Objectives

• Define an information system’s architecture in terms of the KNOWLEDGE, PROCESSES, and COMMUNICATION building blocks.
• Differentiate between logical and physical data flow diagrams, and explain how physical data flow diagrams are used to model an information system’s architecture.
• Describe centralized and distributed computing alternatives, including client/server and Internet-based computing options.
• Describe database and data distribution alternatives for system design.
• Describe user and system interface alternatives for system design.
• Describe various software development environments for information system design.
• Describe strategies for developing or determining architecture of an information system.
• Draw physical data flow diagrams for an information system’s architecture and processes.
Application Architecture – a specification of the technologies to be used to implement information systems. The blueprint to communicate the following design decisions:

- The degree to which the information system will be centralized or distributed.
- The distribution of stored data.
- The implementation technology for software developed in-house.
- The integration of commercial off-the-shelf software.
- The technology to be used to implement the user interface.
- The technology to be used to interface with other systems.
Physical data flow diagram (DFDs) – a process model used to communicate the technical implementation characteristics of an information system.

- Communicate technical choices and other design decisions to those who will actually construct and implement the system.
- Recall from Chapter 9 that DFDs are a type of process model.
Sample Physical Data Flow Diagram
Physical Processes

Physical process – either a processor, such as a computer or person, or a technical implementation of specific work to be performed, such as a computer program or manual process.

- Logical processes may be assigned to physical processors such as PCs, servers, people, or devices in a network. A physical DFD would model that network structure.
- Each logical process requires an implementation as one or more physical processes.
- A logical process may be split into multiple physical processes:
  - To define aspects performed by people or computers.
  - To define aspects implemented by different technologies.
  - To show multiple implementations of the same process.
  - To add processes for exceptions and security.
Physical Process Notation

- ID (optional)
- Action Verb
  + Noun or Object Phrase
- Implementation
Samples of Physical Processes

<table>
<thead>
<tr>
<th>Logical Process</th>
<th>Sample Physical Process Implementations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check Customer Credit</td>
<td></td>
</tr>
<tr>
<td>Acct Clerk</td>
<td>Check Customer Credit</td>
</tr>
<tr>
<td></td>
<td>COBOL/CICS</td>
</tr>
<tr>
<td></td>
<td>Check Customer Credit</td>
</tr>
<tr>
<td></td>
<td>Visual Basic</td>
</tr>
<tr>
<td></td>
<td>Check Customer Credit</td>
</tr>
<tr>
<td></td>
<td>Quickbooks</td>
</tr>
</tbody>
</table>
Possible Computer Process Implementations

- A purchased application software package
  - Also called *commercial off-the-shelf (COTS) software*

- A system or utility program
  - Such as an e-mail/message server or third-party framework

- An existing application program from a program library
  - May require modification

- A program to be written
Sample Physical Process Implementations

Logical Process | Sample Physical Process Implementation
--- | ---
4.3 | 4.3.A
Check Customer Credit | Credit Rejection
Visual Basic | 4.3.B
Reconsider Credit Decision | Credit Mgr

Logical Process | Sample Physical Process Implementations
--- | ---
4.3 | 4.3.A
Check Customer Credit | CHK_CREDIT.COB
COBOL + CICS | 4.3.B
appCheckCredit.vbx
Visual Basic
Physical Data Flows

A physical data flow represents:

- Planned implementation of an input to, or output from a physical process.
- Database command or action such as create, read, update, or delete.
- Import of data from, or export of data to another information system.
- Flow of data between two modules or subroutines (represented as physical processes).

Implementation method:

Data flow name

OR

Data flow name

(Implementation method)
### Sample Physical Data Flows

<table>
<thead>
<tr>
<th>Logical Data Flow</th>
<th>Implementation</th>
<th>Sample Physical Data Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order</td>
<td>Computer Input (Keyboard)</td>
<td>WIN 2000 GUI: Order Form</td>
</tr>
<tr>
<td>Order</td>
<td>Computer Input (Internet)</td>
<td>HTML: Order Form</td>
</tr>
<tr>
<td>Product Sold</td>
<td>Computer Input (Keyless)</td>
<td>BAR CODE: Product UPC</td>
</tr>
<tr>
<td>Hours Worked</td>
<td>Computer Input (Batch File)</td>
<td>KEY-TO-DISK: Hours Worked</td>
</tr>
<tr>
<td>Salary Equity Analysis</td>
<td>Computer Output (Printed)</td>
<td>PRINTOUT: Salary Equity Report</td>
</tr>
<tr>
<td>Account History</td>
<td>Computer Output (On-Line)</td>
<td>WIN 2000 GUI: Account History</td>
</tr>
<tr>
<td>Create Order</td>
<td>Create a record in a database</td>
<td>SQL Insert: New Order</td>
</tr>
</tbody>
</table>
Sample Physical Data Flows (continued)

