Defining the System Architecture I

CSIT883 System Analysis and Project Management

UNIVERSITY
OF WOLLONGONG
AUSTRALIA

Outline

Example of a Modern System Architecture

Architectural Concepts

Part I

Architectural Diagrams

Describing the Environment

Designing Application Components

Part II

Overview

- An important part of new system development is choosing appropriate technologies
- Explain and provide a summary of technology and architectural concepts
- describe the details for the activity – *Describe the Environment*
- Describe the details for the activity – *Design the application components*

Design activities

Describe the environment.
Design the application components.
Design user interface.
Design the database.
Design the software classes and methods.

Core processes	Iterations					
	1	2	3	4	5	6
Identify the problem and obtain approval.						
Plan and monitor the project.						
Discover and understand details.						
Design system components.						
Build, test, and integrate system components.						
Complete system tests and deploy the solution.						

Anatomy of a Modern System

- Architecture of Amazon.com

- **Server** – a computer or group of computers that manages shared resources such as file systems, databases, and Web sites, and enables users and other computers to access those resources over a network
  - Different from personal computing devices or clients (e.g., desktops, laptops, tablets, smartphones)
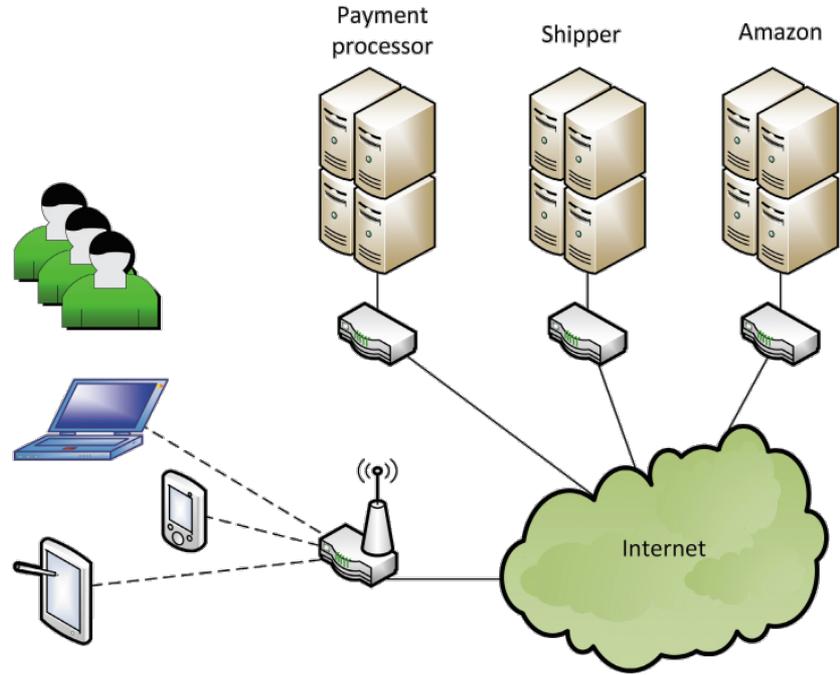
Computer Network

- **Computer network**: a collection of hardware, software, and transmission media that enables computing devices to communicate with one another and to share resources

- **Internet**: Constitutes networks of all sizes interconnect with one another

- **Local area network (LAN)**: a small computer network typically spanning a single home, small office, or one floor of a building

- **World Wide Web** (**WWW** or the Web): an interconnected set of resources accessed via the Internet
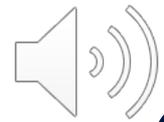
Computer Network

- **Uniform Resource Locator (URL)**: identifier of a Web resource containing a protocol header, server name or address, and resource name
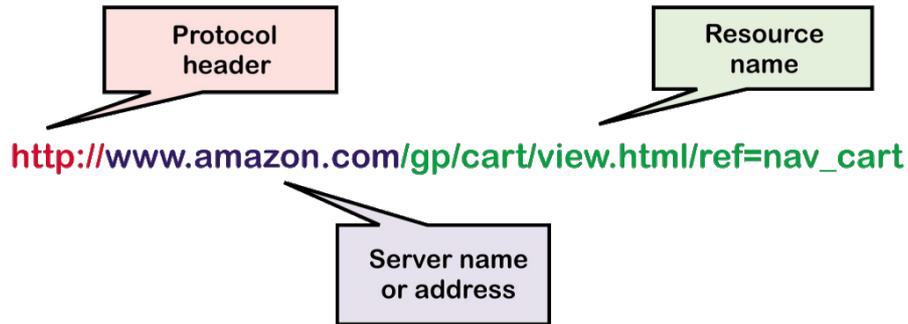

