Distributed Systems: The Overall Architecture

Chapter 5

Outline

- Introduction
  - Four Attributes of Distributed Systems
  - When to Distribute Computing Responsibilities
  - Two Guiding Frameworks
- Seven types of Distributed Systems
  - Host-Based Hierarchy
  - Decentralized Stand-Alone Systems
  - Peer-to-Peer LAN-Based Systems
  - Hybrid Enterprise wide Systems
  - Client-Server Systems
  - Internet-Based Computing
  - Web Services
- Defining the Overall IT Architecture
  - An Enterprise Architecture Framework
  - The Coming Architecture: Service-Oriented Architecture
- The Importance of the IT Infrastructure
  - The Structure of the IT Infrastructure
  - Three Views of Infrastructure

Definitions

- An IT architecture is a blueprint. A blueprint shows how a system, house, vehicle, or product will look and how the parts interrelate.
- An IT infrastructure is the implementation of an architecture. The IT infrastructure includes the processors, software, databases, electronic links, and data centers as well as the standards that ensure the components work together, the skills for managing the operation etc.
- Lately, rather than talk about hardware, software, data, communications etc. as the components of computing, some people now refer only to applications and infrastructure

The Evolution of Distributed Systems

- First IT architecture mainframes (batch processing) with dumb terminals (no processing capabilities)
- With minicomputers moved into departments but the ‘master-slave’ computing model persisted. Processing was mainly centralized
- Microcomputer model changed significantly because processing power moved first onto desktops, then into briefcases and now into pockets
- Now power returning to a type of centralized processing with networks of servers and the Internet
  - Information appliances and thin clients make requests
  - More important in the future as Web Services continues to develop

Four Attributes of Distributed Systems

- The degree to which a system is distributed can be determined by answering four questions:
  - Where is the processing done?
  - How are the processors and other devices interconnected?
  - Where is the information stored?
  - What rules or standards are used?

Where is the processing done?

- **Distributed Processing** is the ability for more than one interconnected processor to be operating at the same time, typically for processing an application on more than one computer at a time
- The goal is move the appropriate processing as close to the user as possible and to let other machines handle the work they do best
- Permits interoperability-capability of different computers using different OS on different networks to work together on tasks
- Two forms of interoperability:
  - Communication between systems
  - Two-way flow between user applications

How are the processors and other devices interconnected?

- **Connectivity Among Processors** means that each processor in a distributed system can send data and messages to any other processor through electronic communication links
- Desirable to have at least two independent paths between two nodes to provide automatic alternate routing (Planned Redundancy)

Where is the information stored?

- **Distributed Databases** either:
  - Divide a database and distribute its portions throughout a system without duplicating the data
    - Users do not need to know where a piece of data is located to access it, because the system knows where all the data is stored
  - Store the same data at several different locations, with one site containing the master file
    - Synchronization of data is a significant problem

What rules or standards are used?

- **System wide Rules** mean that an operating discipline for the distributed system has been developed and is enforced at all times
- These rules govern communication between nodes, security, data accessibility, program and file transfers, and common operating procedures
- Since the 1990s, Open systems concept-mix products from vendors using open standards. Based on open-systems - standardized interfaces that allow products to inter-operate across multi-vendor networks, operating systems and databases
- Now API define the way to present data to another system component. Makes writing distributed systems much easier

When to Distribute Computing Responsibilities

- Information system management needs a corporate policy for deciding when the development, operation and maintenance of an application should be distributed
- Individual end users and departments should not be left on their own to make such decisions, especially where enterprise connectivity is important

When to Distribute Computing Responsibilities

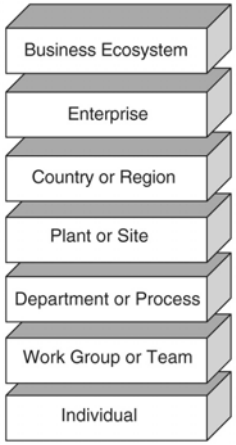
- Systems responsibilities can be distributed unless the following are true:
  - **Are the operations interdependent?**
    - When it is important that one operation knows what other is doing; their planning, development, resources, and operations must be centralized
  - **Are the businesses really homogenous?**
    - If they have a lot in common (e.g., In fast-food franchise: processing may be distributed, but planning and hardware selection should be centralized)
  - **Does the corporate culture support decentralization?**
    - Even if the business units do quite different things and don't need to know what each other is doing, corporate culture might still centralize finance, HR, and systems planning

