

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.						

# Part III-System Design

## Foundations for System Designs

Dr. Yuehua Wang  
yuehua.wang@tamuc.edu



# Learning Objectives

---

- Describe systems design and contrast it with systems analysis
- List the documents and models used as inputs to or output from systems design
- Explain each major design activity
- Describe security methods and controls

# Overview

---

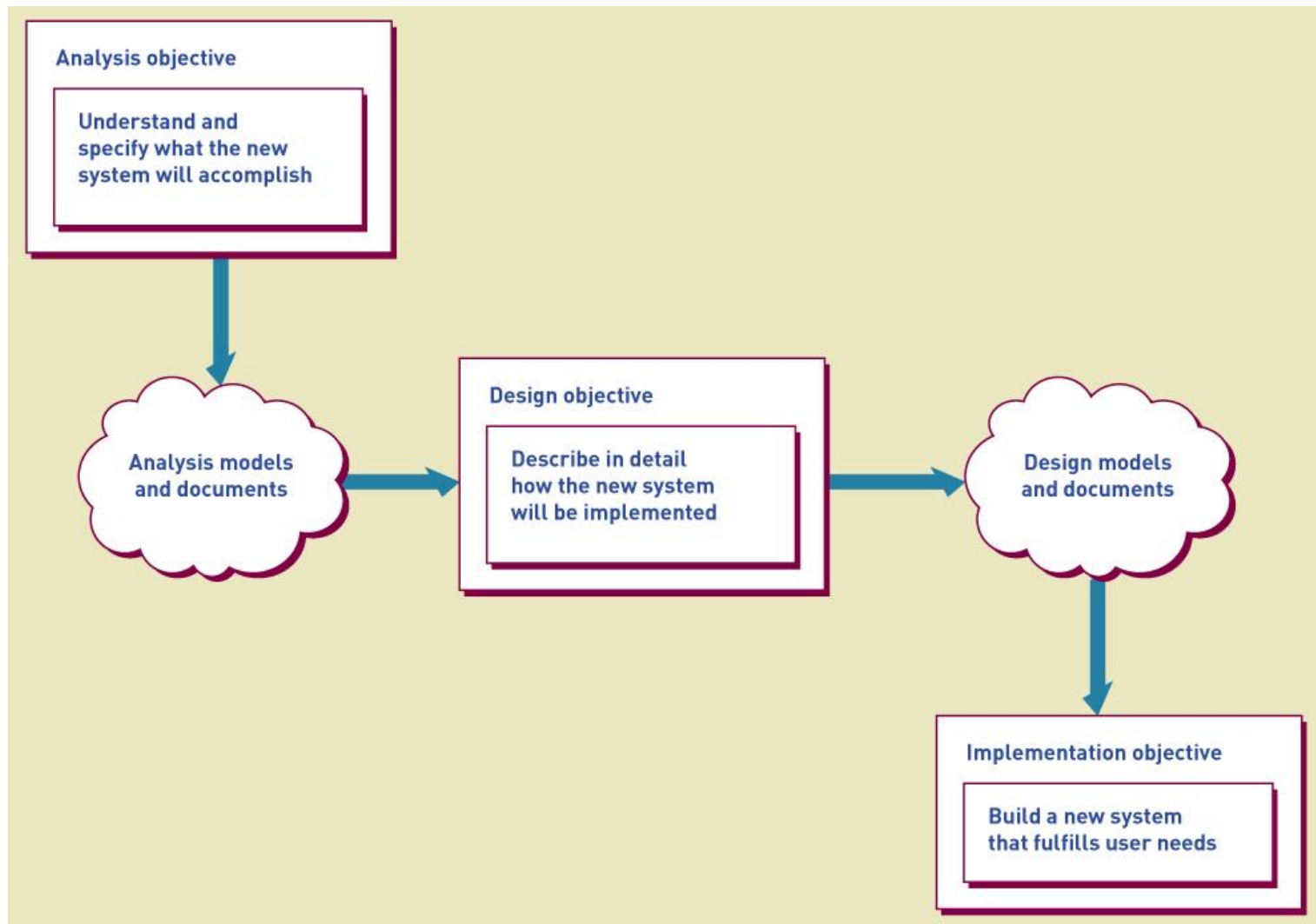
- Analysis says “what” is required and design tells us “how” the system will be configured and constructed
  - covered systems analysis activities (requirements)
- This lecture introduces system design and the design activities involved in systems development
- Design bridges the gap between requirements to actual implementation

# What is Systems Design

---

- Analysis provides the starting point for design
- Design provides the starting point for implementation
- Analysis and design results are documented to coordinate the work
- Objective of design is to define, organize, and structure the components of the final solution to serve as a blueprint for construction

# Analysis to Design to Implementation



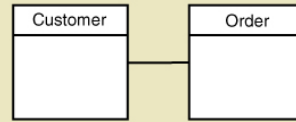
# Design Models

---

- Design is a model building activity
- The formality of the project will dictate the type, complexity, and depth of models
- Agile/iteration projects typically build fewer models, but models are still created
- Jumping to programming without design often causes less than optimum solutions and may require rework

