3-Tier Architecture

Prepared By

Channu Kambalyal

Table of Contents

1.0 Traditional Host Systems

A Central Processing System (Mainframe) provides all processing.

Local Terminals are responsible for display and keyboard for user input and viewing capabilities. Local Terminals do not contain any intelligent processing capabilities.

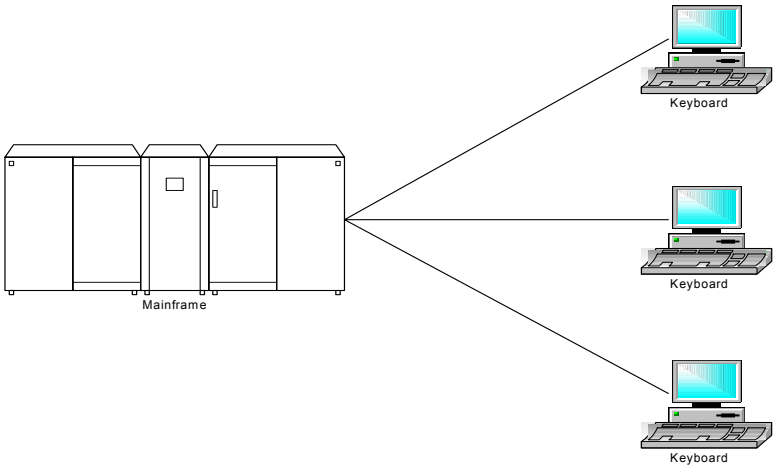

Figure 1.0.1 Non-Client-Server System

File Server and retrieval processing provided by File Server

Word Processing and spreadsheet processing provided by PC workstation.

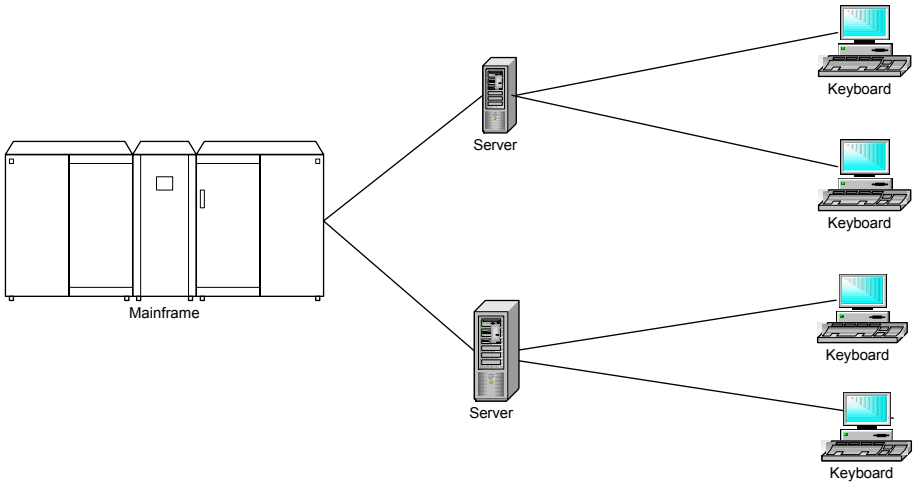

Figure 1.0.2 Traditional Host System with LAN

2.0 Distributed Systems

Distributed System

Both data and transaction processing are divided between one or more computers connected by a network, each computer playing a specific role in the system.

Replication

Ensures data at all sites in a distributed system reflects any changes made anywhere in the system.

