System Dev Exam Notes

Systems Development (Monash University)

Week 1 – Introduction to Project Management

Software Development

Computer Application

- Computer software program that executes on a computing device to carry out specific set of functions
  - Modest scope

Information System

- Set of interrelated components that collects, processes, stores and provides as output the information needed to complete business tasks
  - Broader in scope that "app"
  - Included databases and related manual processes

What are Information Systems?

- Integrated set of components for colleting, storing and processing data for delivery of information
- Main component of an info sys
  - People
  - Procedures
  - Hardware and software
  - Databases
  - Data warehouses
  - Telecommunications

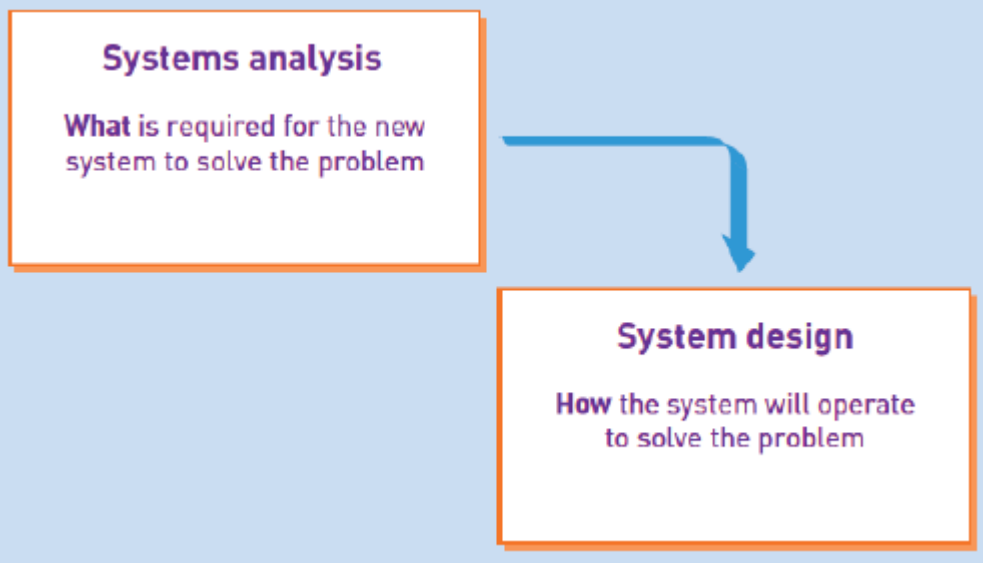
Customer Relationship Management (CRM) system

System Analysis

- Those activities that enable a person to *understand and specify* what an information system should accomplish

System Design

- Those activities that enable a person to *define and describe in detail* the system that solves the needs



System Development Life Cycle (SDLC)

- The process consisting of all activities required to; build, launch and maintain an info sys
- Six core process are:
  1. Identify problem or need and obtain approval
  2. Plan and monitor the project
  3. Discover and understand details of problem or need
  4. Design the system components
  5. Build, test and integrate system components
  6. Complete system tests and deploy solution

Project

- Planned undertaking that has a beginning and end and that produces some definite result
  - Used to develop an info sys
  - Requires knowledge of system analysis and design tools and techniques

System Development Process

- Actual approach used to develop a particular info sys
  - Unified process (UP)
  - Extreme Programming (XP)
  - Scrum

Iterative Development

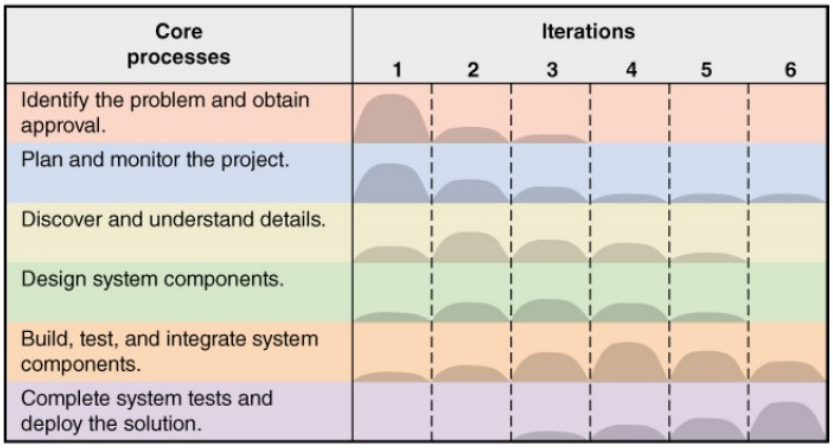
- Approach to Sys Dev in which the system **grows in piece by piece through multiple iterations**

- Complete small parts of system, then repeat processes to refine and add more, the repeat to refine and add more, until done