# Analysis Models to Design Models

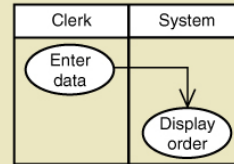
## Requirements models



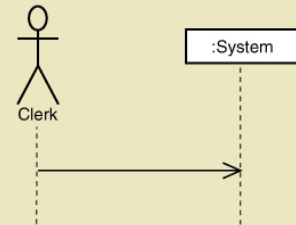
Domain model class diagram



Use case diagrams



Activity diagrams and use case description



System sequence diagrams

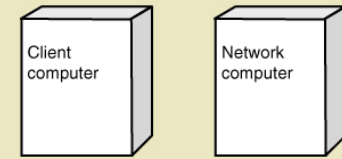


Requirements state machine diagrams

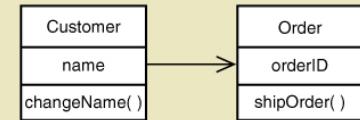
## Design models



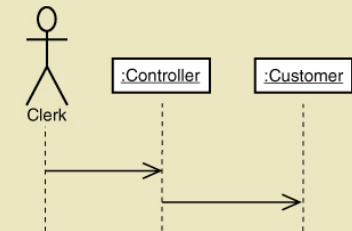
Component diagrams



Deployment diagrams



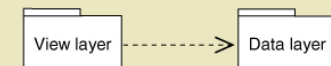
Design class diagrams



Interaction diagrams (sequence diagrams)



Design state machine diagrams



Package diagrams

# Design Activities

---













- Design activities correspond to components of the new system
  - The environment
  - Application components
  - User interface
  - Database
  - Software classes and methods



# Design Activities and Iterations

---

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.						

# Key Design Questions for each Activity

---

Design activity	Key question
Describe the environment	How will this system interact with other systems and with the organization's existing technologies?
Design the application components	What are the key parts of the information system and how will they interact when the system is deployed?
Design the user interface	How will users interact with the information system?
Design the database	How will data be captured, structured, and stored for later use by the information system?
Design the software classes and methods	What internal structure for each application component will ensure efficient construction, rapid deployment, and reliable operation?













# Describe the Environment

---

- Two key elements in the environment
  - Communications with External Systems
    - Precise message formats
    - Web or network of sources and destinations
    - Communication protocols
    - Security methods
    - Error detection and recovery
  - Conforming to an existing Technology Architecture
    - Discover and describe existing architecture

# Design Activities and Iterations

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

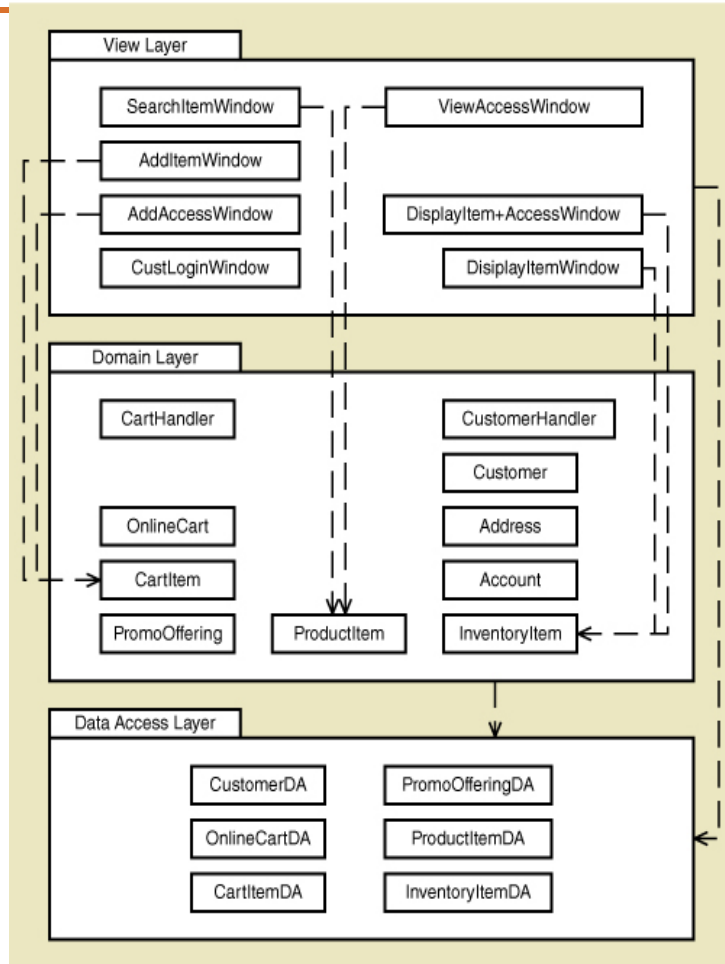
Core processes	Iterations					
	1	2	3	4	5	6
Identify the problem and obtain approval.						
Plan and monitor the project.						
Discover and understand details.						
Design system components.						
Build, test, and integrate system components.						
Complete system tests and deploy the solution.						

# Design the Application Components

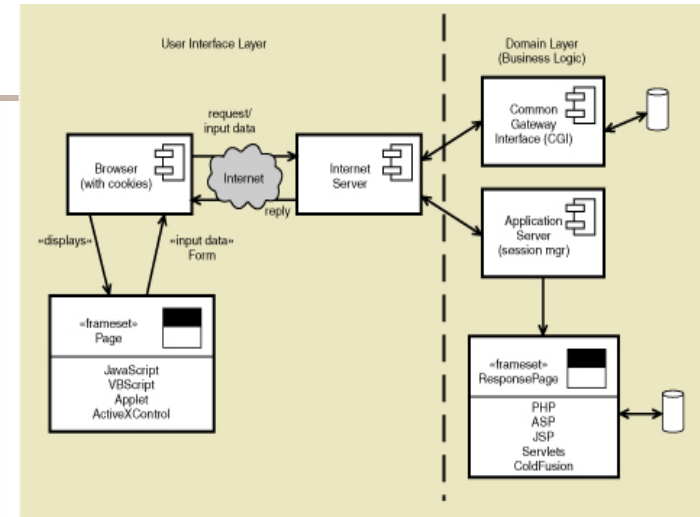
---

- Application component is a well-defined unit of software that performs one or more specific tasks.
- Though that definition sounds simple,
  - it hides complex details, including the following:
    - Scope and size – what are the functions, boundaries, interfaces?
    - Programming language – what are the accepted languages?
    - Build or buy – is an acceptable version available to purchase?
- The result of application component design is a set of models (diagrams), often supplemented with other documents that provide additional detail