<table>
<thead>
<tr>
<th>Unfilled Orders</th>
<th>SQL Select: Unfilled Orders</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read records in a database</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Update Credit rating</th>
<th>SQL Update: Credit Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update a record in a database</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Delete Employee</th>
<th>SQL Delete: Employee</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete a record in a database</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Insurance Accident Claim</th>
<th>IMAGE FILE: Insurance Accident Claim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import a data file</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Schedule of Classes</th>
<th>Comma Delimited File: Schedule of Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Export a data file</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Extended Cost</th>
<th>Form 23: Course Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pass data between modules of a program</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Course Request</th>
<th>Pass a manual form</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Physical External Agents

Physical external agents are carried over from the logical DFD models.

- If scope changes, the logical models should be changed before the physical models are drawn.
Physical Data Stores

A physical data store represents the planned implementation of one of:

- A database
- A table in a database
- A computer file
- A tape or media backup of anything important
- A temporary file or batch
- Any type of noncomputerized file
### Physical Data Store Notation

<table>
<thead>
<tr>
<th>ID (opt)</th>
<th>Implementation Method: Data Store Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID (opt)</td>
<td>Data Store Name (Implementation Method)</td>
</tr>
<tr>
<td>Logical Data Store</td>
<td>Implementation</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------</td>
</tr>
<tr>
<td>Human Resources</td>
<td>A database (multiple tables)</td>
</tr>
<tr>
<td>Marketing</td>
<td>A database view (subset of a database)</td>
</tr>
<tr>
<td>Purchase Orders</td>
<td>A table in a database</td>
</tr>
<tr>
<td>Accounts Receivable</td>
<td>A legacy file</td>
</tr>
<tr>
<td>Tax Rates</td>
<td>Static data</td>
</tr>
<tr>
<td>Orders</td>
<td>An off-line archive</td>
</tr>
<tr>
<td>Employees</td>
<td>A file of paper records</td>
</tr>
<tr>
<td>Faculty/Staff Contact Data</td>
<td>A directory</td>
</tr>
<tr>
<td>Course Enrollments By Date</td>
<td>Archived reports (for reuse and recall)</td>
</tr>
</tbody>
</table>
Distributed versus Centralized Systems

**Distributed system** – a system in which components are distributed across multiple locations and computer networks.

- Accordingly, the processing workload is distributed across multiple computers on the network.

**Centralized systems** – a system in which all components are hosted by a central, multi-user computer.

- Users interact with the system via terminals (or a PC emulating a terminal).
- Virtually all the actual processing and work is done on the host computer.
Why the Trend Toward Distributed Systems?

• Modern businesses are already decentralized (distributed).
• Distributed computing moves information and services closer to the customers and users who need them.
• Distributed computing consolidates the power of personal computers across the enterprise.
• Distributed computing solutions are in general more user-friendly because they use the PC as the user interface processor.
• Personal computers and network servers are less expensive than mainframe computers
  • Though total cost of ownership is at least as expensive
Computing Layers

- **Presentation layer**—the user interface

- **Presentation logic layer**—processing that must be done to generate the presentation, such as editing input data or formatting output data.

- **Application logic layer**—the logic and processing to support business rules, policies, and procedures

- **Data manipulation layer**—to store and retrieve data to and from the database

- **Data layer**—the actual business data
Types of Distributed Computing

- **File Server Solution**
  - Data Layer: Stored on the File Server
  - Data Manipulation Layer: Executed on the Client
  - Application Logic Layer: Executed on the Client
  - Presentation Logic Layer: Executed on the Client
  - Presentation Layer: Displayed on the Client

- **Client/Server Solutions**
  - Distributed Presentation (2 Tier)
  - Distributed Data (2 Tier)
  - Distributed Data & Application (N Tier)

- **Network Computing Solution**
  - Data Layer: Stored on the Database Server
  - Data Manipulation Layer: Executed on the Database Server
  - Application Logic Layer: Executed on the Database Server
  - Presentation Logic Layer: Executed on the Client
  - Presentation Layer: Distributed from the Web Server
Local area network (LAN) – a set of client computers (PCs) connected over a relatively short distance to one or more servers.

File server system – a LAN in which a server hosts the data of an information system.

- All other layers are implemented on the client computers.
- Frequently excessive network traffic to transport data between servers and clients.
- Client must be fairly robust (“fat”) because it does most of the work.
- Database integrity can be compromised.
File Server Architecture

1. Presentation
2. Request to create, read, update, or delete 1 or more records
3. Entire tables
4. Table Locked Until Client Returns Table
5. Response to request returns entire tables
6. Entire Tables with any Updated Records
7. Updated Tables
8. Unlock Tables

Client PC

File Server

Presentation, Application, and Data Manipulation Logic all executed here

File Server Database (e.g. MS Access)
Client/Server Architecture — Clients

**Client/server system** — a distributed computing solution in which the presentation, presentation logic, application logic, data manipulation, and data layers are distributed between client PCs and one or more servers.