- **Hyperlink**: the URL of one Web resource is embedded within another Web resource

Software

- **Application software**: software that performs user- or business-specific tasks and is typically constructed as an app or Web-based application

- **App**: application software that is installed on the storage device of a computer or cell phone

- **System software**: software that works behind the scenes to support application software and control or interact with hardware or software resources
 - E.g. operating systems and Web server software
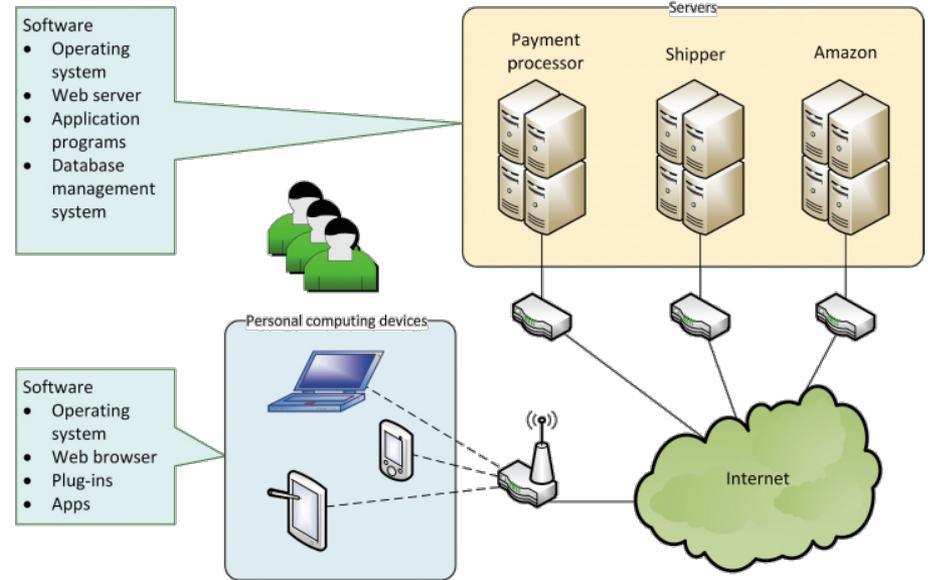
Software

- **Web-based application**: application software that deployed in the Web
  - uses a Web browser as the user interface
  - has a URL for application access
  - uses a Web server and server-side software components
  - uses Web standards for communication between Web browser and serve
- **Embedded software**: software apps or functions embedded within another app, such as within a browser or OS
  - Toolbars, Plug-ins, Widgets
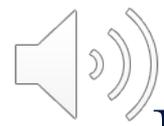
Protocols