Two Guiding Frameworks

- Distribute system via two guiding frameworks
  - An Organizational Framework
  - A Technical Framework

An Organizational Framework

- A driving force behind distributed processing is to give people more control over their work. This autonomy can happen at any of seven levels:
  1. Business ecosystem or value chain (inter-enterprise)
  2. Enterprise
  3. Country or region
  4. Site (plants, warehouses, branch offices)
  5. Department or process
  6. Work group or team
  7. Individuals



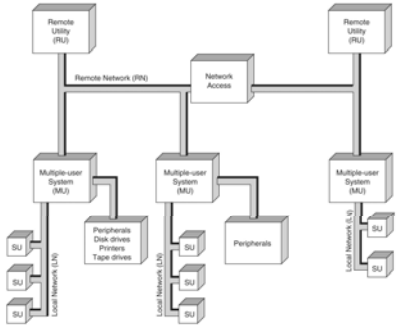
An Organizational Framework

- Locate processing power and database at each level in the organization
  - Top level (Business Ecosystem) deals with organizations that work closely together as buyer-seller, partner etc.
  - Next three levels (Enterprise, Region and Site) are traditional domain of IS, where computers resided in the past.
  - Bottom three levels (Department, Team and Individual) are where the bulk of employees are

Give autonomy and decision-making power to better serve customers

A Technical Framework

- Einar Stefferud, David Farber and Ralph Dement developed a conceptual framework for distributed system: **SUMURU** (**S**ingle **U**ser, **M**ultiple **U**ser, **R**emote **U**tility)
- Migration of computer power to end users will be driving force for network-based Information system.



A Technical Framework

- The components of the SUMURU architecture
  - Processors
  - Services
  - Networks
  - Standards

Processors	*Services*
Single-user systems (SU)	Terminal access
Multiple-user systems (MU)	File transfer
Remote utility systems (RU)	Computer mail
Networks	***Standards***
Local networks (LN)	Operating systems
Remote networks (RN)	Communications protocols
	Database systems

Components of SUMURU

- Processors:
  - **SU:** Single user, stand-alone and connected to LN (local network); clients
  - **MU:** Multiple user, serve local groups of users; server. Also heavy duty computation for SU, backups for MU, program libraries for SU, and database management.
  - **RU:** Remote utility, heavy-duty computing, corporate DB management, corporation mainframes and value-added network services

Components of SUMURU

- Networks:
  - **LN:** Local networks, high-speed information transfer, LAN
  - **RN:** Remote networks, lower transfer speeds, WAN, MAN, Internet

Components of SUMURU

- Services that this network architecture provides:
  - Access
  - File transfer
  - E-mail

Components of SUMURU

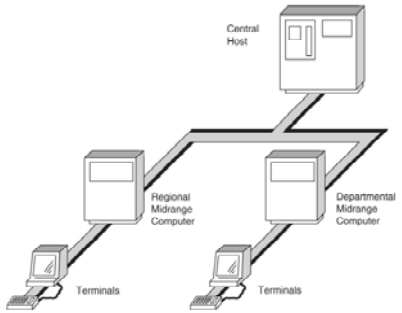
- Standards needed in three areas:
  - Operating System (OS)
  - Communication protocols: TCP/IP
  - Database Management System (DBMS): SQL

Seven Types of Distributed Systems

- The distributed system field has been continually evolving. The seven forms of distributed systems basically developed as follows:
  - Host-Based Hierarchy
  - Decentralized Stand-Alone Systems
  - Peer-to-Peer LAN-Based System
  - Hybrid Enterprisewide System
  - Client-Server Systems
  - Internet-Based Computing
  - Web Services