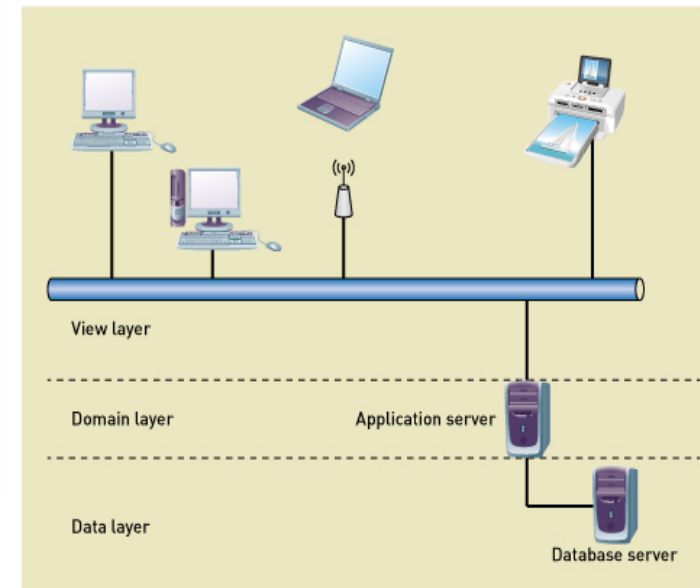
# Typical models for defining application components



Package diagram



Component diagram



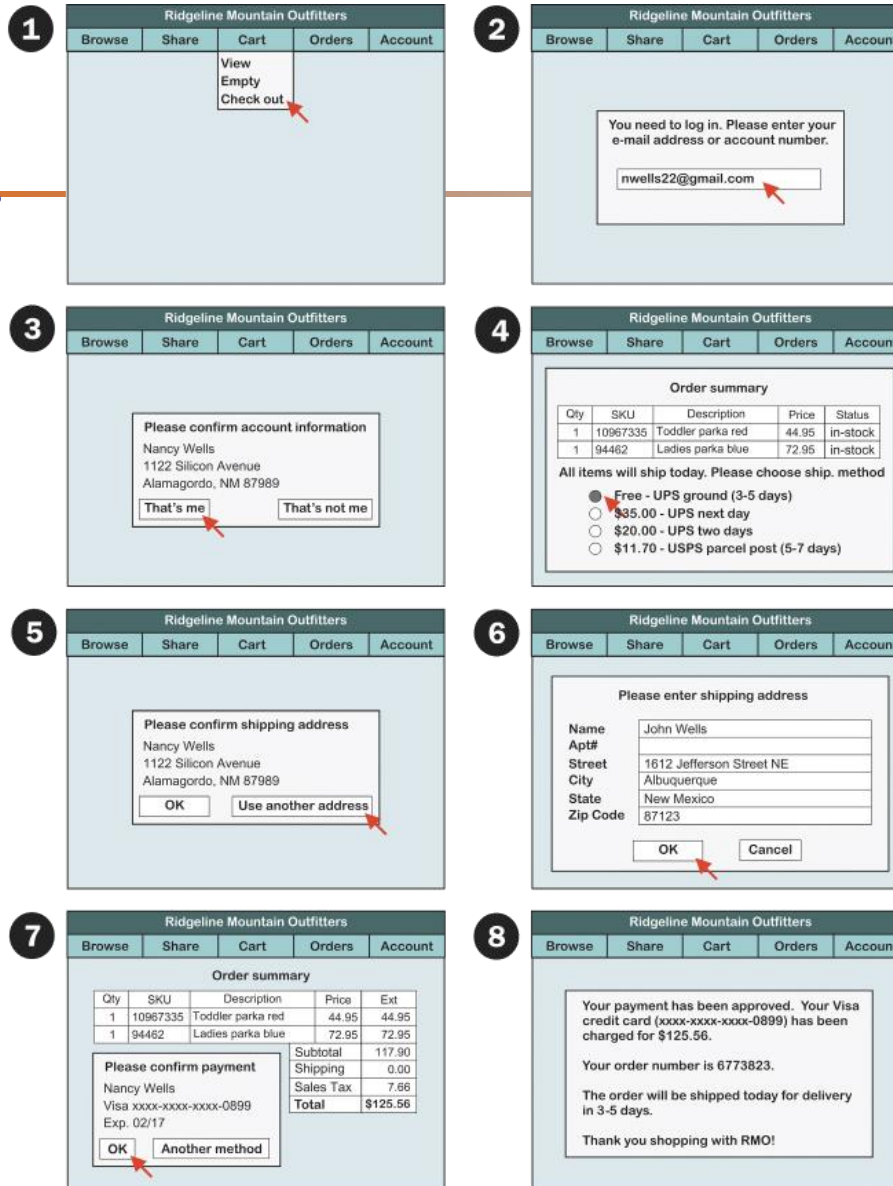
Deployment diagram

# Design the User Interface

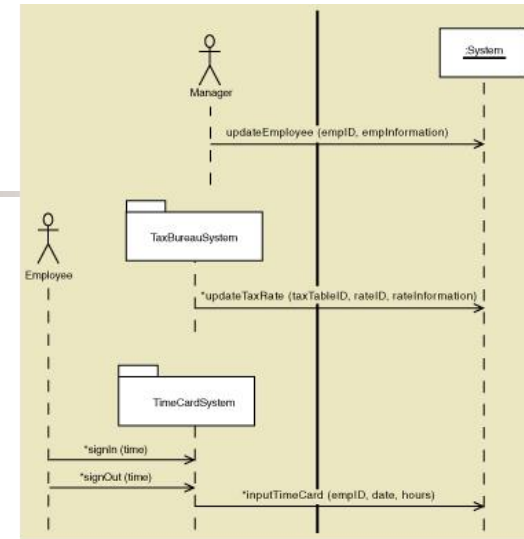
---

- To the user, the User Interface **is** the system.
- The user interface has large impact of user productivity
- Includes both Analysis and Design tasks
  - Requires heavy user involvement
- Current needs require multiple user interfaces
  - Many different devices and environments