Agile Development

- An info sys dev process that **emphasizes flexibility to anticipate new requirements during development**
  - Fast on feed; responsive to change

Iterative and Agile Systems Development Lifecycle(SDLC)



Processes

1. **Initiation: Feasibility - Can it be done?**

- Can you afford to build what you want?
- Time constraints?
- Expertise available
- Willing to compromise
- Mandatory vs option
- Time + Budget

Initiation - Planning your development project

- Scope will stay fixed?
- How to manage scope creep?
- Project planning

2. **Analysis: What do you want?**

- Does the client know what they want?
  - Determines how you go about the process
  - It is vital that you demonstrate to client that you understand his/her requirement
- E.g.
  - Client requirement: Modern 4 bedroom house with 2 toilets and a garage
    - Which house does the client want?
    - The house together with a 1000 other houses would meet the brief?

Analysis - Build or Buy?

- Do you have to build or can you just buy

3. **Design: How are you going to do it?**

- Detailed plans for the build
  - Shows integration of various components
    - Plans for carpenters, electricians, plumbers

4. **Implement: Build/Develop - Construct, Test**

- Good analysis and design is essential for good build
- Good together with building expertise and thorough testing
- Not however just through building expertise only

Implement - Deploy: Is it Ready? Can I move in now?

- Does it meet
  - Government requirements
  - Sustainability requirements
  - CLIENT requirements
- Are your clients happy?
- Very costly exercise if the requirements are not met

5. **Support: Maintain it, Extend it**

- Can it be easily maintained and fixed?
- Can it be added to easily?

System Development Roles

Managerial - PM + TL
Functional - System + Busienss Analyst, Tester, UX Designer
Technical - System Designer, Database Admin,
Other - Quality Assurance, Documentation, Training and Deployment

System Developer Role

- **Understanding Business**
  - Awareness and sensitivity to the business processes and needs that require technology in the first place
- **Broad and up - to - date understanding of technology**
  - Can be invaluable in creating the "best" solution for the organisation
- **Multiple Perspective**
  - Ability to understand that there are multiple perspectives to solving problems is required to find the best solution
- **People/Soft Skills**
  - Ability to interact with other people and to be a part of a team
- **Continuous Learning**
  - Essential to a high change industry, like IT

Week 2 – Approaches To Systems Development

SDLC

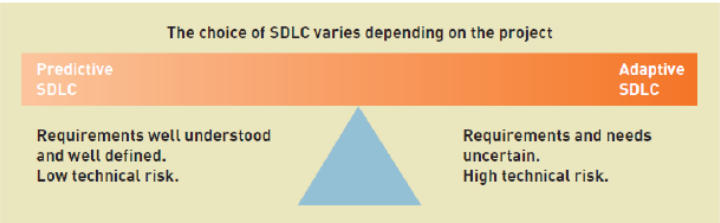
- Two general approaches

- Predictive Approach
  a. Waterfall model
  b. Assume project can be planned in advance + info system can be developed according to the plan
  c. Requirements are well understood and/or low technical risk

- Adaptive approach
  a. Iterative model
  b. Assume project is flexible and adapt to changing needs as project progresses
  c. Requirements and needs are uncertain and/or high technical risk

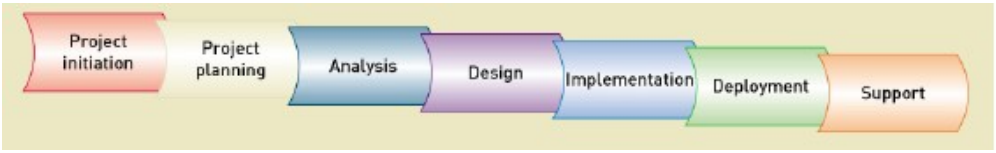
The System Development Life Cycle (SDLC)

- Most projects fall on a continuum between Predictive and Adaptive

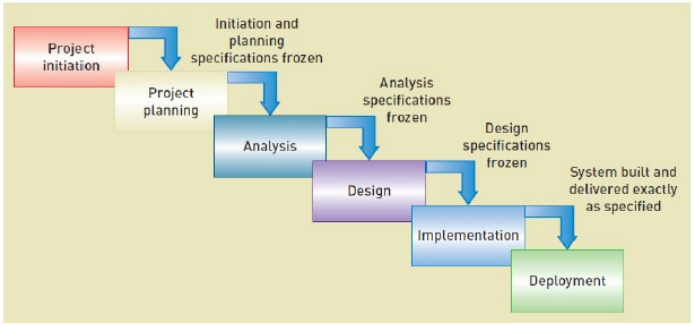


Traditional Predictive SDLC

- Earlier approach based on engineering
- Typically have sequential *phases*
  a. Phases are related groups of development activities, such as planning, analysis designing, implementation, and deployment
  - Waterfall Model
    a. SDLC that assumes phases can be completed sequentially with no overlap or iteration
    b. Once phase is completed, you fall over the waterfall to next phase, no going back
  -