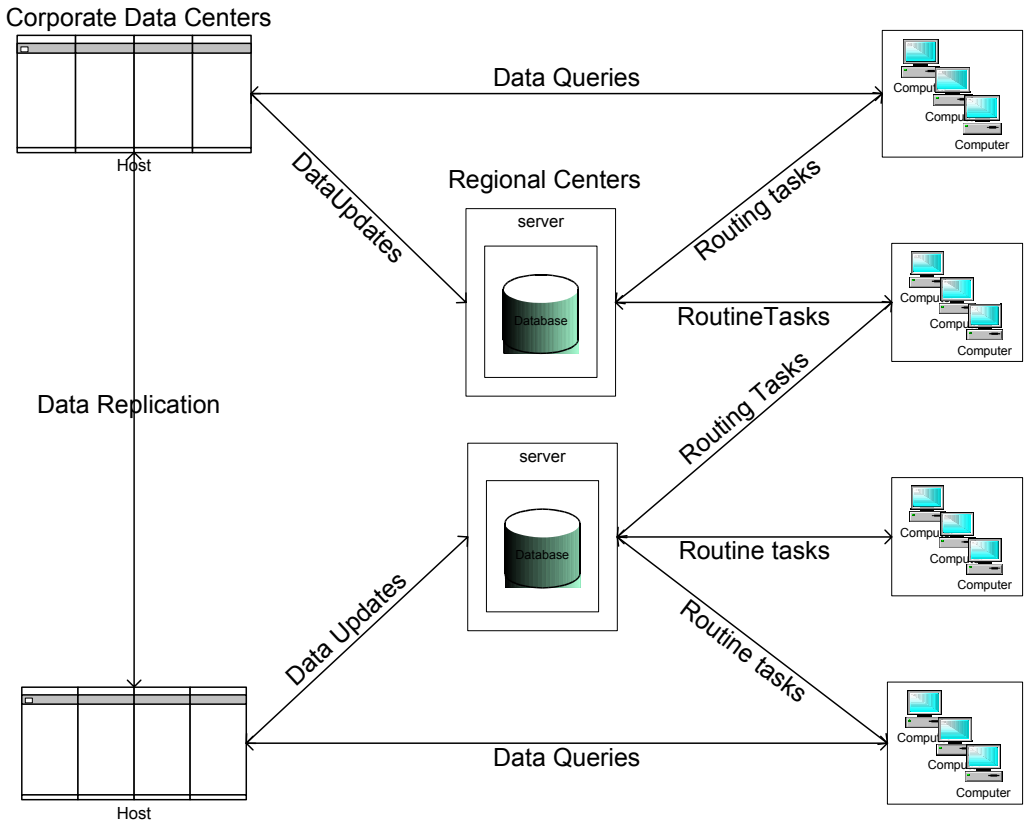

Figure 1.3. Distributed Data Centers

3.0 Client/Server Model

- Complements distributed systems
- Responds to limitations found in the two host data processing models:
    1. The traditional mainframe host model, in which a single mainframe provides shared data access to many dumb terminals, and;
    2. The local area network (LAN) model, in which many isolated systems access a file server that provides no central processing power.
- Provides integration of data and services
- Application Processing provided by multiple tiers –
    1. Database Server
    2. Application Server
    3. PC Workstation

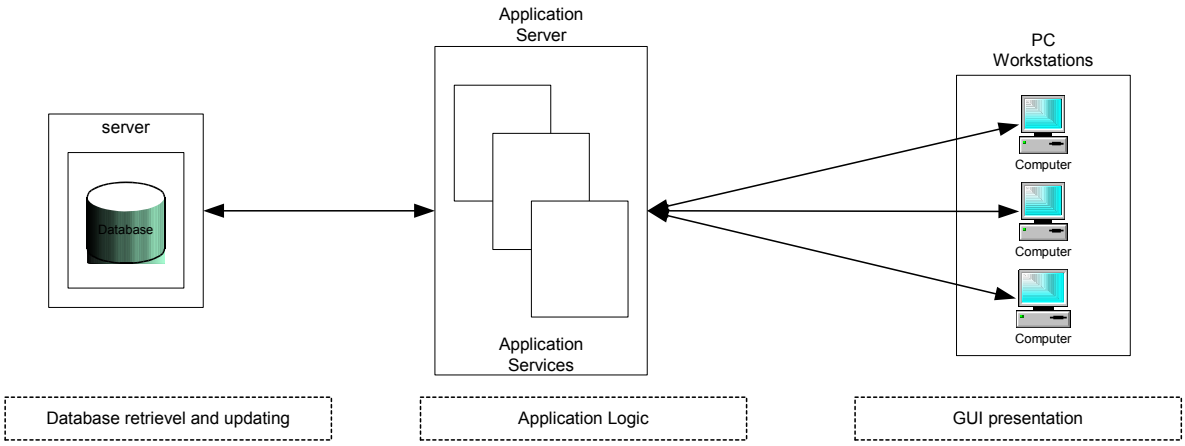

Figure 3.1 Client/Server 3-Tier Model

4.0 Distributed Client/Server Model

- Application processing provided by all tiers of the network –

    1. Mainframe
    2. Application Servers
    3. Workstations

- Multiple databases to support distributed data requirements
- Supports high volume, load balancing and scalability (extendibility)
- Requires extensive network administration and application management.