# Typical models for user interface design



Storyboard



System sequence diagram



Small screen menu prototype

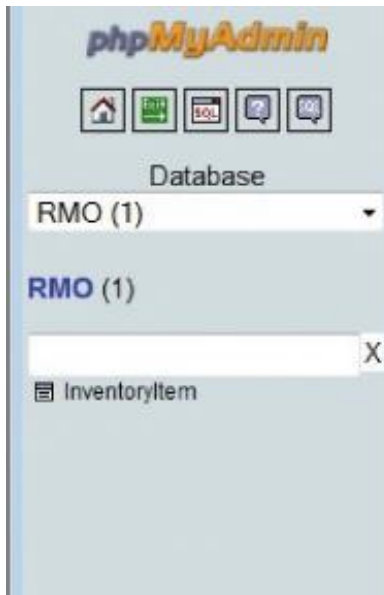


# Design the Database

---

- By definition, an Information System requires data – usually in a database
- Current technology frequently use Relational Database Management Systems (RDBMS)
- Requires converting the data model to a relational database
- Requires addressing of many other technical issues
  - Throughput and response time
  - Security

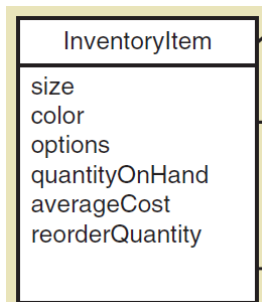
# Typical Table Definition as part of Database Schema



localhost ▶ RMO ▶ InventoryItem

[Browse](#) [Structure](#) [SQL](#) [Search](#) [Insert](#) [Export](#) [Import](#) [Operations](#)

	Field	Type	Collation	Attributes	Null	Default	Extra
<input type="checkbox"/>	<u>productItem</u>	varchar(15)	latin1_swedish_ci		No	None	
<input type="checkbox"/>	<u>inventoryItem</u>	mediumint(9)			No	None	
<input type="checkbox"/>	size	varchar(8)	latin1_swedish_ci		No	None	
<input type="checkbox"/>	color	varchar(10)	latin1_swedish_ci		No	None	
<input type="checkbox"/>	options	varchar(12)	latin1_swedish_ci		No	None	
<input type="checkbox"/>	quantityOnHand	mediumint(9)			No	None	
<input type="checkbox"/>	averageCost	decimal(8,2)			No	None	
<input type="checkbox"/>	reorderQuantity	mediumint(9)			No	None	
<input type="checkbox"/>	dateLastOrder	date			No	None	
<input type="checkbox"/>	dateLastShipment	date			No	None	