- **Protocol**: a set of languages and rules to ensure communication and data exchange between hardware and software
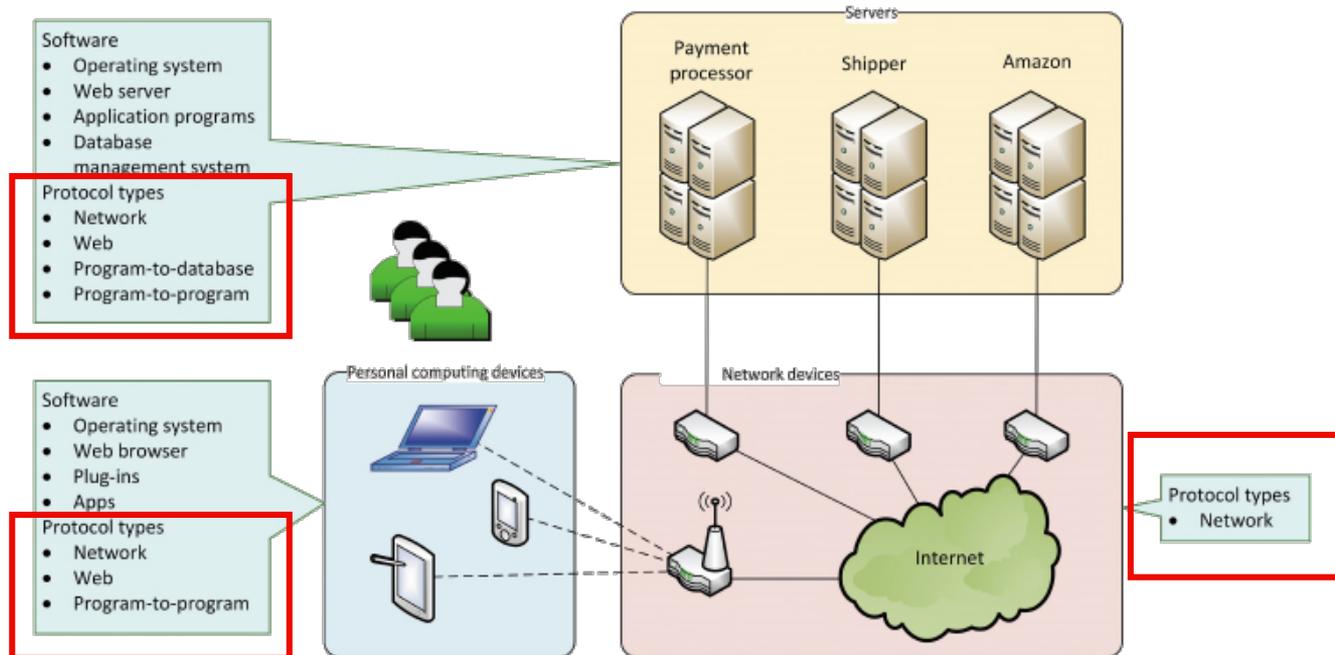
Protocols

- **Network protocol**: a protocol enabling accurate and secure message transmission among various computers and software components
    - Implemented within all computers and network hardware devices.

Protocols

- **Network protocol**: a protocol enabling accurate and secure message transmission among various computers and software components
    - Implemented within all computers and network hardware devices.

- **Web protocols** suite:
    - URL (Uniform Recourse Locator): defines a resource mane, location and encoding protocol
    - HTML (Hypertext Markup Language): defines the structure and context a Web page, including encoding methods for retrieving contents in various forms (e.g., text, tables, images and animation)
    - XML (Extensible Markup Language): defines extension to HTML that enables groups of users to define additional encoding with Web documents
    - HTTP (hypertext transfer protocol): defines format and content for transfer of Web documents
    - HTTPS (hypertext transfer protocol secure): encrypted and secure http transfers

Protocols

- **Network protocol**: a protocol enabling accurate and secure message transmission among various computers and software components
  - Implemented within all computers and network hardware devices.

- **Web protocols** suite:
  - URL (Uniform Recourse Locator): defines a resource mane, location and encoding protocol
  - HTML (Hypertext Markup Language): defines the structure and context a Web page, including encoding methods for retrieving contents in various forms (e.g., text, tables, images and animation)
  - XML (Extensible Markup Language): defines extension to HTML that enables groups of users to define additional encoding with Web documents
  - HTTP (hypertext transfer protocol): defines format and content for transfer of Web documents
  - HTTPS (hypertext transfer protocol secure): encrypted and secure http transfers

- **Application protocol**:
  - Program-to-Program (e.g., Remote Procedure Call (RPC))
  - Program-to-Database (e.g., Open Database Connectivity (ODBC))

Architecture

- **Technology architecture**
  - The set of computers, network hardware, and system software in an organisation

- **Application architecture**
  - The set of software applications and their configuration in an organisation
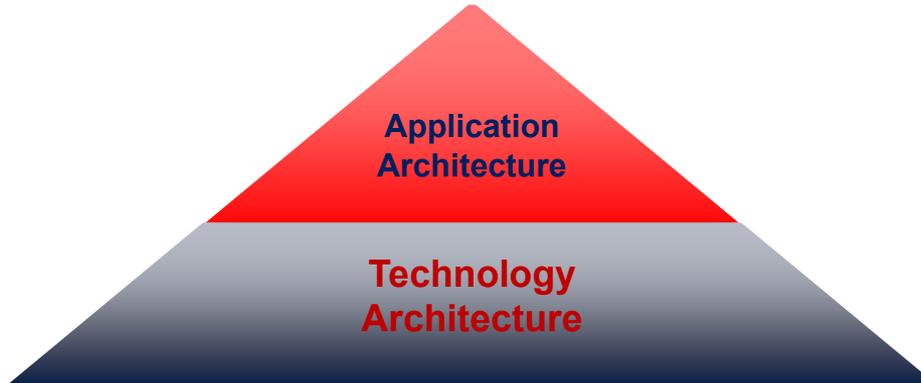
Services