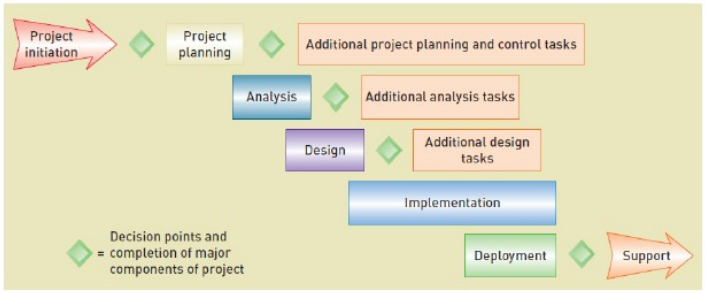


Waterfall Predictive SDLC



Modified Waterfall with Overlapping Phases

- More flexibility, but still assumes predictive planning and sequential phases



Newer Adaptive SDLC

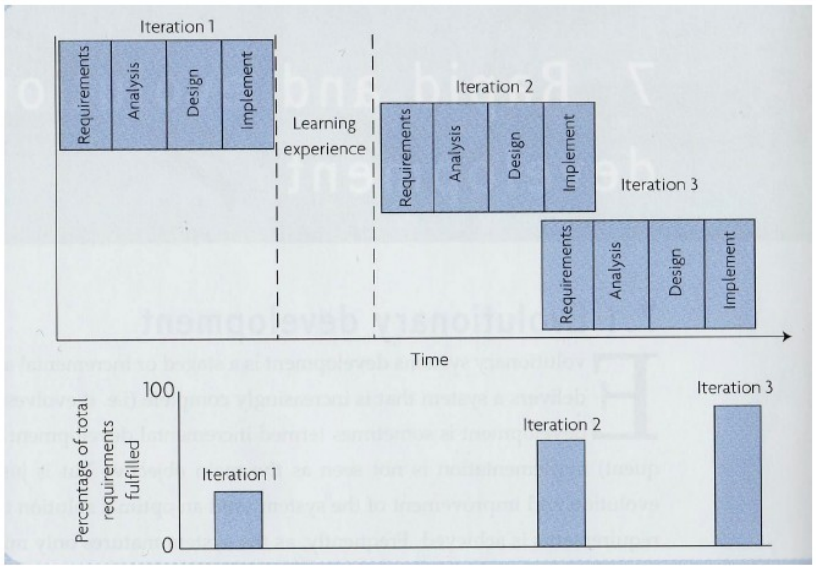
- Emerged in response to increasingly complex requirements and uncertain technological environment
- Always included iterations where some design and implementation is done from the beginning
- Many developers claim it is the only way to develop info sys

- Incremental Development
  a. Completes portions of the system in small increments and integrated as the project progress
  b. Sometimes considered growing a system
- Walking skeleton
  a. The complete system structure is built first, but with bare bone functionality

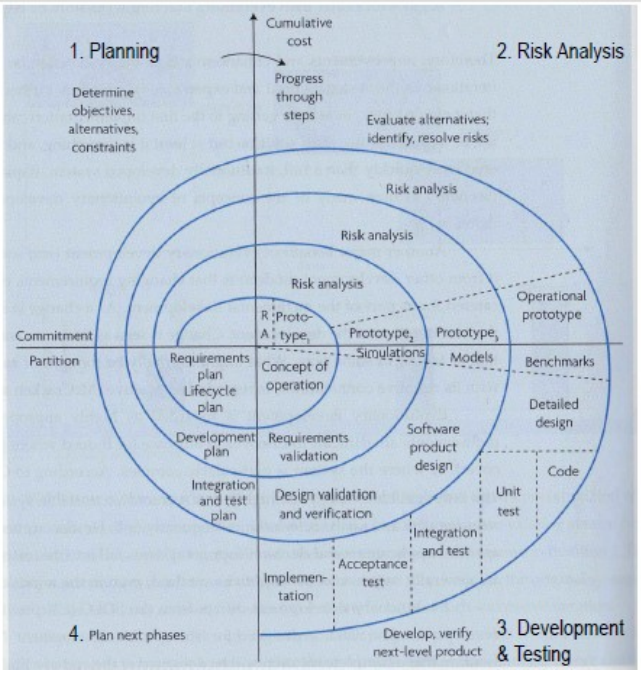
Iterative or Evolutionary Development

- Fundamentally different pgilosophy to SDLC
- Iterative development
- Staged or incremental approach
  a. Design is not perfect
  b. Accommodates change
  c. Requirements not forzen
  d. Does not have to be comprehensive
- Normally a period of learning between each stage
- Most useful when requirement are unclear