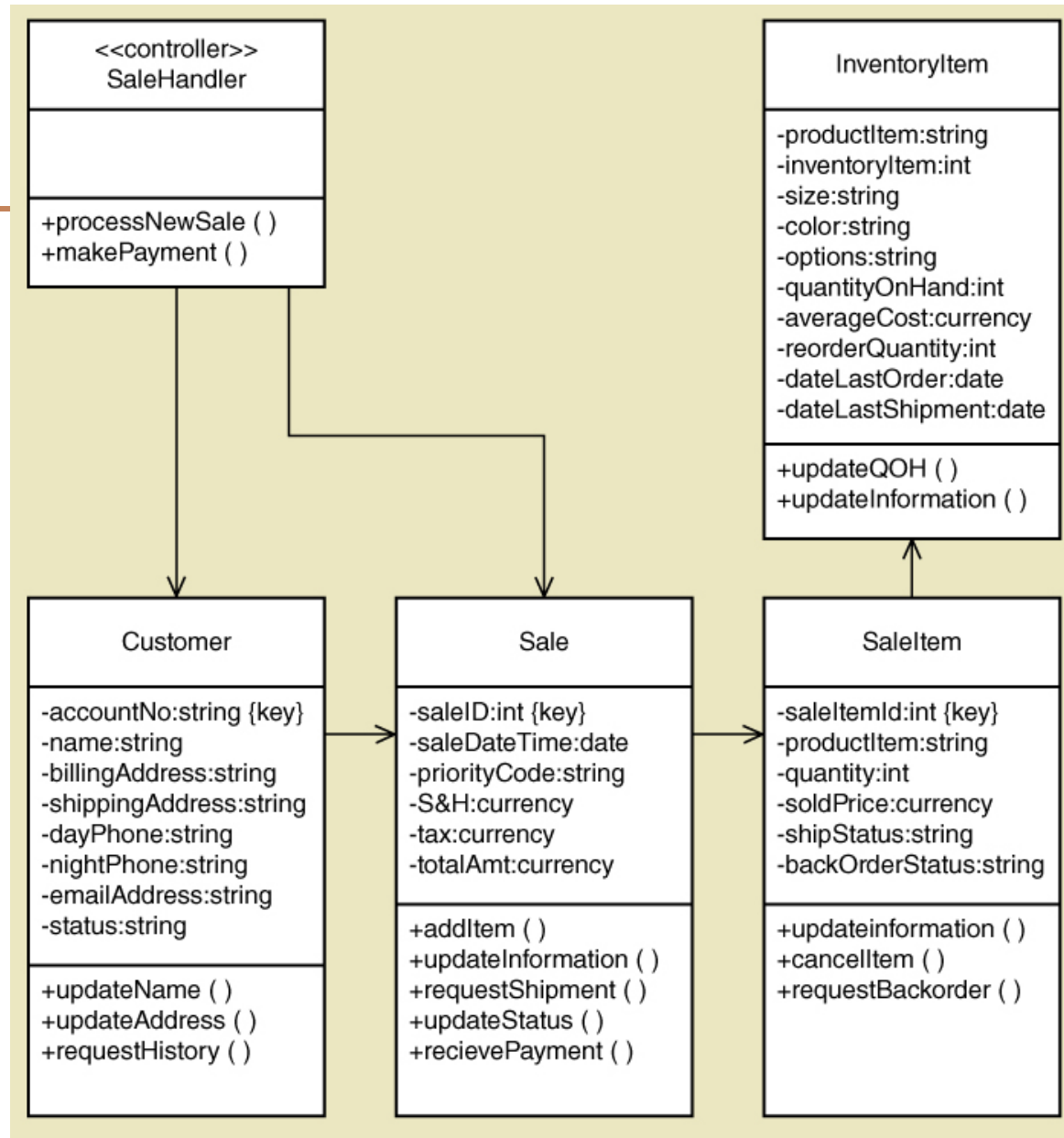


# Design Software Classes and Methods

---

- AKA Detailed Design
- The analysis models include
  - Design Class Diagram
  - Sequence Diagrams
  - State-Machine Diagrams
- Will be extended to incorporate software-specific elements
- Designers also create additional models such as sequence diagrams and class diagram with methods.

# Typical Design Class Diagram with attributes and methods



# System Controls and Security

---

- Modern systems are subject to a variety of risks, ranging from malfunction due to
  - incorrect data input,
  - inadvertently revealing confidential information,
  - and malicious destruction by outside parties.
- *Control* and *security* are blanket terms for the wide variety of ways that system designers and operators mitigate such risks

# Designing Integrity Controls

---

## ■ Controls

- are mechanisms and procedures that are built into a system to safeguard the system and the information within it.
- Here are a few scenarios that illustrate the need for controls:
  - A furniture store sells merchandise on credit with internal financing. Salespeople sometimes sell furniture on credit to friends and relatives. How do we ensure that only authorized employees can extend credit and record payments and adjustments to credit accounts?
  - An online retailer collects and stores credit card and other information about customers. How does the company ensure that customer data is protected and secure?

---

- Integrity controls

- controls that reject invalid data inputs, prevent unauthorized data outputs, and protect data and programs against accidental or malicious tampering
  - Controls that maintain integrity of inputs, outputs and data and programs
- must be integrated into both the application programs that are being developed and the database that supports them.

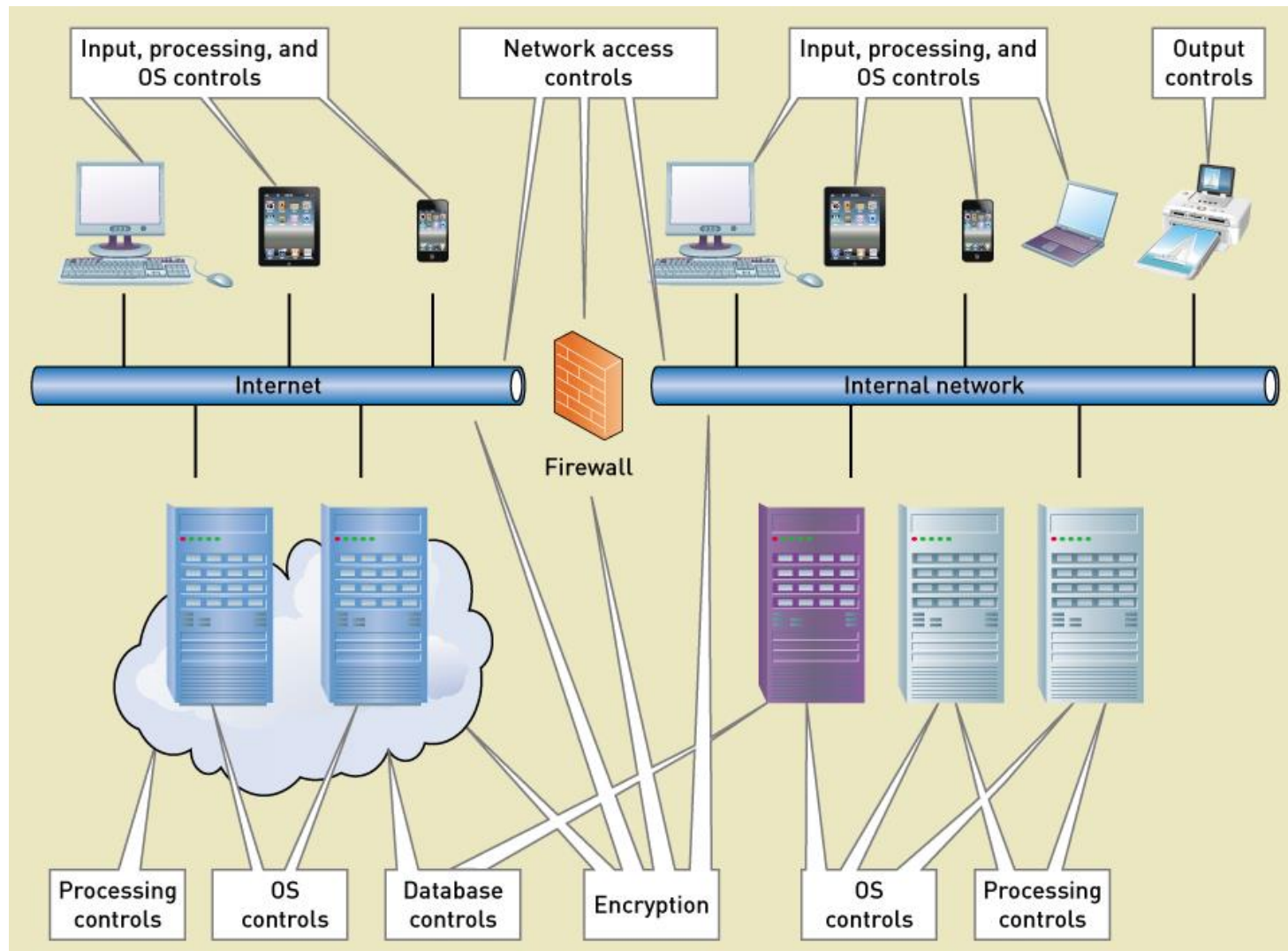
- Other controls—usually called security controls