- **Software-as-a-Service (SaaS)**: a software delivery model similar to utility, in which application software is accessed via the Internet without locally installed programs.
  - Little (or no) software is installed on users' devices
  - Application services is accessed remotely
  - User data is isolated and stored on common servers
- **Web service**: software function that is executed with Web standards
  - Access via a URL
  - Inputs sent via the URL
  - Executes remotely
  - Data returned within a Web page
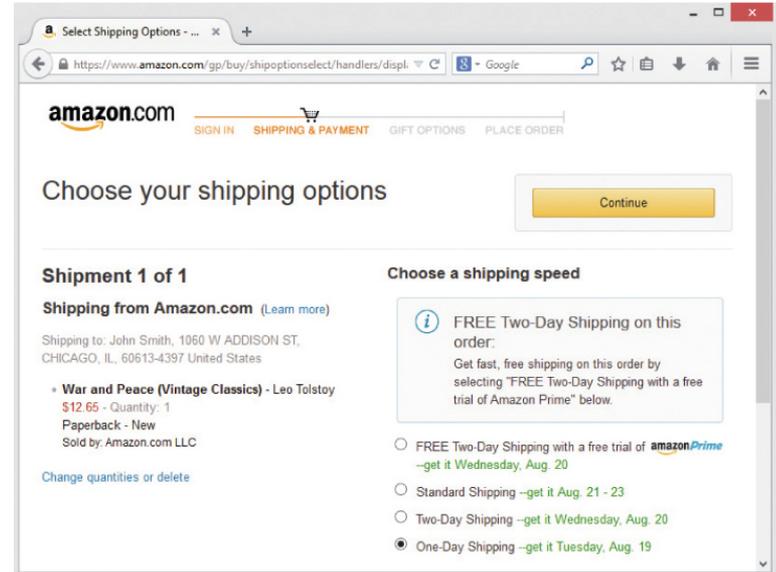
Distributed Architectures

- **Client/Server Architecture**: software design with part of the application on a server and part on the client
    - In the Client/Server Architecture, a server may refer to software (rather than a computing device).
    - A server manages system resources and provides access to these resources through a well-defined communication interface.
    - A client uses the communication interface to request resources and to get responses from the server.
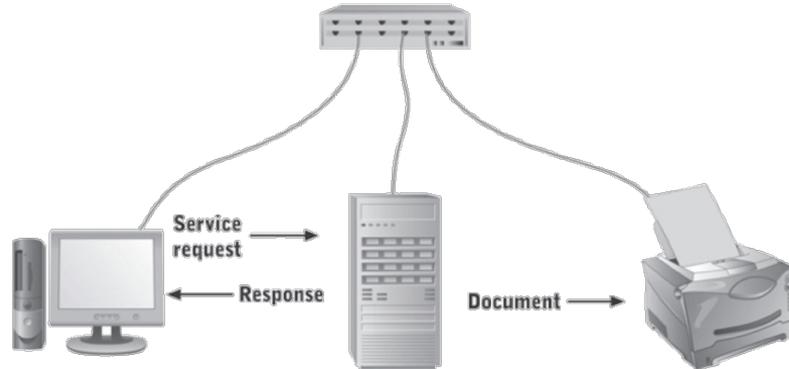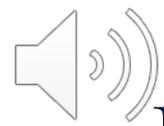
Distributed Architecture

- **Three-Layer architecture**
    - Client/server architecture with application divided into view layer, logic layer, and data layer
    - **View layer** – the user interface
    - **Business logic layer** – program logic to implement the functional requirements
    - **Data access layer** – the functions to access the data