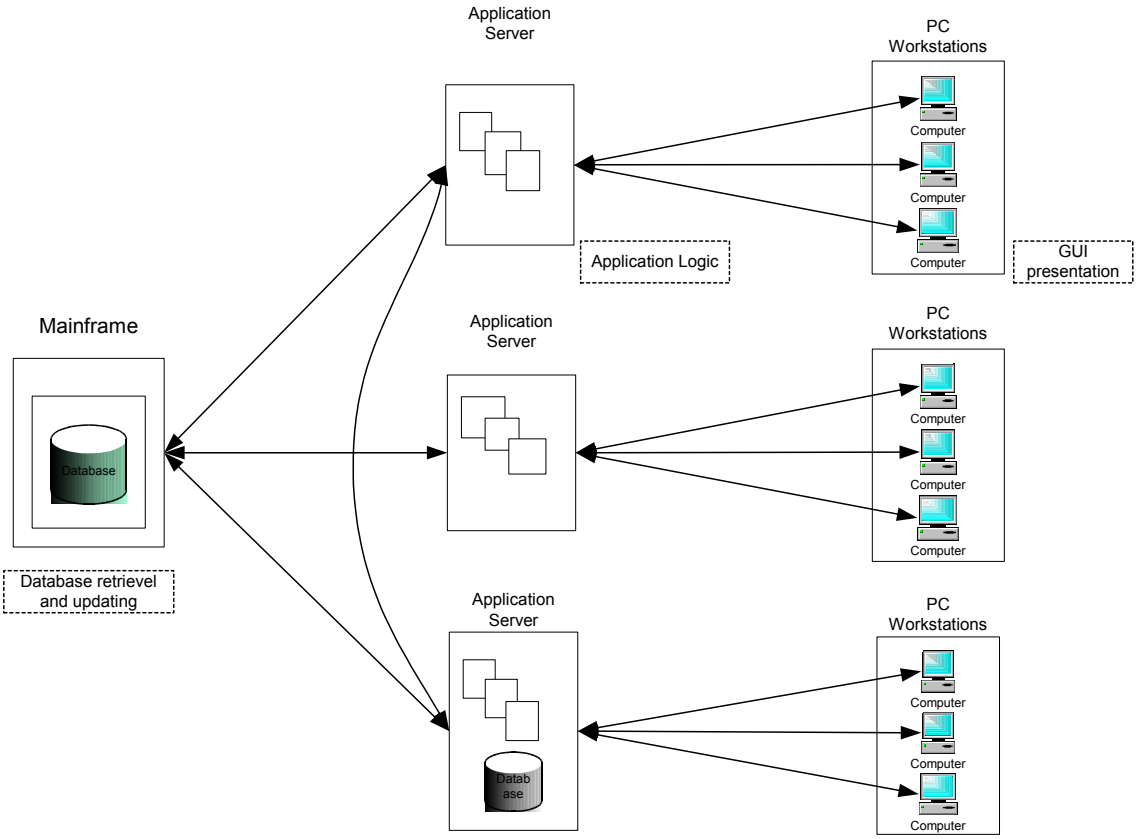

Figure 4.1 Distributed Client/Server Model

5.0 Inter-process Communication

- Basis for client/server computing
- Client process communicates with server process
- Each process performs separate functions
- Data is passed between processes using IPC functions

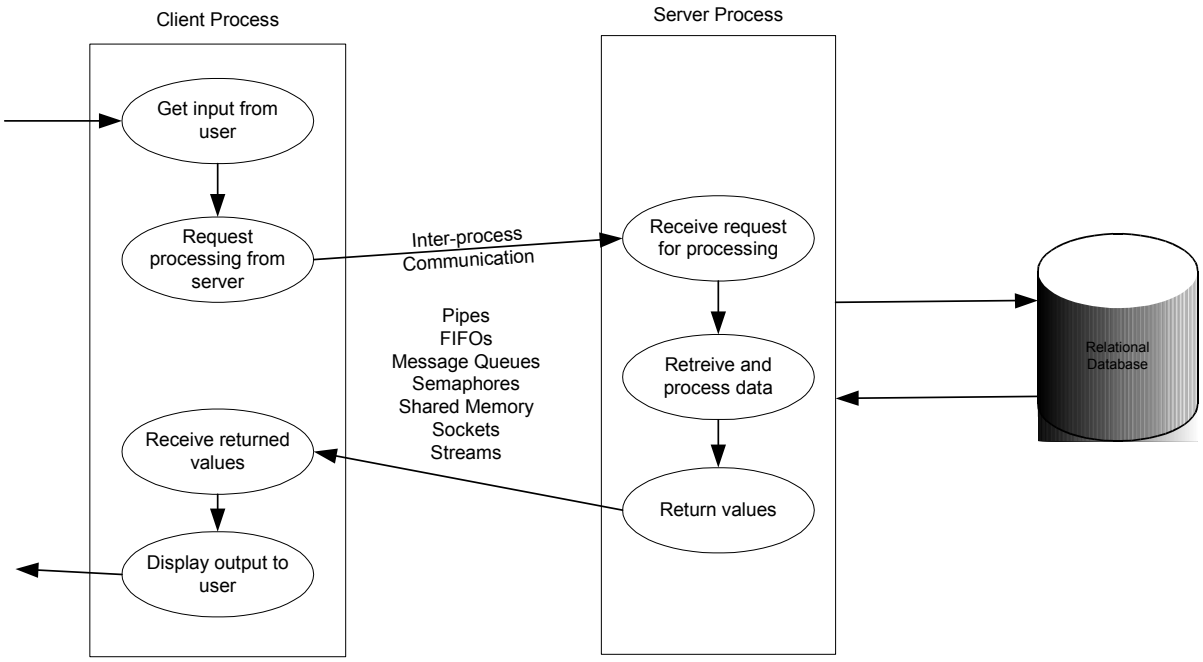

Figure 5.1 Inter-Process Communication

6.0 Benefits of the Client/Server Model

- **Divides Application Processing** across multiple machines:
  - Non-critical data and functions are processed on the client
  - Critical functions are processed on the server
- **Optimizes Client Workstations** for data input and presentation (e.g., graphics and mouse support)
- **Optimizes the Server** for data processing and storage (e.g., large amount of memory and disk space)
- **Scales Horizontally** – Multiple servers, each server having capabilities and processing power, can be added to distribute processing load.
- **Scales Vertically** - Can be moved to more powerful machines, such as minicomputer or a mainframe to take advantage of the larger system's performance
- **Reduces Data Replication** - Data stored on the servers instead of each client, reducing the amount of data replication for the application.

7.0 Client/Server 2-Tier Architecture

Two-tier client/server architectures have 2 essential components

1. A Client PC and
2. A Database Server

2-Tier Considerations:

- Client program accesses database directly
    - Requires a code change to port to a different database
    - Potential bottleneck for data requests
    - High volume of traffic due to data shipping
- Client program executes application logic
    - Limited by processing capability of client workstation (memory, CPU)
    - Requires application code to be distributed to each client workstation

