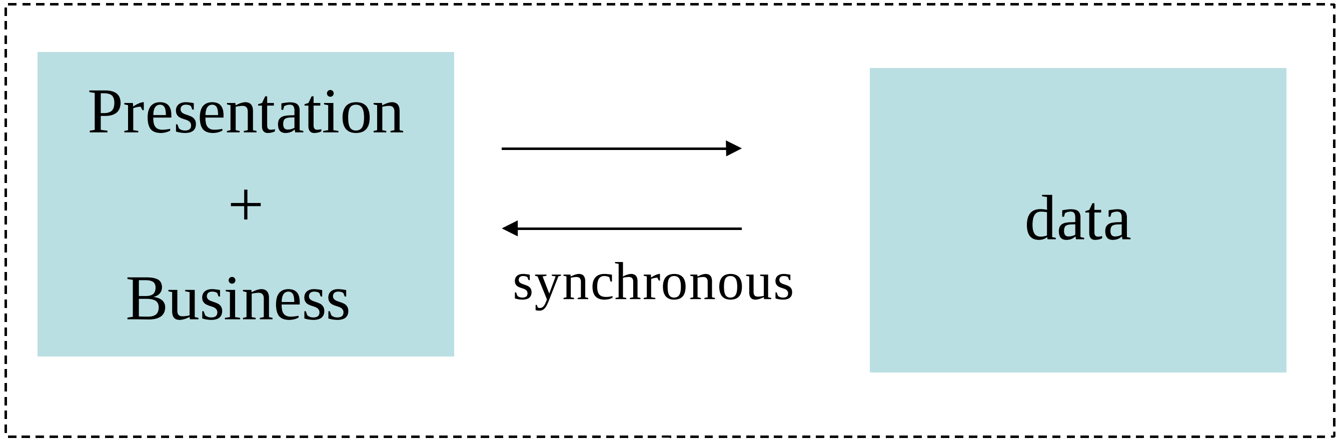
Minimal processing

Intelligent server

All application logic

Bulk processing

Two-tier client-server Architecture



Fat (intelligent) client

Bulk application logic

Bulk of processing

Database server

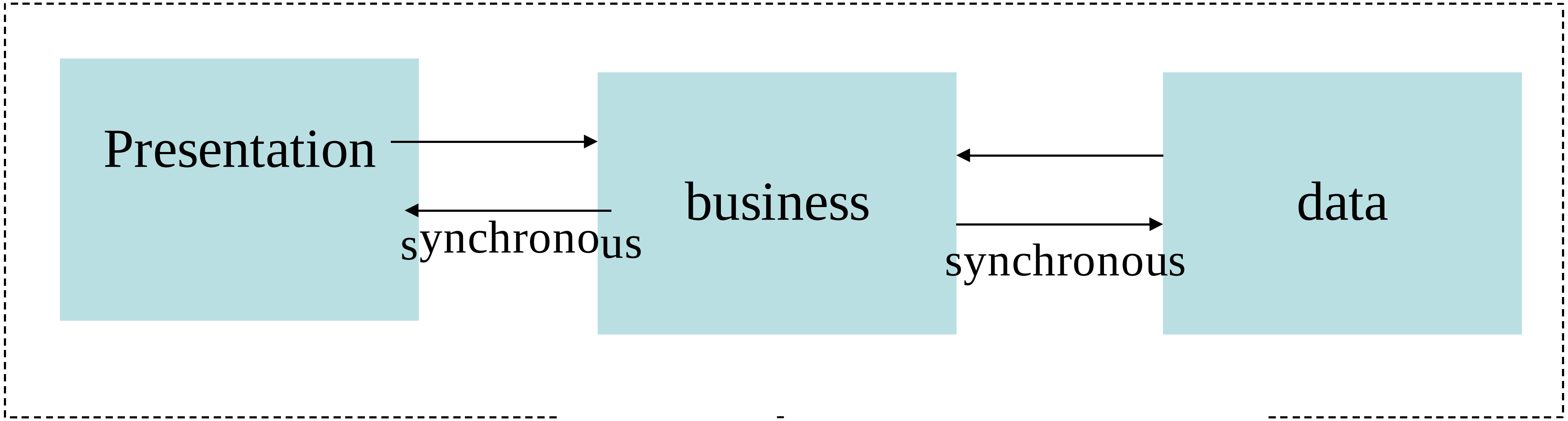
Data-related logic, some business rules

Data-related processing, one connection per user

SOA vs Client-Server Architecture

	CS environments	SOA
Application logic	**Exists in client**	**Presentation layer design is open and specific to solution's requirement**
Application Processing	**80%-20% exists in client and db server. Each client establish its own dB connection that is persistent and synchronous**	**Multiple servers each hosting sets of WS and supporting middleware. Many choices to position and deploy services. Communication can be synchronous or asynchronous promoting autonomous stateless services**
Technology	**4GL – VB in client** **Oracle, Sybase in server**	**Web technologies (HTML,HTTP) with XML data representation architecture and SOAP**
Security	**Sofisticated and simple**	**Complex WS-Security framework**