Distributed Architecture

View layer		Software layers
		View layer logic—format screens/reports
Domain layer	Web/application server	Domain layer logic—implement business rules
		Data layer logic—formulate queries
Data layer	Database server	
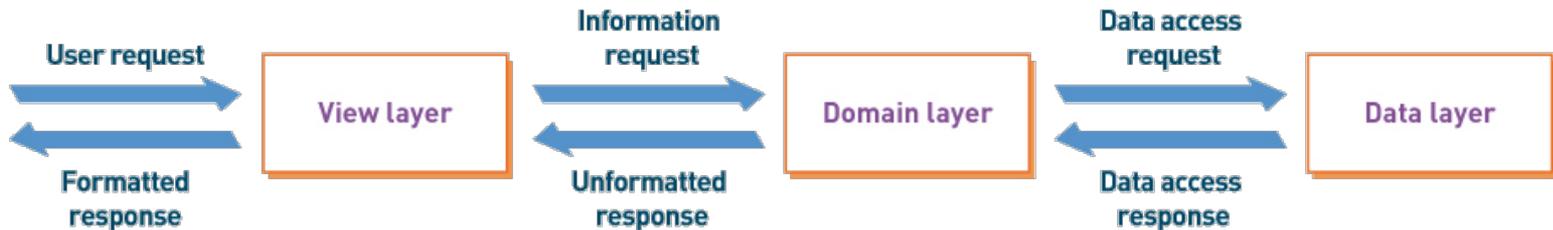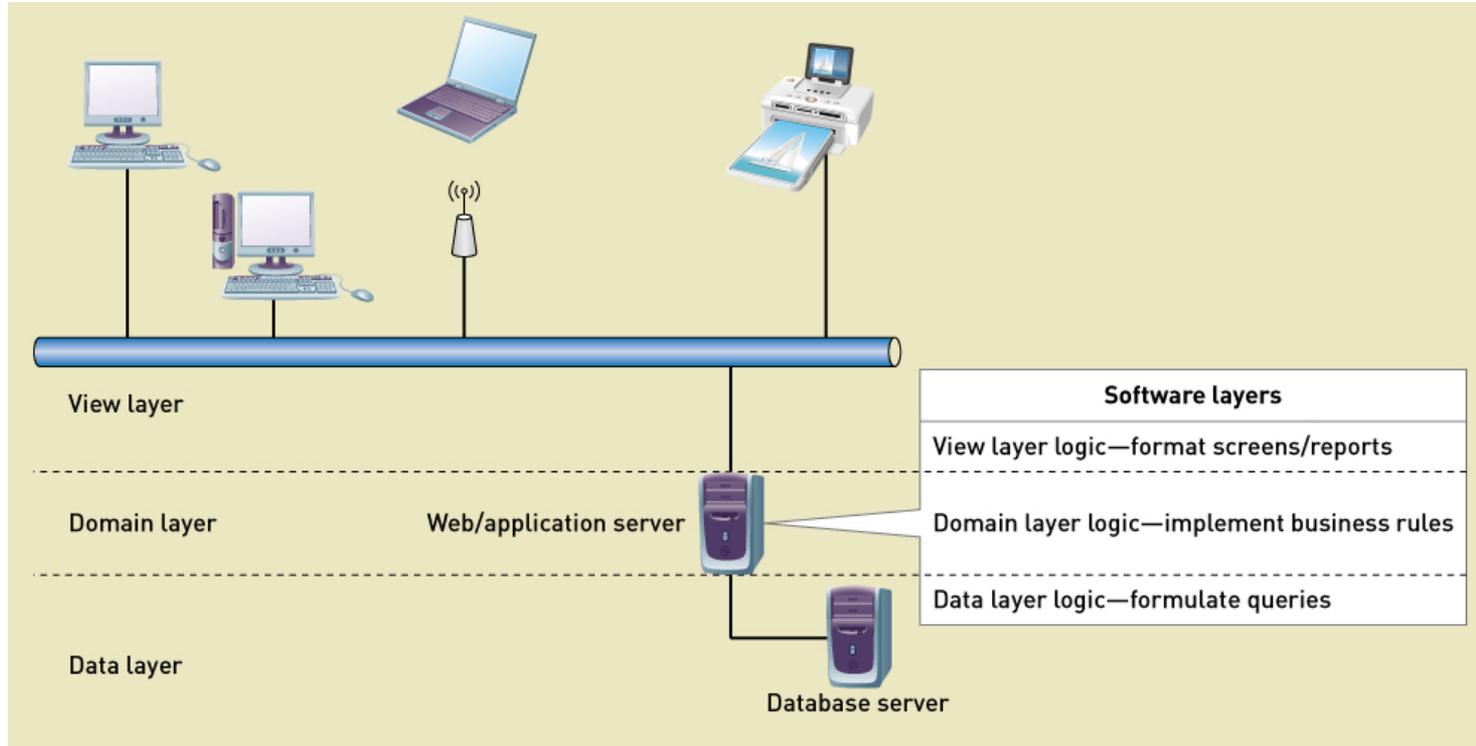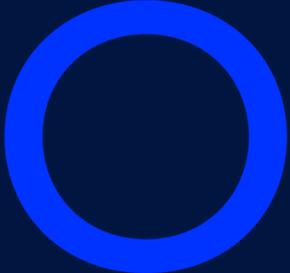
Defining the System Architecture II