- are part of the operating system and the network
- tend to be less application-specific
- Controls that protect the assets from threats, internal and external

- 
- The distinction between the two isn't precise because there is some overlap and because designers typically use both types.
  - No control type is sufficient by itself to achieve all the objectives; thus, many types of controls into their systems and databases are needed.



# Integrity and Security Controls



# Designing Integrity Controls

---

- Integrated into application programs and DBMS
- Objectives of Integrity Controls
  - Ensure that **only appropriate and correct business transactions** are accepted
  - Ensure that **transactions are recorded and processed correctly**
  - To **protect and safeguard assets** such as the database

# Designing Integrity Controls

---

- Integrity Controls

- Input Controls

- Prevent invalid or erroneous data from entering the system
    - Commonly used input control types include:
      - Value limit controls, Completeness controls, Data validation controls, and Field combination controls

- Output Controls

- Ensure that output arrives at proper destination (for authorized eyes) and is accurate, current, and complete
    - It is especially important that outputs with sensitive information arrive at the proper destination and that they cannot be accessed by unauthorized persons.

# Designing Integrity Controls

---

- Commonly used input control types include:
  - Physical access to printers and display devices
  - Discarded data – protect from “dumpster diving”
  - Labels on printed and electronic output to correctly identify source of data
- Redundancy, Backup, and Recovery
  - are designed to protect software and data from hardware failure and from such catastrophes as fire, flood, and malicious destruction.
  - Most operating systems and DBMSs incorporate support for all three.
  - On-site versus off-site copies
- Fraud Prevention

# Designing Integrity Controls

---

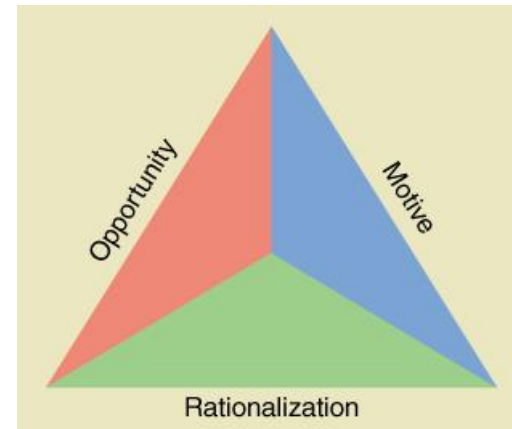
- Fraud Prevention

- should be aware of the fundamental elements that make fraud possible and incorporate controls to combat it.

- In the 1950s, fraud researchers developed a widely used model called the **fraud triangle**

- Require three elements:

- Opportunity
    - Motive
    - Rationalization



- System designers have little or no control over motive and rationalization, but they can minimize or eliminate opportunity by designing and implementing effective controls.

# Designing Security Controls

---

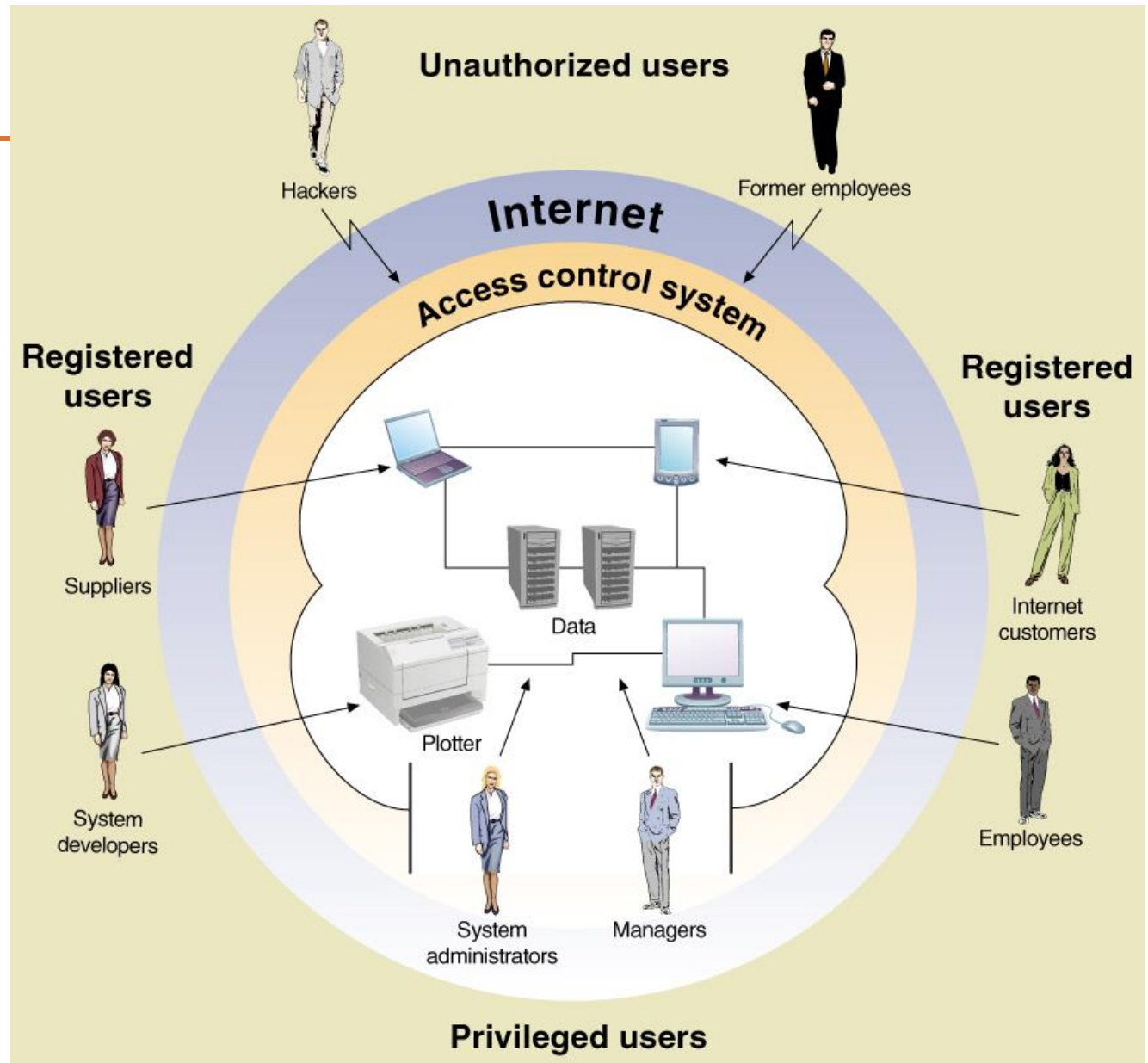
- Although the objective of **security controls** is to protect the assets of an organization from all threats, the primary focus is on external threats.
- Other objectives
  - Protect and maintain a stable, functioning operating environment 24/7 (equipment, operating systems, DBMSs)
  - Protect information and transactions during transmission across networks and Internet