**Thin client** — a personal computer that does not have to be very powerful because it only presents the user interface to the user.

**Fat client** — a personal computer, notebook computer, or workstation that is typically powerful.
Client/Server Architecture — Servers

- **Database server** — a server that hosts one or more databases and executes all data manipulation commands at the server.
- **Transaction server** — a server that hosts services which ensure that all database updates for a transaction succeed or fail as a whole.
- **Application server** — a server that hosts application logic and services for an information system.
- **Messaging or groupware server** — a server that hosts services for e-mail, calendaring, and other work group functionality.
- **Web server** — a server that hosts Internet or intranet websites.
Distributed presentation – a client/server system in which the presentation and presentation logic layers are shifted from the server to reside on the client.

- The application logic, data manipulation, and data layers remain on the server (frequently a mainframe).
- Character user interface (CUI)
- Graphical user interface (GUI)
Building a GUI From a CUI – Screen Scrapers
Client/Server—Distributed Presentation

- User interacts with Presentation Logic
- Presentation Logic is executed on the Client PC
- Legacy App (e.g., COBOL) handles Application and Data Manipulation Logic
- Mainframe performs Reads and/or Updates
- Conventional Files (e.g., VSAM) or Database (e.g., Oracle) are managed separately

GUI: Input and/or Commands for Processing
Output and Instructions for translation to GUI
Distributed data – a client/server system in which the data and data manipulation layers are placed on the server(s), and other layers are placed on the clients.

- Sometimes called two-tiered client/server computing.
- Difference to file server systems is where the data manipulation commands are executed.
- Much less network traffic than file server systems because only the database requests and the results of those requests are transported across the network.
- Database integrity is easier to maintain.
Client/Server—Distributed Data
Distributed data and application – client/server system:

1. The data and data manipulation layers are placed on their own server(s),
2. The application logic is placed on its own server,
3. The presentation logic and presentation layers are placed on the clients.

- Also called three-tiered or n-tiered client/server computing.
- Requires design partitioning.

Partitioning – the art of determining how to best distribute (duplicate) application components across the network.
Client/Server — Distributed Data and Application
Network computing system – presentation and presentation logic layers implemented in client-side Web browsers using content downloaded from a Web server.

- Presentation logic layer connects to application logic layer running on application server, which connects to database servers on the backside of the system.
- The greatest potential of this approach is its applicability to redesign of traditional information systems to run on an intranet.

Intranet – a secure network that uses Internet technology to integrate desktop, work group, and enterprise computing into a cohesive framework.
Network Computing System: Internet/Intranet
Internet and Intranet Technologies

- **Java**
  - Mostly for programming server-side application logic called “servlets”
  - Occasionally for programming client-side application logic called “applets”

- **HTML (HyperText Markup Language)**
  - Mostly for programming the presentation layer

- **XML (Extensible Markup Language)**
  - Mostly for programming data content to be transported across the web

- **SQL (Structured Query Language)**
  - Universal standard language for database manipulation

- **Web Browsers**
Data Architectures

**Relational database** stores data in tabular form. Each file is implemented as a table. Each field is a column in the table. Related records between two tables are implemented by intentionally duplicated columns in the two tables.

**Distributed relational database** – A database system that duplicates tables to multiple database servers located in geographically important locations.

**Distributed relational database management system** – a software program that controls access to and maintenance of stored data in the relational format.
**Types of Data(base) Distribution**

**Data partitioning** truly distributes rows and columns of tables to specific database servers with little or no duplication between servers.
- Vertical partitioning assigns different columns to different servers.
- Horizontal partitioning assigns different rows to different servers.

**Data replication** duplicates some or all tables on more than one database server.
- Propagates updates on one database server to any other database server where the data is duplicated.
# Data Partitioning versus Data Replication

<table>
<thead>
<tr>
<th>Logical Data Store</th>
<th>Physical Data Stores Using Partitioning</th>
<th>Physical Data Stores Using Replication</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>Oracle 7:</strong> REGION 1 CUSTOMERS</td>
<td>Not applicable. Branch offices do not need access to data about customers outside of their own sales region.</td>
</tr>
<tr>
<td>CUSTOMERS</td>
<td>1P.#</td>
<td><strong>Oracle 8i:</strong> PRODUCTS (Master)</td>
</tr>
<tr>
<td></td>
<td><strong>Oracle 7:</strong> REGION 2 CUSTOMERS</td>
<td><strong>Oracle 8i:</strong> PRODUCTS (Replicated Copy)</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODUCTS</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Not applicable. All branch offices need access to data for all products, regardless of sales region.
Interface Architectures – Inputs, Outputs, & Middleware