Evolutionary Development



Software Engineering: Spiral Model of Development



Prototyping

- "Users only see their Info sys at implementation time when it is too late to make changes"
- Two main types of prototype
  a. Facades
  b. Working
  - Two main types of working prototype
    a. Throwaway
    b. Evolutionary
  - Cost of prototype could be issue

Prototyping Methods

- Analysis Phase
  a. Understand current system and suggest improvements and functional requirements
- Prototyping Phase
  a. Construct a prototyping for evaluation by user
- Evaluation and prototype modification stages
  a. Could be multiple stages
- Design and develop the Target System
  a. Prototype is basis of specification

Criticism of Prototyping

- Inadequate and incomplete system designs
  a. Quick and dirty
- Difficult to manage and control
  a. Poor documentation
- Can become the system without proper engineering

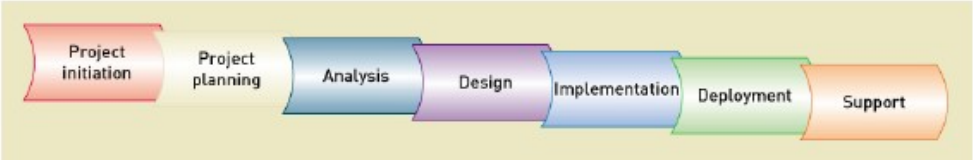
A Generic Adaptive Approach

- Six core processes go across iterations
- Multiple iteration required

Methodologies, Models, Tools, and Techniques

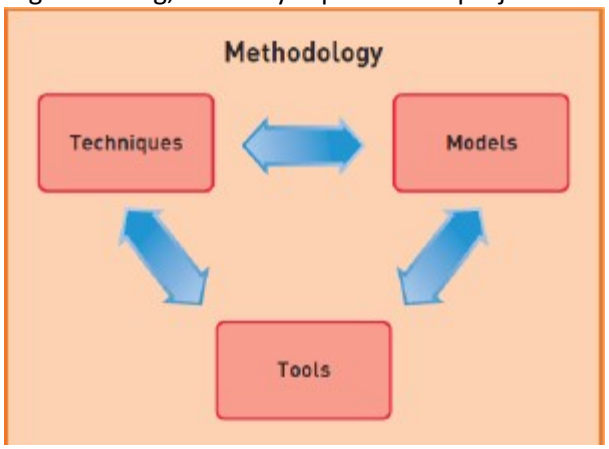
- Methodologies
  a. Provides guidelines for every facet of sys development
  b. Specifies an SDLC with activities and tasks
  c. Specifies project planning and project management models and reporting
  d. Specifics analysis and design models to create
  e. Specifies implementation and testing techniques
  f. Specifies development and support techniques