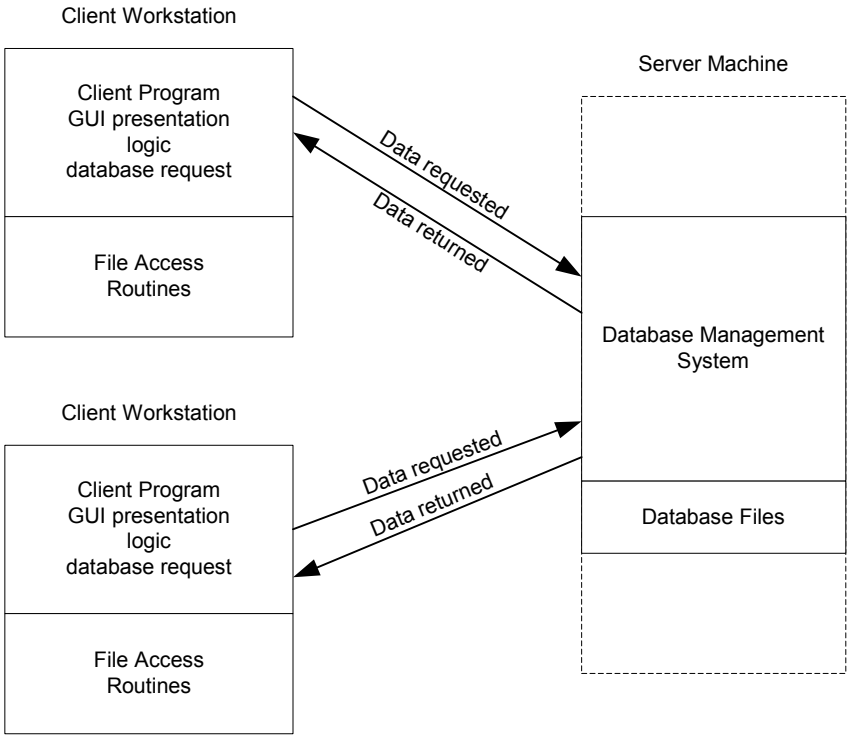

Figure 7.1 Client/Server 2-Tier Architecture

Two – Tier Pros and Cons

Advantages	Disadvantages
Development Issues: • Simple structure • Easy to setup and maintain	*Development Issues:* • Complex application rules difficult to implement in database server – requires more code for the client • Complex application rules difficult to implement in client and have poor performance • Changes to business logic not automatically enforced by a server – changes require new client side software to be distributed and installed • Not portable to other database server platforms
Performance: • Adequate performance for low to medium volume environments • Business logic and database are physically close, which provides higher performance.	*Performance:* • Inadequate performance for medium to high volume environments, since database server is required to perform business logic. This slows down database operations on database server.

8.0 3-Tier Client/Server Architecture

3-Tier client-server architectures have 3 essential components:

1. A Client PC
2. An Application Server
3. A Database Server

3-Tier Architecture Considerations:

- Client program contains presentation logic only
  - Less resources needed for client workstation
  - No client modification if database location changes
  - Less code to distribute to client workstations
- One server handles many client requests
  - More resources available for server program
  - Reduces data traffic on the network

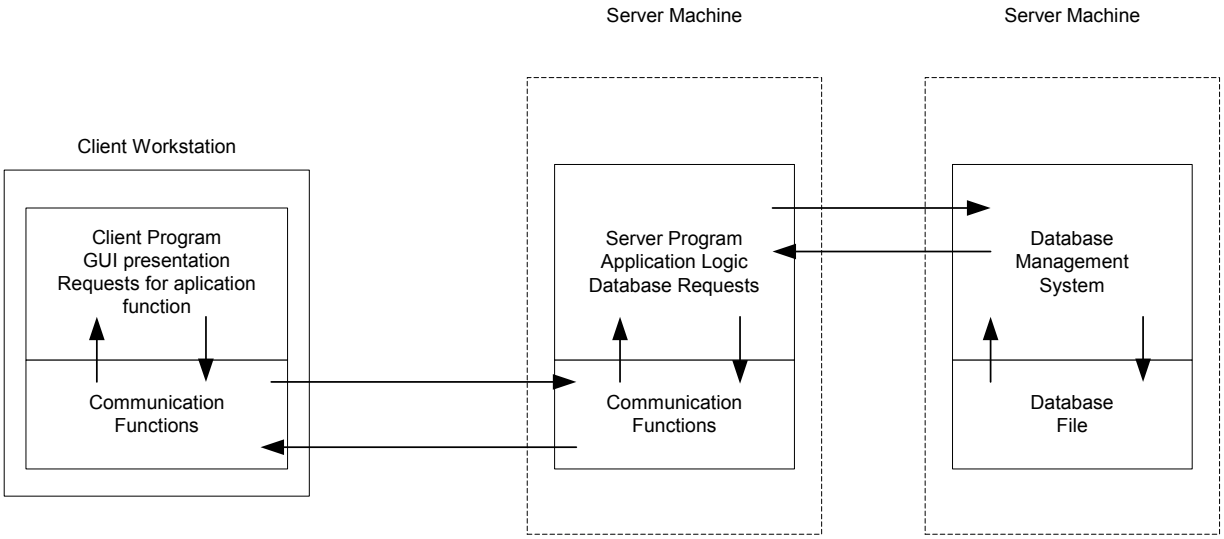

Figure 1.8. Typical 3 – Tier Architecture

3 – Tier Pros and Cons

Advantages	Disadvantages
Development Issues: • Complex application rules easy to implement in application server • Business logic off-loaded from database server and client, which improves performance • Changes to business logic automatically enforced by server – changes require only new application server software to be installed • Application server logic is portable to other database server platforms by virtue of the application software	*Development Issues:* • More complex structure • More difficult to setup and maintain.
Performance: • Superior performance for medium to high volume environments	*Performance:* • The physical separation of application servers containing business logic functions and database servers containing databases may moderately affect performance.

9.0 Middleware

Simplifies 3-tier application development and administration by providing an extra application server layer to manage communication between components.

Middleware Characteristics:

- Simplifies partitioning of application processing among clients and servers
- Manages distributed transactions among multiple databases
- Communicates with heterogeneous database products within a single application.
- Supports application scalability
- Supports service requests prioritization, load-balancing, data dependant routing and queuing.