- Batch inputs and outputs
- Online inputs and outputs
- Remote batch
- Keyless data entry (and automatic identification)
- Pen input
- Electronic messaging and work group technology
- Electronic Data Interchange (EDI)
- Imaging and document interchange
- Middleware
### Batch Inputs and Outputs

<table>
<thead>
<tr>
<th>Logical Data Flow (input)</th>
<th>Physical Data Flow Implementation (as batch input)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME CARD</td>
<td>KTD Batch: TIME CARDS batch Comma delimited file: TIMECARDS</td>
</tr>
<tr>
<td></td>
<td>KTD Batch: TIME CARDS End of Month -1 day</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Logical Data Flow (output)</th>
<th>Physical Data Flow Implementation (as batch output on preprinted forms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAYCHECK</td>
<td>Preprinted Form Batch: PAYCHECKS</td>
</tr>
</tbody>
</table>
On-Line Inputs and Outputs

Logical Data Flow (input and output)

- INSURANCE CLAIM

Physical Data Flow Implementation (as on-line input and output; 2 alternatives shown)

- Win 2000 Form: INSURANCE CLAIM
- HTML Form: ORDER CONFIRMATION
  - MAPI Email Message: ORDER CONFIRMATION

ORDER CONFIRMATION
Remote Batch
Keyless Data Entry (and Automatic Identification)
Pen Input

Pen: Customer Signature

Pen: Package Delivery

Cellular: Package Delivery
Electronic Data Interchange (EDI) – the standardized electronic flow of business transactions or data between businesses.

• Typically, many businesses must agree to a common data format to make EDI feasible.
Middleware – utility software that enables communication between different processors in a system.

- It may be built into the respective operating systems or added through purchased middleware products.
- Presentation middleware
- Application middleware
- Database middleware
Process Architectures

Software development environment (SDE) – a language and tool kit for developing applications.

- SDEs exist for centralized computing
- SDEs exist for distributed presentation
- SDEs exist for two-tiered client/server
- SDEs exist for multi-tiered client/server
- SDEs exist for Internet and intranet client/server

Clean layering – a design strategy that requires that presentation, application, and data layers of an application be physically separated.

- Allows components of each layer to be revised or enhanced without affecting the other layers.
Application Architecture Strategies for System Design

• The Enterprise Application Architecture Strategy
  • Enterprise-wide information technology architecture to be followed in all development projects.
    • Approved network, data, interface, and processing technologies and development tools.
    • Strategy for integrating legacy systems and technologies.
    • On-going process for continuously reviewing application architecture.
    • On-going process for researching emerging technologies
    • Process for analyzing requests for variances from the above.

• The Tactical Application Architecture Strategy
  • Defines architecture for each new system on an application-by-application basis as needed.
  • Requires feasibility analysis for each application.
Drawing Physical DFDs for Network Architecture

• Develop a physical data flow diagram (DFD) for the network architecture.
  • Each process symbol represents a server or class of clients.

• For each processor, develop a physical DFD to show the event processes (from Chapter 9) that are assigned to that processor.

• All but simple processes should be factored into design units and modeled as a more detailed physical DFDs.
**Design Units**

**Design unit** – a self-contained collection of processes, data stores, and data flows that share similar design characteristics.

- A design unit serves as a subset of the total system whose inputs, outputs, files and databases, and programs can be designed, constructed, and tested as a self-contained unit.
- Ultimately, design units must be integrated into a whole system.
Network architecture – a physical DFD that allocates processors (clients and servers) and devices (machines and robots) to a network and establishes:

- the connectivity between clients and servers
- where users will interface with the processors
Network Architecture DFD
Data Distribution Options

• Store all data on a single server.
• Store specific tables on different servers.
• Store subsets of specific tables on different servers.
• Replicate (duplicate) specific tables or subsets on different servers.
Data Distribution and Technology Assignments DFD
Process Distribution and Technology Assignments

- For two-tiered client/server systems, all logical even diagrams are assigned to the client.
- For three-tiered client/server and network computing systems, must closely examine each event’s primitive (detailed) DFD.
  - Determine which primitive processes should be assigned to the client and which should be assigned to an application server.
  - Generally data capture and editing are assigned to servers.
  - If different aspects of a single DFD are partitioned to different clients and servers, draw separate physical DFD for each.
Physical DFD for an Event

See Figure 13-13 in text for a more readable version
The Person/Machine Boundary

See Figure 13-14 in text for a more readable version
A Manual Design Unit