- Other term used is *System Development Process*

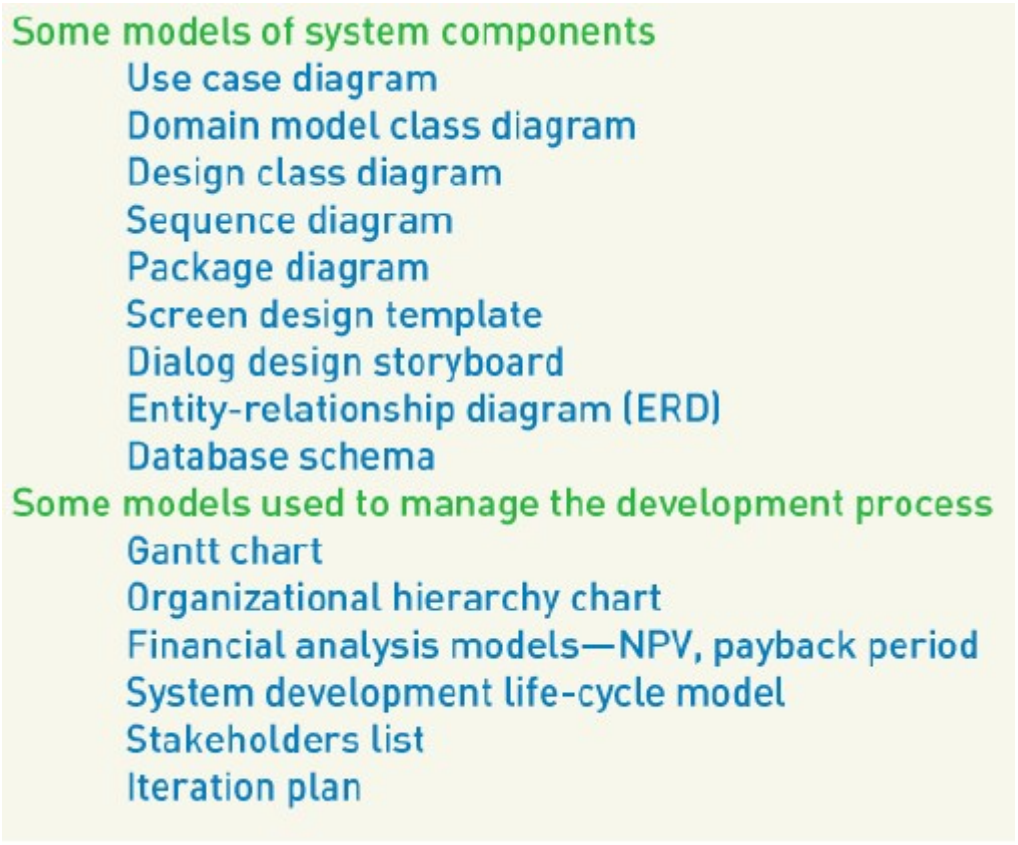


Methodology

- Includes a collection of techniques that are used to complete activities and tasks, including modelling, for every aspect of the project



- Model
  a. An abstraction of an important aspect of the real world
  b. Each model shows a different aspect of the concept
- In IS, models are of system components that will be developed
- Other models are used to manage the development process

- 

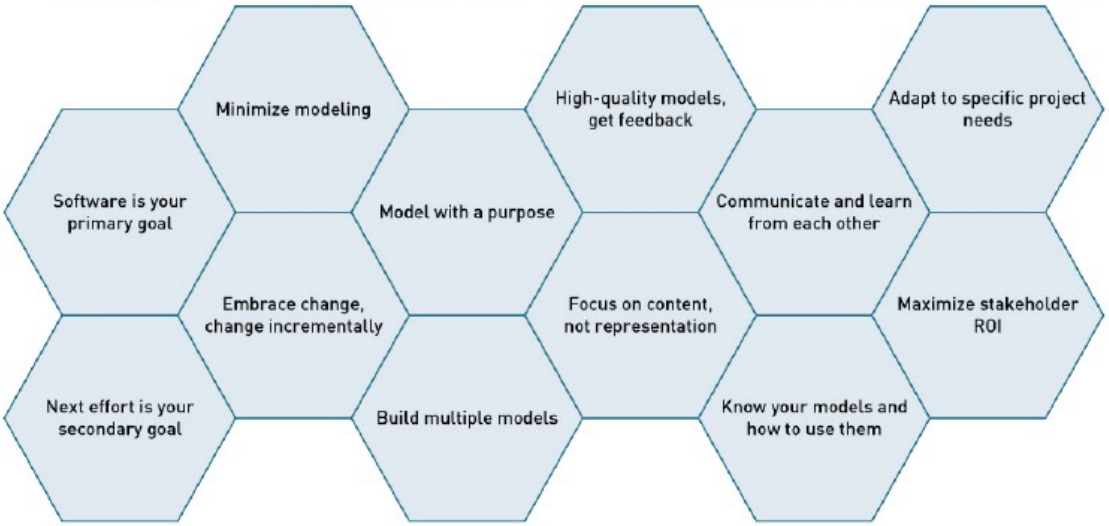
- Tools
  a. Software app that assists developers in creating models or other components required for a project
- Integrated Development Environment
  a. Set of tools that work together to provide a comprehensive development environment
- Visual modelling tools
  a. Tools to create graphical models

- Techniques
  a. Collection of guidelines that help an analyst complete an activity or task
  b. Learning techniques is the key to having expertise in a field

Agile Development

- Guiding philosophy and set of guidelines for developing info system in an **unknown, rapidly changing environment**
- Chaordic
  a. Term for adaptive project - chaotic yet ordered
- Agile Values
  a. Values responding to change over following a plan
  b. Values individuals and interactions over processes and tools
  c. Values working software over comprehensive documentation
  d. Values customer collaboration over contract negotiation

- Agile Modelling (AM) – 12 Principles
  - A philosophy – build only necessary models that are useful and at the right level of detail