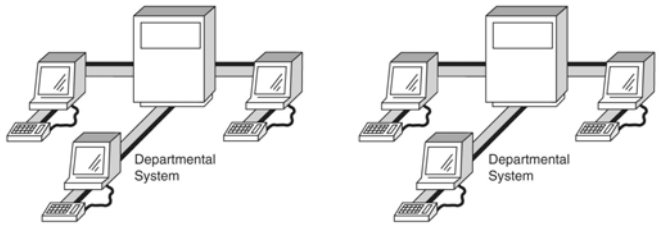
Host-Based Hierarchy

- First data processing distributed system. Host computer central, controlling component; terminals are access systems
- In Master-slave relationship: a central mainframe at the top, PCs at the bottom, minicomputers in between
- Data could be stored at any level



Decentralized Stand-Alone Systems

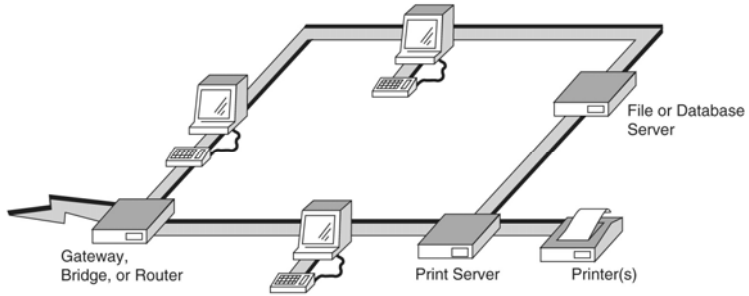
- Decentralized but does not really form a distributed system
- Holdover from the past where departments put in their own minicomputers with no intention of connecting them to the corporate host or to other departmental systems



A major goal in introducing ERP systems was to replace such disparate systems with a single platform of inter-connectable modules to serve these various functions

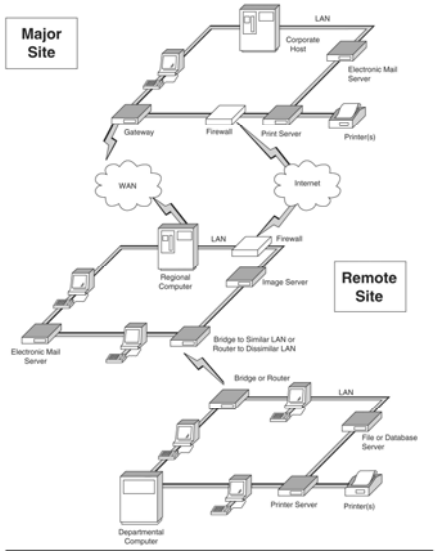
Peer-to-Peer LAN-Based Systems

- No hierarchy
- Peer-to-peer communications
  - Interconnecting LANs rather than hierarchical communications through a central hub
  - No superior computer



Hybrid Enterprisewide Systems

- Combination two types of distributed system via network (e.g. WAN)
  - Host-Based Hierarchy
    - Mainframe-based
    - Favored for corporate computing
  - Peer-to-Peer LAN-Based System
    - Favored by departments



Hybrid Enterprisewide Systems (cont')

- Allows company to link "Automation Islands" and retain IT investments, begin to automate business processes
- Such cooperating processes allow companies to take advantage of specialized computer programs, while at the same time extending the usefulness of some legacy systems
  - The process of pulling together such individual applications or components is called **Systems Integration**

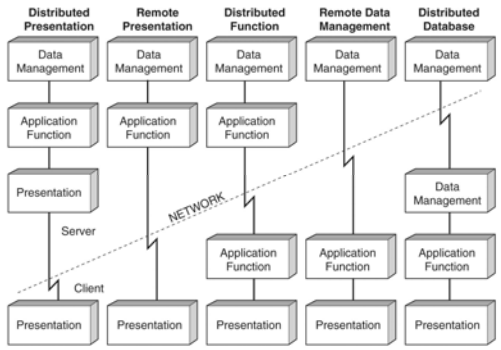
Read Case Example P.192-194

Client-Server Systems

- Arose to take advantage of the processing capabilities of both host machines and PCs in the same system
  - The computational flow of pure client-server systems is asymmetric.
  - Splitting work between clients and servers.
- Constraints on the use of the client-server style
  - Limit the number of clients that can be connected to a server
  - Impose a restriction that servers cannot interact with other servers.