CSIT883 System Analysis and Project Management

Outline

Example of a Modern System Architecture

Architectural Concepts

Part I

Architectural Diagrams

Describing the Environment

Designing Application Components

Part II
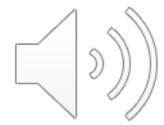
Architectural Diagrams

Location diagram

Network diagram

Deployment diagram

Architectural Diagrams

- Location Diagrams
 - Identify geographical placement of hardware, software, and users

Retail stores
Distribution warehouse centers
Manufacturing plants

Architectural Diagrams

- Network Diagrams
  - How the components of a network are interconnected
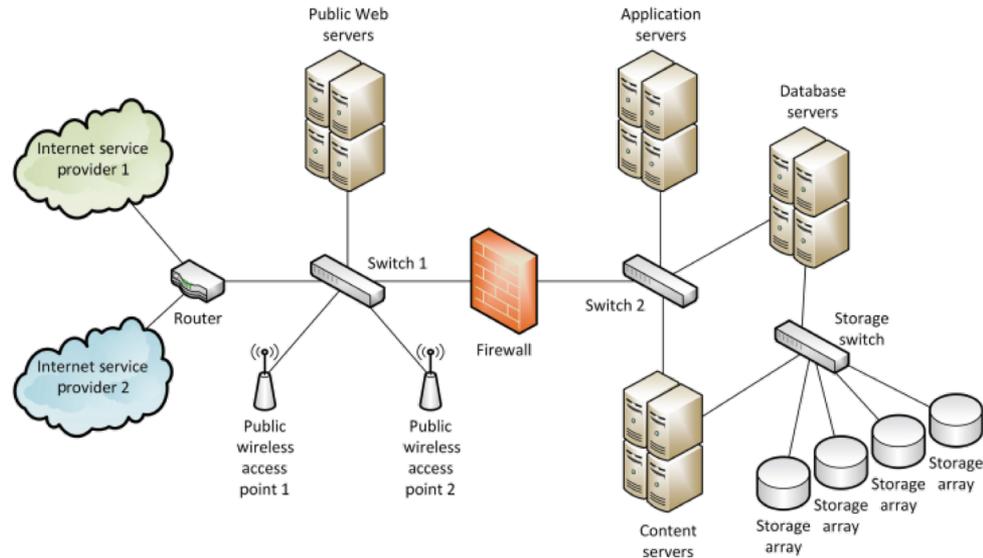
Architectural Diagrams