Multi-tier client-server Architecture



Semi intelligent client

Presentation related logic

Presentation related processing

Semi intelligent application server

Componentized application logic

Business rules resource pooling

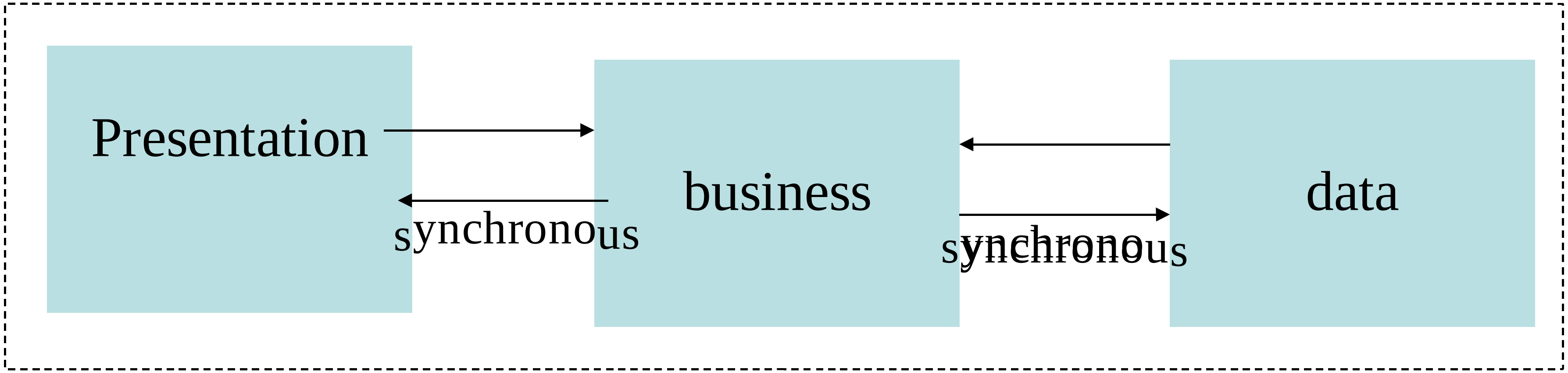
Bulk of processing

Database server

Data-related logic, some business rules

Data-related processing, less connection per user

Distributed internet Architecture



Presentation

business

data

synchronous

synchronous

dumb client (browser)

No installed logic

Presentation related processing

intelligent application and web server

Presentation logic, business rules, connection pooling

Componentized application logic Bulk processing

Database server

Data-related logic, some business rules

Data-related processing, less connection per user

SOA vs distributed internet Architecture

Design considerations

- How application logic should be partitioned
- Where the partitioned units of processing logic should reside
- How the units of processing logic should interact

- **Differences lie in the principles used to determine these three design considerations**

SOA vs distributed internet Architecture

	Distibuted internet architecture	SOA
Application logic	Components of varying degrees of functional granularity reside in 1 or more application servers. Communication is via API or RPC. Tightly bound component network – not easily altered	Functionality is wrapped within a serviceand exposed via open standard interface. Services communicate via SOAP messages. Loosely coupled solution agnostic servicespromotes reuse and cross application interoperability
Application Processing	Relies on proprietary communication protocols – efficent, reliable, support stateful and stateless components that interact with synchronous data exchanges	Communication – slower than RPC with processing overhead but it promotes creation of autonomous services that support wide range of message exchange patterns and optimizes processing by minimizing communication

References

- Coyle, “XML, Web Servcies and Data Revolution”, Pearson Education, 2002.
- Chatterjee and Webber, “Developing Enterprise Web Services – An Architect’s Guide”, Pearson Education, 2004.
- Liu, “Distributed Computing – Principles and Applications”, Pearson Education, 2004.
- http://www.microsoft.comarchitecture/soa
- http://www.ibm.com/soa
- http://www.sun.com/products/soa

Thank you