Agile Principle

- Develop software as primary goal
  a. Don't get distracted by documentation or models
- Enable the next effort as your secondary goal
  a. Be aware of next step versions or revision
- Minimize your modelling activity
  a. Only build what helps move the project forward
- Embrace change and change incrementally
- Model with a prupose
  a. Model to understand
  b. Model to communicate
- Build multi models
  a. Look at problem from different perspective
- Build high-quality model and get feedback
- Focus on content rather than representation
  a. Informal hand-drawn models are sometimes ok

b. Always focus on stakeholder needs

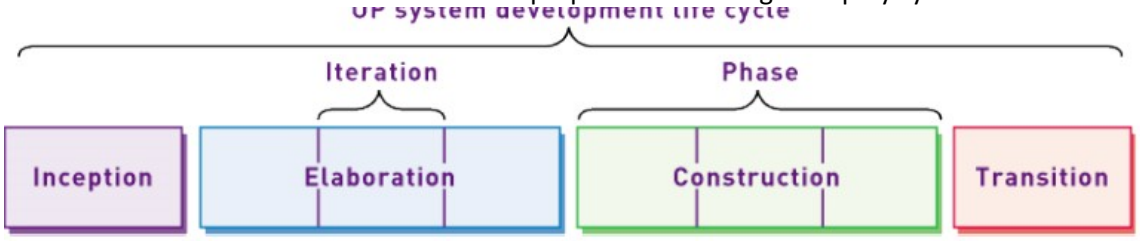
- Learn from each other with open communication
- Know your model and how to use them
- Adapt to specific project needs
- Max stakeholder RMI

Unified Process (UP)

- UP & UML (Unified Modelling Language) were developed together
- Phases

a. UP Phase organize iterations into 4 primary areas of focus during a project

- Inception - getting project started
- Elaboration - understanding system requirement
- Construction - building system
- Transition - prep for and moving to deploy system



Objectives of each phase

UP Phase	Objective
Inception	Develop an approximate vision of the system, make the business case, define the scope, and produce rough estimates for cost and schedule.
Elaboration	Define the vision, identify and describe all requirements, finalize the scope, design and implement the core architecture and functions, resolve high risks, and produce realistic estimates for cost and schedule.
Construction	Iteratively implement the remaining lower-risk, predictable, and easier elements and prepare for deployment.
Transition	Complete the beta test and deployment so users have a working system and are ready to benefit as expected.

- Disiplines

a. Set of functionally related development activities

b. Each discipline are all the activities related to achieving one objective in the development project

c. Two types of disciplines

- Systems development activities
- Project management activities
  - Planning and control disciplines

d. Software Development D

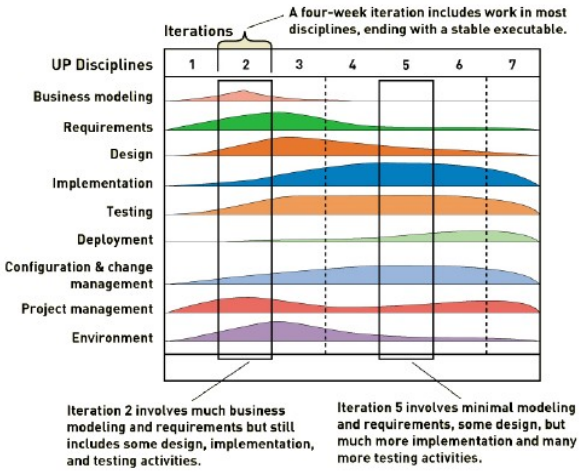
- Business modelling
- Requirements
- Design

- Implementation
- Testing
- Deployment

e. Planning and Control D
- Configuration and change management
- PM
- Environment

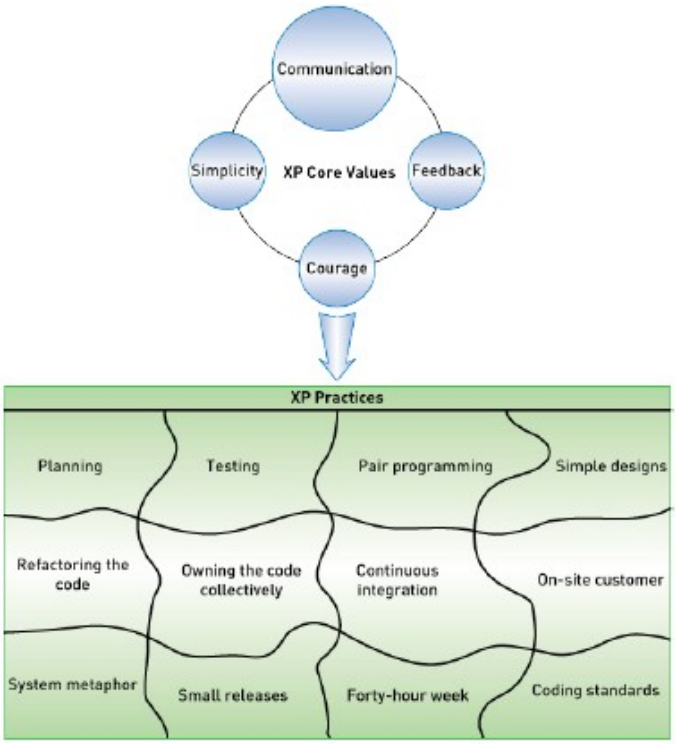
f.