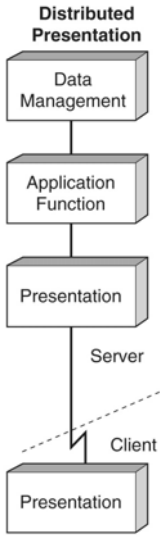
Client-Server Systems

- Three components being split in Client-Server Computing:
  - Presentation Software
  - Application
  - Data Management Software



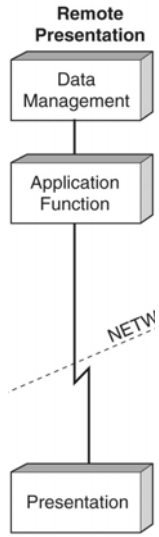
Client-Server Computing

- Distributed Presentation
  - Put all the data, all the application software, and some of the presentation software on server
  - Only part of the presentation is on the client



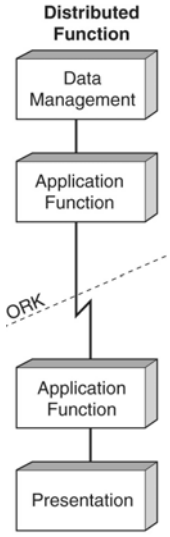
Client-Server Computing

- Remote Presentation
  - Most of the Web-based content systems today
    - Database sitting at the back
    - Web servers provide contents
    - Browsers display contents
  - Traditional GIS
    - Spatial data in database
    - GIS server processing data and rendering image
    - Clients display maps



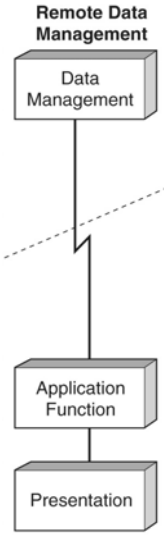
Client-Server Computing

- Distributed Function
  - Some new Web-GIS systems
    - Image rendering moved to the client
      - Reduce network flow
      - Load balancing
  - Most of the online-gaming systems



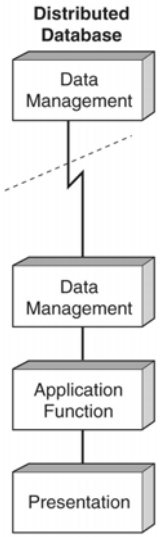
Client-Server Computing

- Remote Data Management
  - Traditional Client-Server database applications (Fat Client)
    - DBMS stores and retrieves data
    - Client applications process and display data
    - Client maintenance may be challenging



Client-Server Computing

- Distributed Database
  - Mobile computing
    - Less dynamic data stored locally
      - E.g. product category
    - Data synchronize when connected
  - Some applications in retail chains
    - Local data copy to improve service availability
      - E.g. Microsoft Access and Microsoft SQL Server



3-Tier Client-Server Architecture

- The preferred architecture has been three tiered:

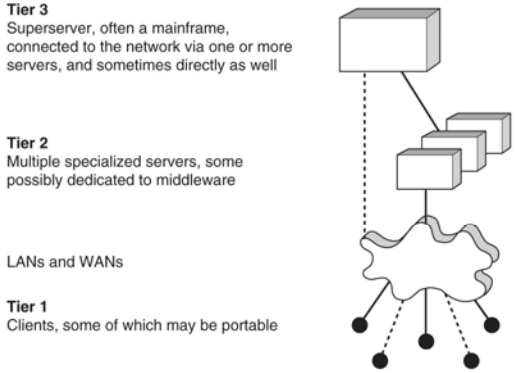


FIGURE 5-9 The Trend to Three-Tier Client-Server Arrangements

Source: Roger Woolfe, *Managing the Move to Client-Server*, Wentworth Research Program (now part of Gartner EXP, 56 Top Gallant, Stamford, CT 06904), January 1995.