# Designing Security Controls

---

- Access Controls – Limit a person's ability to access servers, files, data, applications, relay in these common elements.
  - Authentication – to identify users
    - Multifactor Authentication
  - Access control list – list of valid users
  - Authorization – authenticated user's list of permission level for each resource
- Categorize system users and determine what type(s) of access each resource requires.
  - Registered Users – those with authorization
  - Unauthorized Users – anyone not registered
  - Privileged Users – those that maintain lists and systems

## ■ Types of users





# Data Encryption

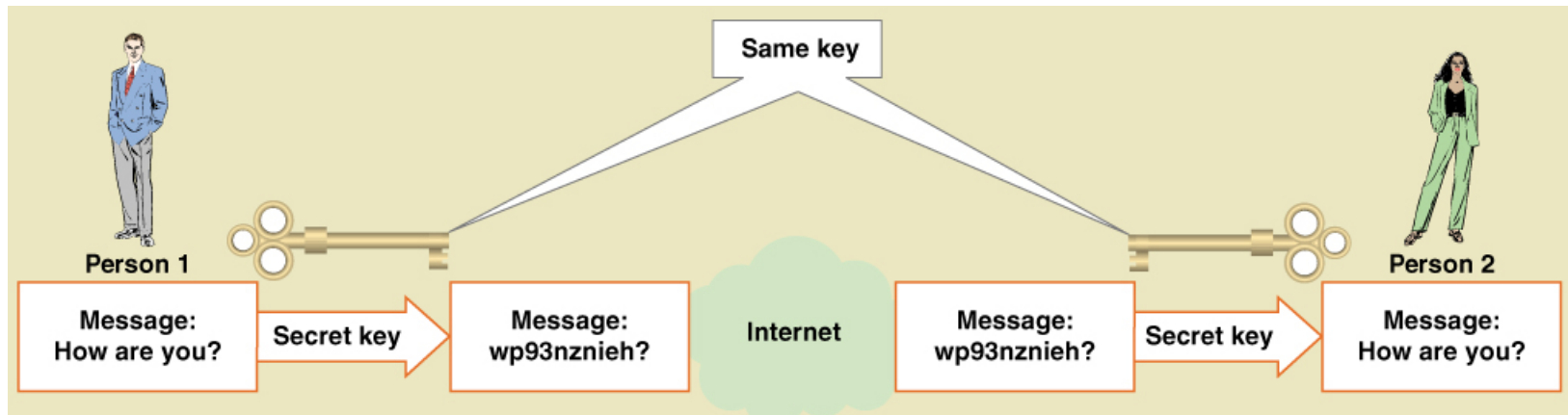
---



- No access control system is perfect, so designers must anticipate access control breaches and provide other measures to protect the confidentiality of data.
- Method to secure data – stored or in transmission
  - Encryption – alter data so it is unrecognizable
  - Decryption – converted encrypted data back to readable format
  - Encryption Algorithm – mathematical transformation of the data
  - Encryption Key – a long data string that allows the same algorithm to produce unique encryptions