- Disciplines are used across all iterations



Extreme Programming (XP)

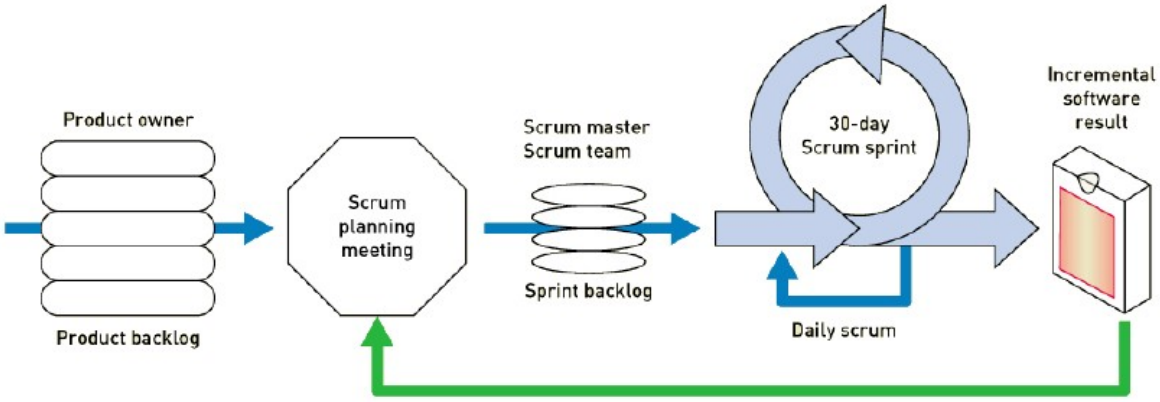
- Takes the best practice of software dev and extends them "to the extreme"
  a. Focuses intensely on proven industry practices
  b. Combines them in unique ways to get better results
- XP core values
  a. Communication
  b. Simplicity
  c. Feedback
  d. Courage
- XP Practices
  a. Planning - based on user stories
  b. Testing - thorough testing at each step
  c. Pair Programming - watch, inspect, trade off
  d. Simple Designs- Agile modelling principles
  e. Refactoring - redo and clean-up as you go
  f. Owning the code collectively - egoless dev, anyone can review and improve code
  g. Continuous integration - grow the software continuously
  h. Onsite customer - get sign-off as you go
  i. System Metaphor - what should the final system look like
  j. Small release - turn over to user frequently
  k. Forty-hour work week - don't overload developers
  l. Coding standards - follow standards for code



- XP Project Approach
  a. Three Levels
     - Outside ring - create user stories and define acceptance test
     - Middle ring - conduct test and do overall planning
     - Inside ring - iteration of coding and testing

Scrum

- Combo of principles of Rugby and Agile
- Product backlog
  a. Prioritized list of user requirements
- Product owners
  a. Client stakeholder who controls backlog
- Scrum master
  a. Scrum PM
- Scrum Sprint
  a. Time controlled mini-project to implement part of the system

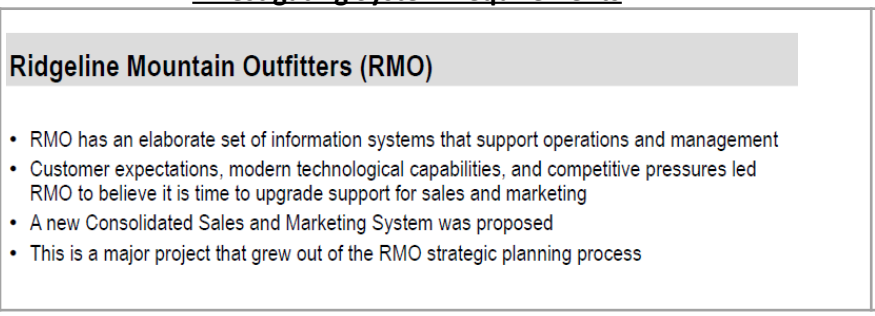


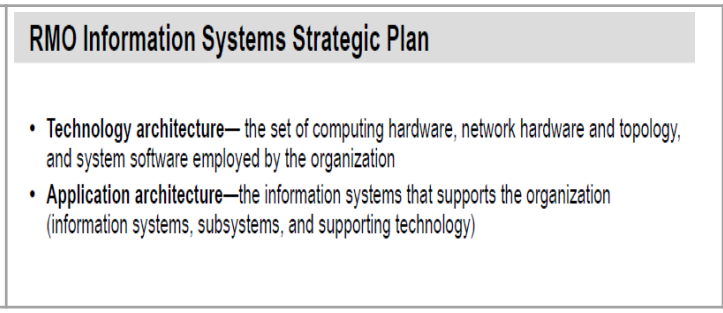
- Scrum Practices

a. Scope of each sprint is frozen
b. Time period is kept constant
c. Daily scrum meeting
   - What have you done the last daily scrum
   - What will you do by the next daily scrum
   - What kept you or is keeping you from completing your work