Benefits of Client-Server Computing

- Better access to information
- Shift the focus of computing to user and empower employees
- Increases organizational flexibility:
  - Allows new technology to be added more easily without affecting rest of system
  - Streamlines work flow between functional areas
  - Encourages people to work together via networks
  - Supports new organizational structures via its connectivity

Drawbacks of Client-Server Computing

- Not lower in cost
- Easier for users, far more complex for Information System
- What looked like simple connections have turned into large, often fragile, complex systems

Read Case Example P.197-199

Internet-Based Computing

- In the late 1990s, the client-server trend was augmented by the Internet
  - The tenets of client-server remain
    - With Internet sitting at the heart
  - Ubiquitous computing
    - Various different platforms: Servers, PC, handhelds
  - A new computing environment
    - Irregular request arrival patterns
    - Unpredictable user numbers

Internet-Based Computing

- Network computers have not taken off desktops but the concept of utilizing programs off the Internet has
  - Network Computers (such as thin clients) coming into their own
  - Thin clients logical for hand held but now increasingly popular for the desktop
    - Updating new versions of software
    - Authorized software (firm and purchased)
    - One copy of software

Read Case Example P.201-203

Internet-Based Computing

- Server-Base Computing
  - With more use of laptops which do not have strong security features
    - Updating en masse is not easy
    - Even individual downloads can require helpdesk support
    - Applications reside on corporate servers rather than on laptops
    - Applications can be securely accessed by any device, they can be updated directly on the server, and they do not have to be tailored to run on specific machines

Read Case Example P.203

Internet-Based Computing

- Peer-to-Peer Computing
  - This form of Internet computing distributes a task over a wide number of computers connected to the Internet
  - This grassroots movement, like the open source movement, is now taken seriously by some corporations. It became famous with Napster, the music swapping P2P network
  - The main issue now is how to make money in this environment. One answer: subscriptions, where people pay for access rather than for ownership

Web Services

- This second-generation Internet-based distributed system gives software modules URLs (Internet addresses) so they can be called upon to perform their function as a service via the Internet
- This development will permit widespread computer-to-computer use of the Internet. One computer program or Web Service makes a request of another Web Service to perform its task (or set of tasks) and pass back the answer
  - **Wrapping** – Encapsulate functionality from an existing application in an XML envelope
  - **Exposing** – For use by others

Web Services Standards

- Three software standards:
  - XML (e**X**tensible **M**arkup **L**anguage)
  - WSDL (**W**eb Services **D**efinition **L**anguage)
  - UDDI (**U**niversal **D**iscovery, **D**escription, and **I**ntegration)
- Three communication standards
  - SOAP (**S**imple **O**bject **A**ccess **P**rotocol)
  - HTTP (**H**yper**T**ext **T**ransfer **P**rotocol)
  - TCP/IP (**T**ransmission **C**ontrol **P**rotocol / **I**nternet **P**rotocol)

Significance of Web Services

- Viewing IS as proprietary has led to rigid business processes, which are slow to change and respond to market changes
- Web Services offers an IT architecture based on the openness of the Internet. Rather than build proprietary systems, companies can obtain the functionality they need from the Internet
  - This modularity permits handling a huge variety of possibilities by mixing and matching, and allows easier cross-company system linking
  - Companies thus only pay for the functionality they use when they use it, which reduces the number of IT assets companies need to house and maintain

Read Case Example P.207-208

Defining the Overall IT Architecture

- The intent of an IT architecture is to bring order to the otherwise chaotic world of information systems by defining a set of guidelines and standards, and then adhering to them
- Because the architecture needs to support how the company operates, it reflects the business strategy
- As business changes, the architecture needs to keep pace

An Enterprise Architecture Framework

- Describe the information system architecture, look at the roles people and components play
- Use of the framework:
  - When IS users bring in a package that follows a data model inconsistent with the rules of the company, a lot will be spent fixing the package.