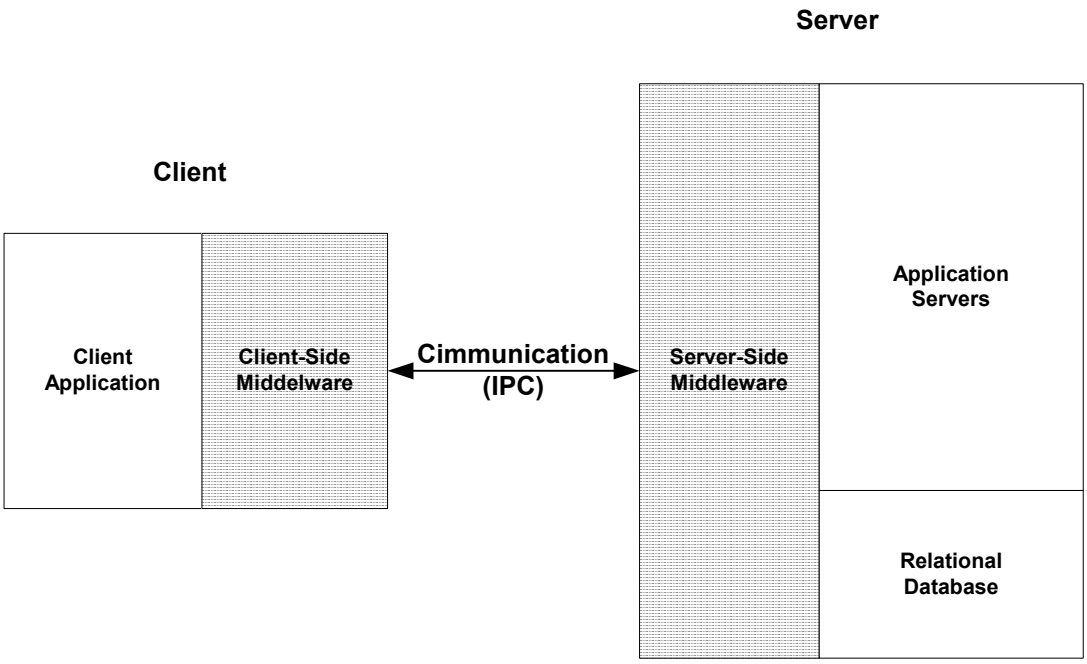

Figure 1.9 Middleware

10.0 Architectures in Discover Financial Services

10.1 Current DAS Architecture

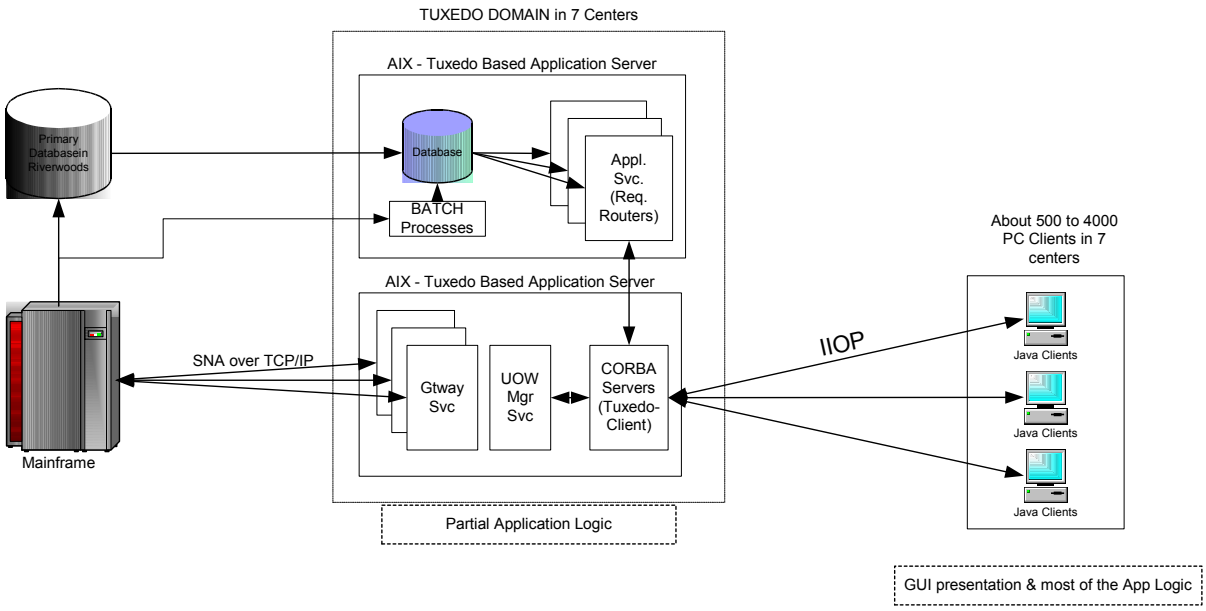

Fig. 10.1 Current DAS Architecture

Key Features:

1. DAS – built on DCE-RPC based BEA Tuxedo Middleware (C based)
2. Clients to Tuxedo based on CORBA based Visibroker (C++/Java based)
3. Java based client applications using IIOP for Corba
4. Mainframe connectivity uses SNA over TCP/IP