# Symmetric Key Encryption

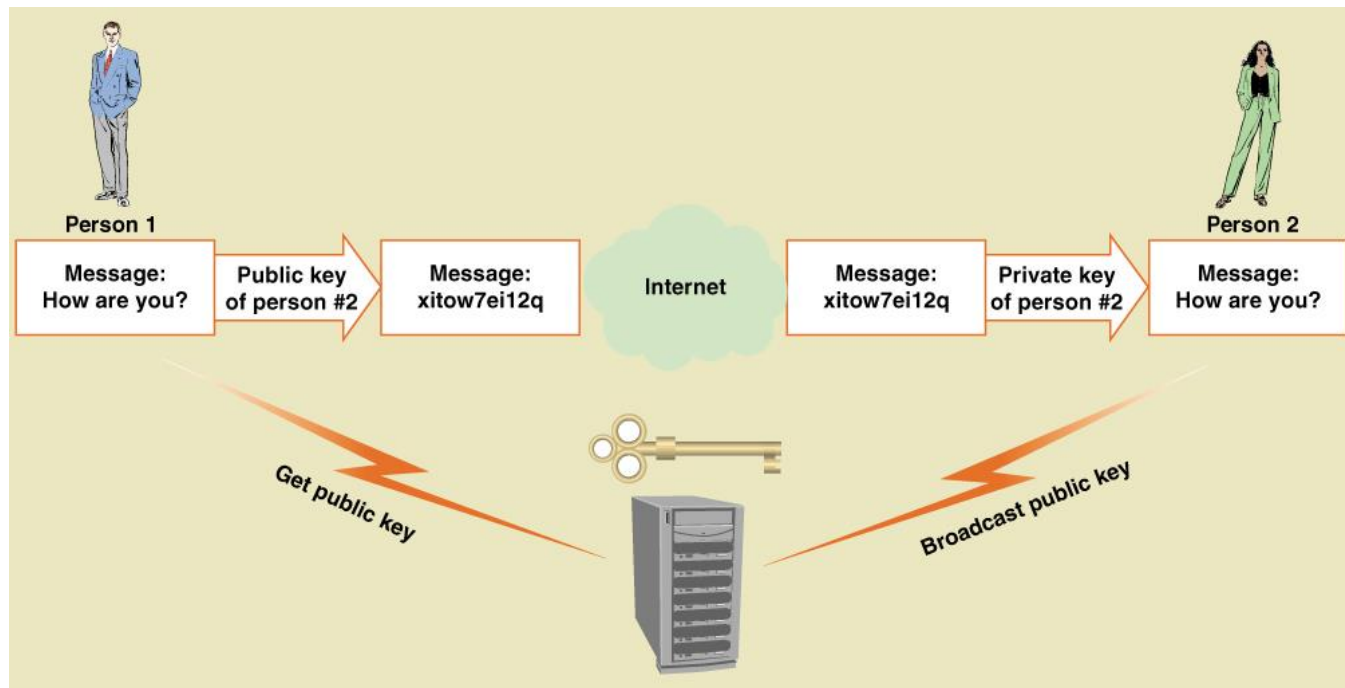
- Encryption method that uses the same key to encrypt and decrypt



- the same key must be created and shared securely.
  - Security is compromised if the key is transmitted over the same channel as messages encrypted with the key.
  - sharing a key among many users increases the possibility of key theft.

# Asymmetric Key Encryption

- Encryption method that uses different keys to encrypt and decrypt
  - AKA Public Key Encryption



---

How do you know that the entity on the  
other end of the communication is really  
who you think it is?

## Digital Signatures



### ■ Digital signature

- a technique in which a message or document is encrypted with a private key and decrypted with the public key.

### ■ Digital certificate

- is an institution's name and public key (plus other information, such as address, Web site URL, and validity date of the certificate), encrypted and certified by a third party.
- <https://www.youtube.com/watch?v=stsWa9A3sOM>



## Client

3. Client verifies certificate signer is a trusted certifying authority and authenticates server.

4. Client generates a secret key to be used for the session and encrypts it with the server's public key.

1. Client sends request to connect to secure server.

2. Server sends signed digital certificate (containing server's public key).

5. Client sends encrypted secret session key.

7. Client and server communicate securely using the secret session key.



## Secure server

6. Server uses its private key to decrypt secret session key.

# Secure Transactions

---

- Secure electronic transactions require a standard set of methods and protocols that address authentication, authorization, privacy, and integrity.
  - Secure Sockets Layer (SSL) – standard set of protocols for authentication and authorization
  - Transport Layer Security (TLS) – an Internet standard equivalent to SSL
  - IP Security (IPSec) – Internet security protocol at a low-level transmission
  - Hypertext Transfer Protocol Secure (HTTPS) – Internet standard to transmit Web pages

# Summary

---

- introduces the concept of Systems Design
  - Analysis is fact finding and modeling
  - Design is modeling to specify how system will be implemented
  - Design is bridge between analysis and implementation
- Activities of Systems Design
  - Describe the environment
  - Design the application components
  - Design the User Interface
  - Design the database
  - Design the software classes and methods



---

## ■ System Controls and Security

### ■ Integrity Controls

- Input controls
- Output controls
- Backup and recovery
- Fraud prevention

### ■ Security Controls

- Access controls
- Data encryption
- Digital signatures and certificates
- Secure transactions

A green rectangular road sign with rounded corners and a white border, mounted on two wooden posts. The word "Ethics" is written in large, white, sans-serif capital letters. The sign is tilted slightly to the right. The background is a bright blue sky with scattered white clouds.

**Ethics**

# A question of ethics

---



- Sally works as a junior analyst for a medium-sized IT consulting firm. Her manager, Bob, has asked her to draft a response to an RFP from a large company that is seeking IT consulting services in connection with a new accounting system.
- As Sally worked on the RFP, she noticed a specific question about her firm's recent experience on this type of system. To the best of her knowledge, the firm has only worked on one other accounting project in the last three years.



- 
- When Bob saw Sally's draft response, he was upset about the way she answered the question. "You don't have to be quite that candid," he said. "Even though we only had one formal project, we do have several people who worked on accounting systems before they came here."
  - "Yes," Sally replied, "But that isn't what the question is asking." As he left her office, Bob's final comment was, "If we want that job, we'll have to come up with a better answer." Thinking about it, Sally isn't comfortable with anything but a straight answer. **Is this an ethical question? What are Sally's options?**