	Data (What)	Function (How)	Network (Where)
Scope Planner			
Enterprise Model Owner			
Information System Model Designer			
Technology Model Builder			
Components Subcontractor			
Functioning System Consumer or User			

Read Case Example P.212-214

An Enterprise Architecture Framework

- Rows: Views must be taken into account when building complex products:
  - Planner (scope statement)
  - Owner (model of the enterprise)
  - Designer (model of the information system)
  - Builder (technology model)
  - Subcontractor (description of the components)
  - User (functioning system)

An Enterprise Architecture Framework

- Columns:
  - Information system components: Represent physical manifestations of the system
    - Data models (what it is made of)
    - Functional models (how it works)
    - Network models (where the components are located)
  - Represent the soft side of system:
    - People (who)
    - Time (when)
    - Motivation (why)

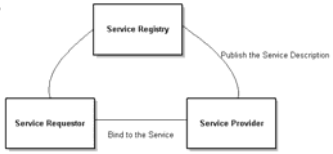
Enterprise Architecture Framework

	Data *What*	Function *How*	Network *Where*	People *Who*	Time *When*	Motivation *Why*
Objectives/ Scope (Contextual) *Planner*	List of things important to the business Entity = Class of business thing	List of processes the business performs Function = Class of business process	List of locations in which the business operates Node = Major business location	List of Organizations/agents important to the business Agent = Class of agent	List of events significant to the business Time = Major business event	List of business goals/ strategies Ends/Means = Major bus goal/ critical success factor
Enterprise Model (Conceptual) *Owner*	e.g., Semantic model Ent = Business entity Reln = Business relationship	e.g., Business process model Proc = Business process I/O = Business resources	e.g., Logistics network Node = Business location Link = Business linkage	e.g., Organization chart Agent = Organization unit Work = Work product	e.g., Master schedule Time = Business event Cycle = Business cycle	e.g., Business plan End = Business objective Means = Business strategy
System Model (Logical) *Designer*	e.g., Data model Ent = Data entity Reln = Data relationship	e.g., "Application architecture" Proc = Application function I/O = User views	e.g., Distributed system architecture Node = I/S function (processor, storage, etc.) Link = Line characteristics	e.g., Human interface architecture Agent = Role Work = Deliverable	e.g., Processing structure Time = System event Cycle = Processing cycle	e.g., Knowledge architecture Ends = Criterion Means = Business rules
Technology Model (Physical) *Builder*	e.g., Data design Ent = Segment/Row/etc Reln = Pointer/Key/etc	e.g., System design Proc = Computer function I/O =Screen/Device formats	e.g., System architecture Node = Hardware/System software Link = Line specifications	e.g., Human/technology interface Agent = User Work = Job	e.g., Control structure Time = Execute Cycle = Component cycle	e.g., Knowledge design Ends = Condition Means = Action
Detailed Represen-tations (out-of-context) *Sub-Contractor*	e.g., Data definition Ent = Field Reln = Address	e.g., Program Proc = Language stmt I/O = Control block	e.g., Network architecture Node = Addresses Link = Protocols	e.g., Security architecture Agent = Identity Work = "Transaction"	e.g., Timing definition Time = Interrupt Cycle = Machine cycle	e.g., Knowledge definition End = Subcondition Means = Step
Functioning System	e.g., Data	e.g., Function	e.g., Network	e.g., Organization	e.g., Schedule	e.g., Strategy

The coming Architecture: Service-Oriented Architecture (SOA)

- Move component interactions from hard-coded to dynamically discovered and invoked
  - By exposing the data and functions in a way that other systems can easily use
- Service-Oriented Architecture (SOA)
  - An **Architecture** is a formal description of a system, defining its components, their interrelationships, and the principles and guidelines governing their design and evolution over time
  - A **Service** is a software component that can be accessed via a network to provide functionality to a service requester
  - **SOA** is a style of building reliable distributed systems that deliver functionality as services, with the additional emphasis on loose coupling between interacting services