10.2 Migration from DAS-Tuxedo based to J2EE - WebSphere based system

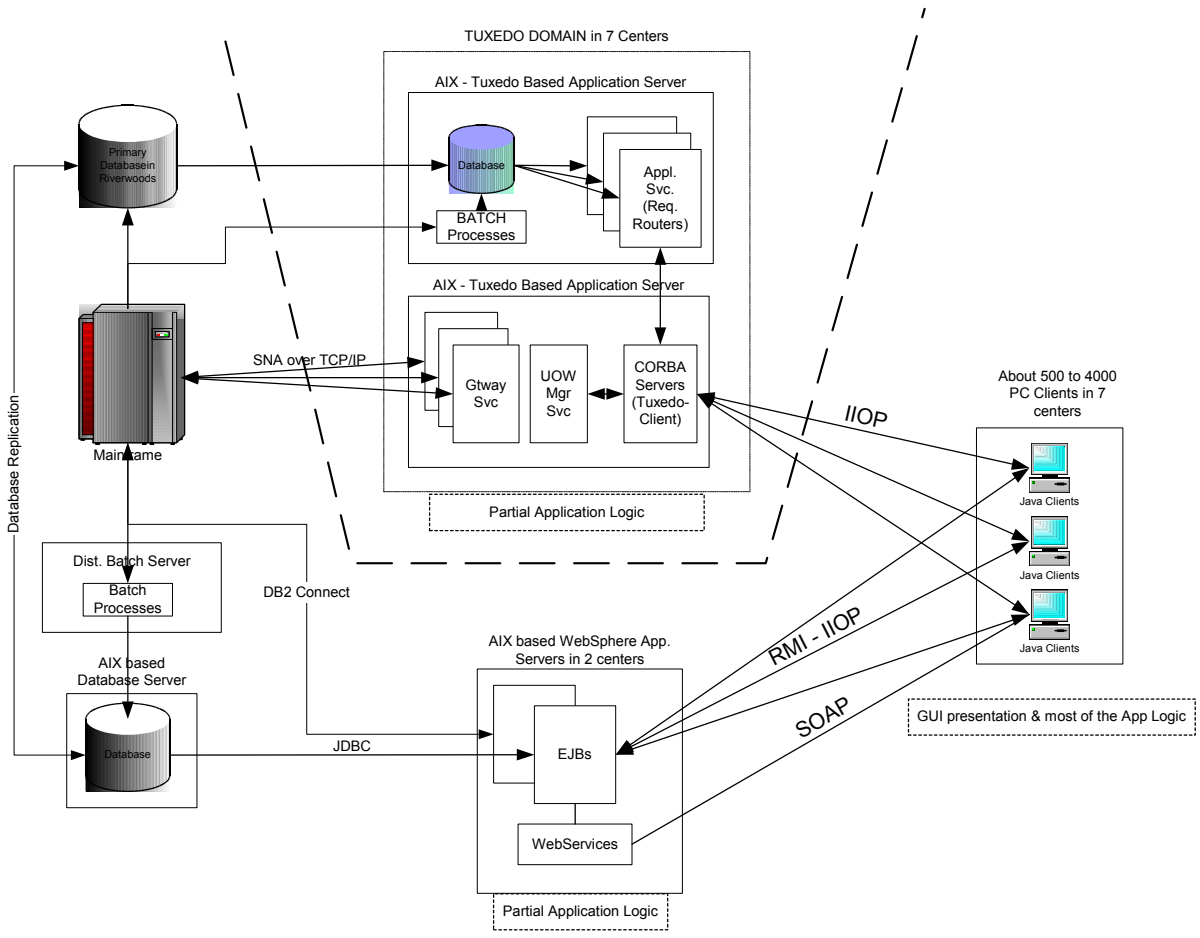

Fig. 10.2 Migration from DAS- Tuxedo to WebSphere based system

Current Status:

- Currently out of about 800 TPS about 45% of transaction currently run through DAS and remaining through WebSphere
- Expected date of migration is July/August 2005

10.3 Future WebSphere Based System

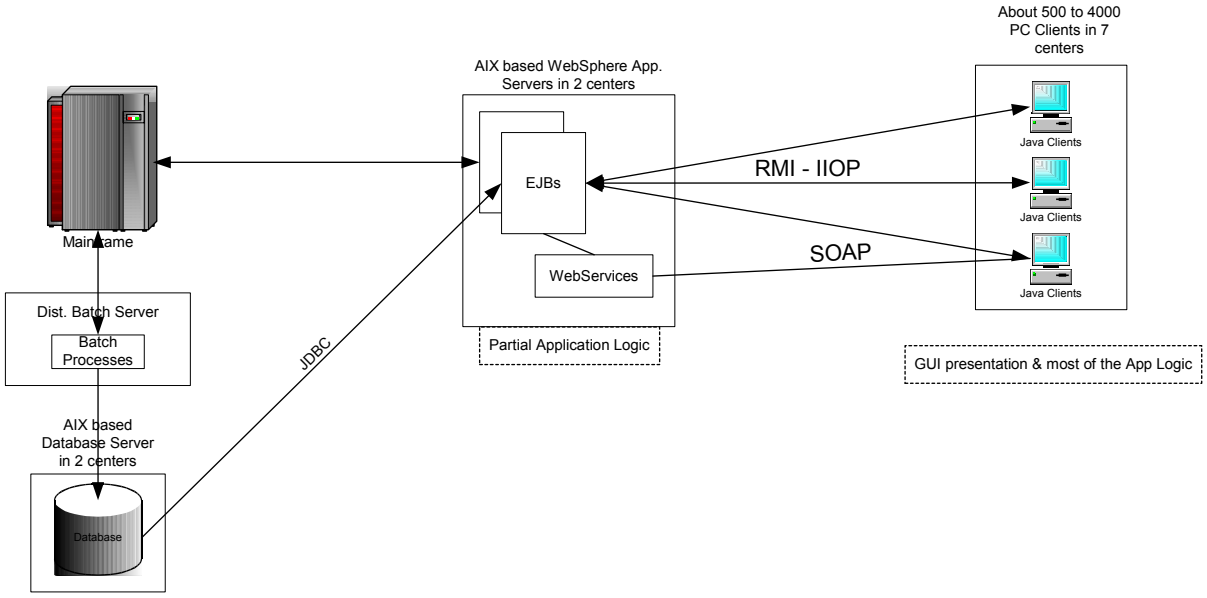

Fig 10.3 Future WebSphere Based 3-Tier Architecture

Key Features:

1. Latest J2EE based client applications using RMI over IIOP
2. Few SOAP based implementations with few Web-Services
3. J2EE based WebSphere Application Server
4. DB2 Connect used for Java – Mainframe – DB2 connectivity
5. JDBC used for Oracle – EJB connectivity and Transaction Management

11.0 Architecture Trends

11.1 Web Services, J2EE Connectors, Message Brokers, etc

Web Services

- A standardized way of integrating Web-based applications using the XML, SOAP, WSDL and UDDI
- A means for businesses to communicate with each other and with clients
- Allow organizations to communicate data without intimate knowledge of each other's IT systems behind the firewall
- Unlike traditional client/server models, such as a Web server/Web page system, Web services do not provide the user with a GUI. Web services instead share business logic, data and processes through a programmatic interface across a network. The applications interface, not the users. Developers can then add the Web service to a GUI (such as a Web page or an executable program) to offer specific functionality to users.
- Different applications from different sources can communicate with each other without time-consuming custom coding, as all communication is in XML, Web services are not tied to any one operating system or programming language. For example, Java can talk with Perl, Windows applications can talk with UNIX applications
- XML – E**x**tensible Markup Language - Used to tag the data
- SOAP – Simple Object Access Protocol - Used to transfer the data
- WSDL is used for describing the services available
- UDDI - Universal Description, Discovery and Integration **-** Used to list what services are available.

JMS – Java Message Service

- JMS defines the standard for reliable Enterprise Messaging
- Enterprise messaging, often also referred to as Messaging Oriented Middleware (MOM), universally recognized as an essential tool for building enterprise applications
- Provides a reliable, flexible service for the asynchronous exchange of critical business data and events throughout an enterprise.
- Message-driven beans enable the asynchronous consumption of JMS messages.
- Message sends and receives can participate in Java Transaction API (JTA) transactions.
- J2EE Connector Architecture allows JMS implementations from different vendors to be externally plugged into a J2EE 1.4 application server.

J2EE Connector Architecture

- The J2EE Connector architecture enables an EIS vendor to provide a standard resource adapter for its EIS.
- Resource adapter plugs into an application server, providing connectivity between the EIS, the application server, and the enterprise application.
- An EIS vendor needs to provide just one standard resource adapter, which has the capability to plug in to any application server that supports the J2EE Connector architecture.

See diagram of N-Tier Architecture for an overview of all these technologies and how they fit in.