- Deployment Diagrams
  - How the application software is deployed across the hardware and system software
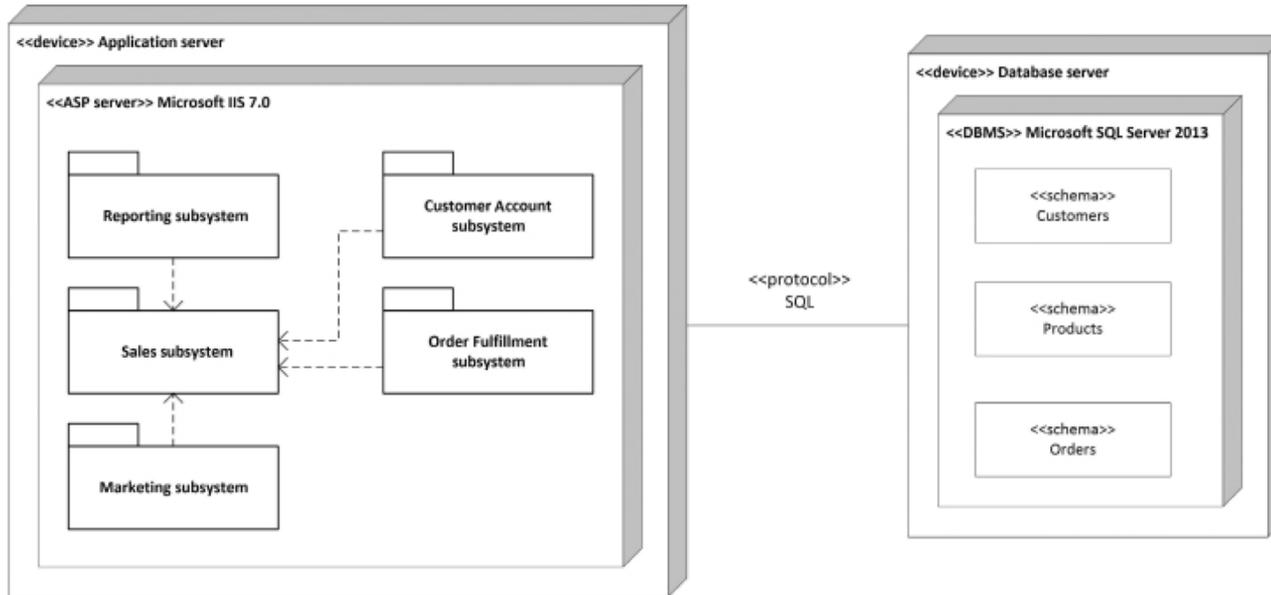
Describing the Environment

- Two key elements to be described: External systems and technology architecture
- Key Questions to help describe accurately
  - What are the key features of existing or new environment?
  - What are the external systems or DBMSs?
  - What devices will be required?
  - What user-interface technology will be used?

Designing Application Components

- Application Component Boundaries
  - Which components perform which functions?
  - Top-down or bottom-up (factoring or grouping)
  - How to measure similarity to build components?
    - Actors – what functions to particular actors use
    - Shared data – what functions use the same data
    - Events – what functions occur in common business events

RMO CSMS Application Components (partial)

Use case	User/actor	Domain class(es)	Event(s)	Group
Create phone sale	Phone sales representative	ProductItem, InventoryItem, SaleItem, Sale, SaleTrans	Customer request while shopping by phone	A
Create store sale	Store sales representative	ProductItem, InventoryItem, SaleItem, Sale, SaleTrans	Customer request while shopping in store	B
Create/update customer account	Customer, phone or store sales representative	Customer, Account, Address	Customer request or sale to a new customer	C
Look up order status	Shipping, customer, management, phone or store sales representative	ProductItem, InventoryItem, SaleItem, Sale, SaleTrans, Shipment, ReturnItem	Customer, representative, shipping, or management request	
Track shipment	Shipping, customer, management, phone or store sales representative	Shipment, Shipper, SaleItem	Customer, representative, shipping, or management request	
Create item return	Customer, phone or store sales representative	SaleItem, ReturnItem	Customer requests return	
Search for item	Customer, phone or store sales representative	ProductItem	Customer request while shopping online, by phone, or in store	
View product comments and ratings	Customer, phone or store sales representative	ProductItem, ProductComment	Customer request while shopping online, by phone, or in store	
View accessory combinations	Customer, phone or store sales representative	ProductItem, AcessoryPackage	Customer request while shopping online, by phone, or in store	
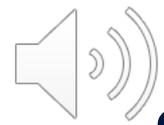
Summary

➢ Describing the system environment and designing its application components bridge the gap between systems analysis activities and detailed design activities (e.g., user-interface design, database design, and internal design of software components).

➢ Describing the system's environment requires describing its technical architecture and any interfaces with other systems.

➢ Designing the application components requires allocating system functions to components, describing how data is shared among those components, and describing how the application components are distributed across the technical architecture.

➢ Diagrams developed during these design activities include locations, network, and deployment diagrams.