Read Case Example P.216-218



IT Infrastructure

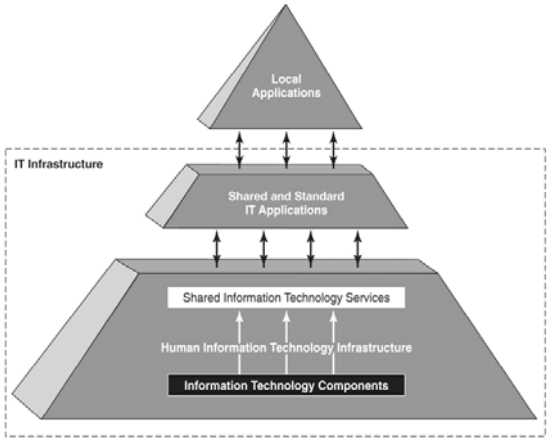
- The shared and reliable services that provide the foundation for the enterprise IT portfolio
  - Enabling business applications sitting above
  - Difficult to cost-justify
- Technologists and business people have different views of infrastructure
  - Peter Weill et al presented a model that meshes different views together
    - Gives technologists and business users a common language

The Importance of IT Infrastructure

- Infrastructure investments are a vital part of corporate information systems portfolios
- Yet they are the most difficult to cost-justify initially and to quantify benefits afterwards
- IT infrastructure is the foundation of an enterprise's IT portfolio
  - Provides the capability for reliable services and sharing
  - Includes both the technical and managerial expertise required to provide these services
  - Is linked to external industry infrastructure (e.g. Banking payments, airline reservations)
  - Shared characteristics differentiate an infrastructure from IT investments used by just one function

The Structure of the IT Infrastructure

- Weill and Broadbent divide the IT infrastructure into four layers



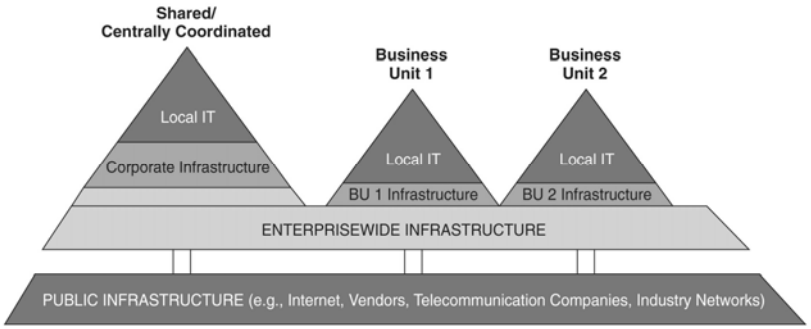
Four Layers of IT Infrastructure

- IT Components
  - Technology components, such as computers, DBMS packages, OS etc.
  - Generally not understood by business people
- Human IT infrastructure
  - Translation of the IT components layer into services
  - Experts' knowledge, skills, experience and standards
- Shared IT services
  - The infrastructure as a set of services that users can draw upon and share to conduct business.
- Shared and standard applications
  - Stable applications such as accounting and HR

IT Infrastructure

- Weill and Broadbent note that IT infrastructure sits on top of the public infrastructure.

FIGURE 5-16 IT Infrastructure



Source: Adapted from Peter Weill and Marianne Broadbent, *Leveraging the New Infrastructure: How Market Leaders Capitalize on IT* (Boston: Harvard Business School Press 1998).

Similar to Public Infrastructure

- Weill's model in turn sits on top of the public infrastructure
  - E.g. the Internet, vendors, telecomm companies.
- Similarity between IT infrastructure and public infrastructure such as roads, hospital, schools etc.
  - Everyone wants but no-one wants to pay
  - Provided by a central authority
    - Government or IT Department
    - Delicate and difficult investment balance

Three Views of Infrastructure

- **Utility**: Economies of scale
  - Infrastructure cost as an administrative expense
  - Minimize expense
  - Outsourcing may be viewed favorably because the IT infrastructure is not seen as strategic
- **Dependent**: Support for business programs
  - Infrastructure treated as business expense
  - Measured by short-term business benefits
  - Infrastructure planning in current business plan
- **Enabling**: Flexibility to meet changes in the marketplace
  - Primary benefit long-term flexibility
  - Intended to provide the foundation for changing direction in the future
  - IT cost seen as business investment

Read Case Example P.222-223