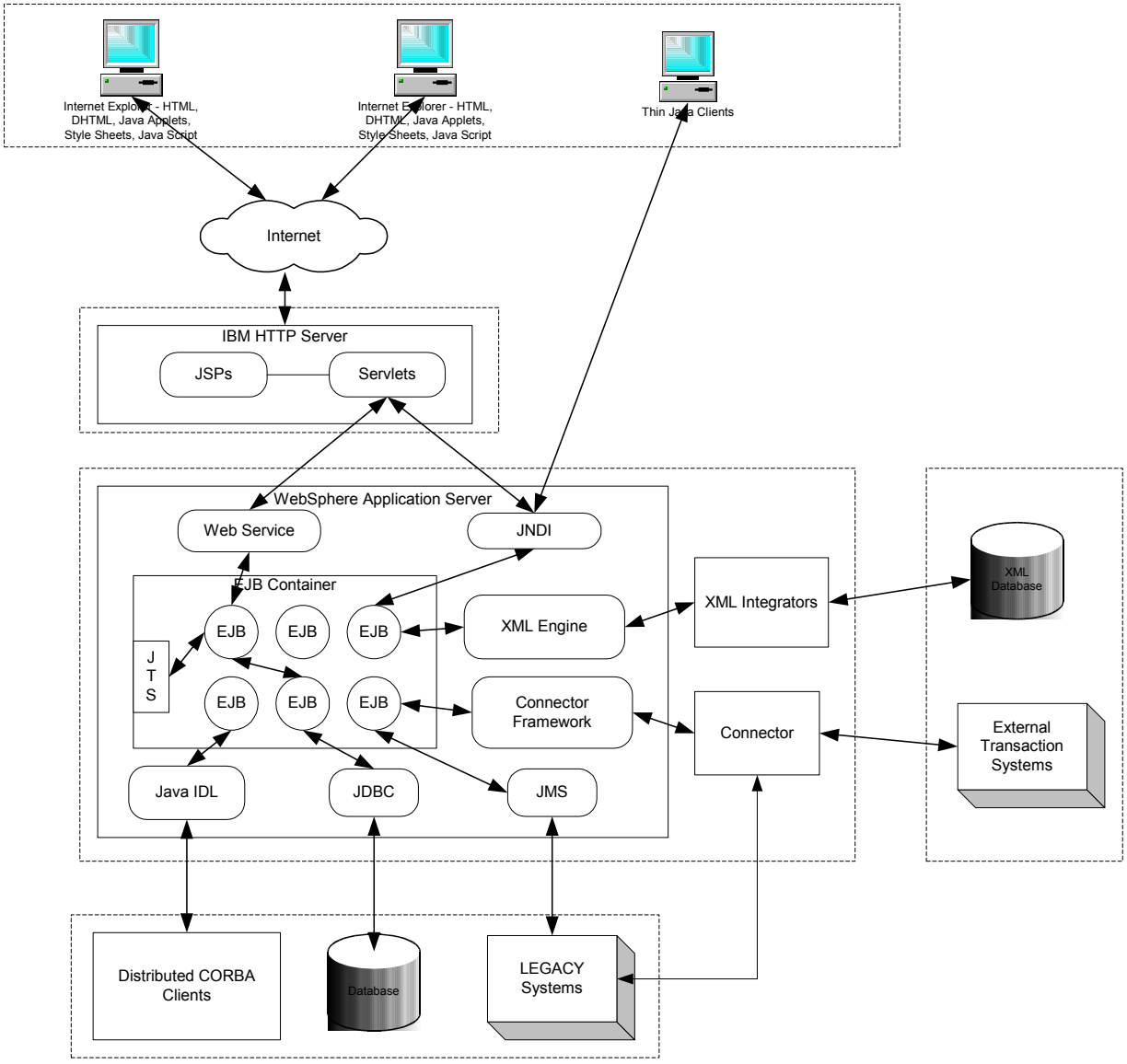

Figure 11.1 Typical N-Tier Architecture using Web Services, JMS, Connectors, CORBA, XML Databases, EJB, Servlets, JSP

11.2 Business Process Management (BPM)

- Business Process Management (BPM) enables the integration of diverse applications and human participants, as well as the coordinated exchange of information between trading partners outside of the enterprise.
- Focuses on Business Process Documentation to Executable!
- Further Reference - BPMI.org

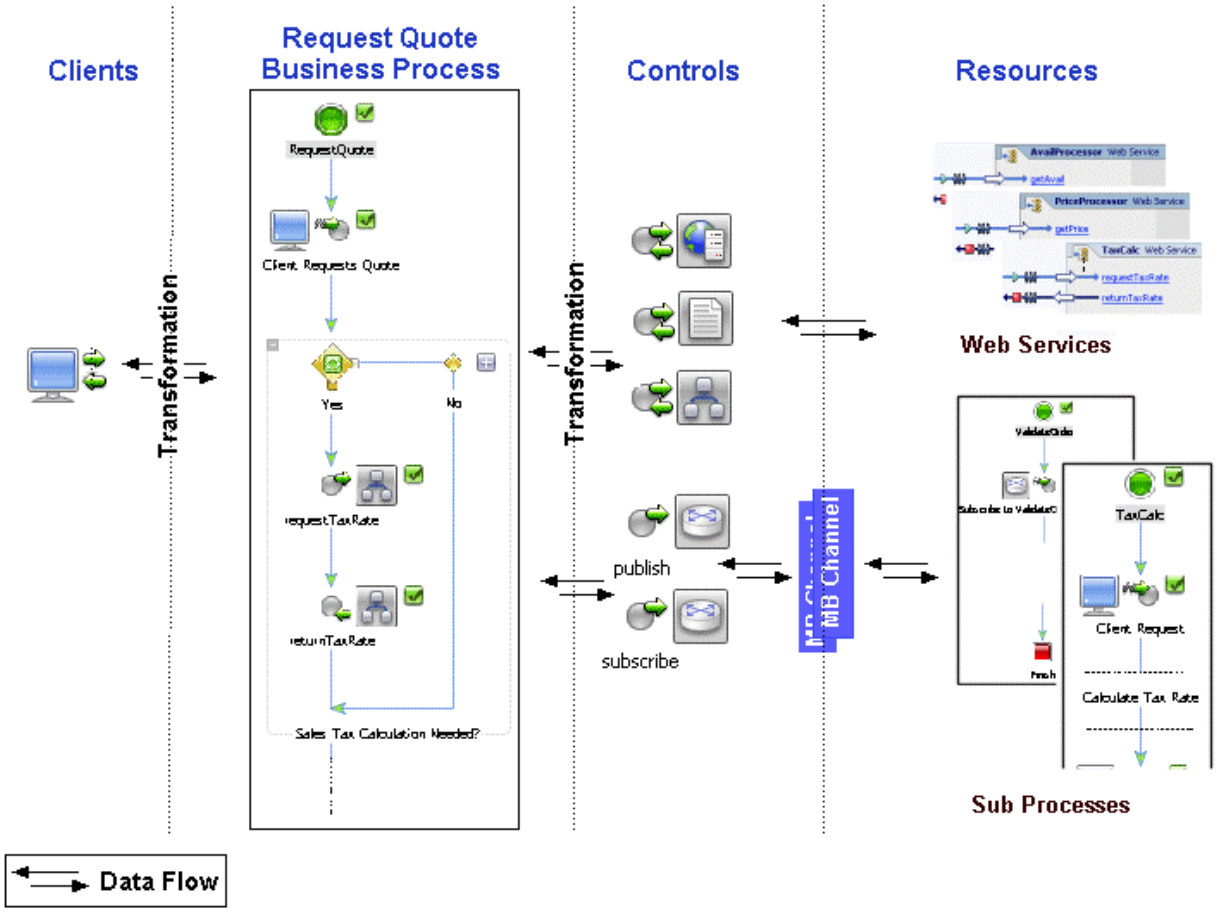

Figure 11.2 Typical Business Process Model
(Source: WebLogic Workshop Integrator 8.0)