Week 3 – Investigating System Requirement

Investigating System Requirements



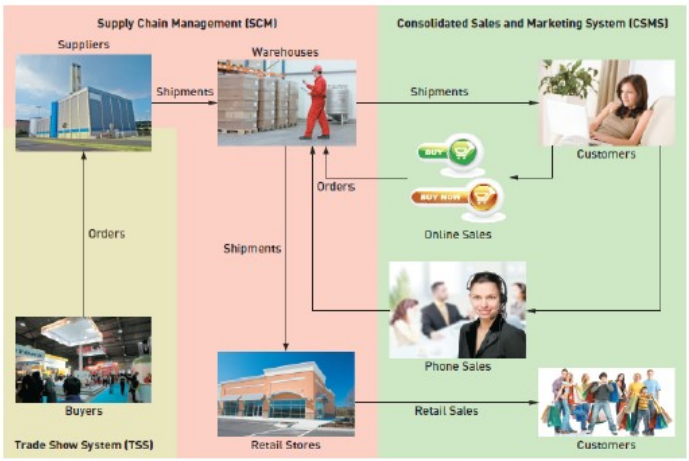


RMO Existing Applicatio nArchitecture

- Supply Chain Management (SCM)
  - 5 year old: Java/Oracle
  - Tradeshow system will interface with SCM
- Phone/Mail Order System
  - 12 years old: Visual Studio/SQL
  - Reached Capacity; minimal integration
- Retail Store System

- Older package solution; minimal integration
- Customer Support System (CSS)
  - Web based system; evolved over the years; minimal integration

Proposed Application Architecture: Integrate SCM and New CSMS



New Consolidated Sales and Marketing System (CSMS)

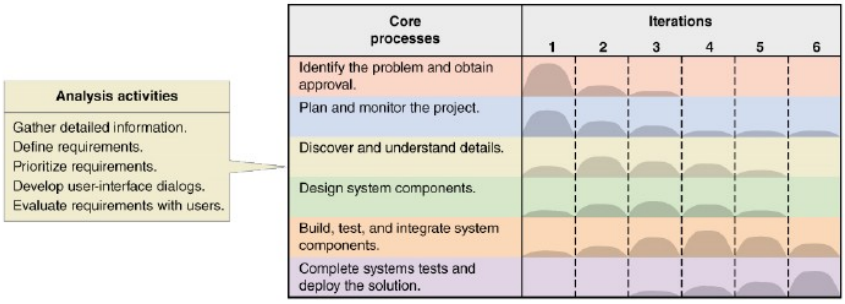
- Sales Subsystem
  - Integrates online, phone, and retail stores
- Order fulfillment
  - Tracks shipments, rate products and services
- Customer account
  - Shopping history, linkups, reward system
- Marketing
  - Promo package, partner relationship, more complete merch info and reporting

System Analysis

- CSMS will require discovering and understanding extensive and complex business processes and business rules
- SDLC indicates the project starts with identifying problem, obtaining approval and planning project

Systems Analysis Activities

Involve discovery and understanding



System Analysis Activities

- Gather Detailed Info
  - Interview, questions, documents, observing business process, research vendors
- Define Requirements
  - Modelling functional requirement and non-functional requirements
- Prioritize Requirements
  - Essential, important vs nice to have
- Develop User Interface Dialogs
  - Flow of interaction between user and system
- Evaluate requirements with users
  - User involvement, feedback, adapt to changes

What are requirements?

- System Requirements =
  - Functional
  - Non-functional
- Functional - activities the system must perform
  - Business uses, functions the user carry out
  - Shown as use case
- Non-functional - other system characteristics
  - Constraints and performance goals

FRUPS + Requirements Acronym

- **F**unctional
- **U**sability
- **R**eliability
- **P**erformance
- **S**ecurity
- **+** even more categories

Requirement categories	FURPS categories	Example requirements
Functional	Functions	Business rules and processes
Nonfunctional	Usability Reliability Performance Security	User interface, ease of use Failure rate, recovery methods Response time, throughput Access controls, encryption

Additional Requirement Categories

- Design constraints
  - Specific restrictions for hardware and software
- Implementation requirements
  - Specific language, tools, protocols
- Interface requirements
  - Interface links to other systems
- Physical requirements
  - Physical facilities and equipment constraints
- Supportability requirements
  - Auto updates and enhancement methods

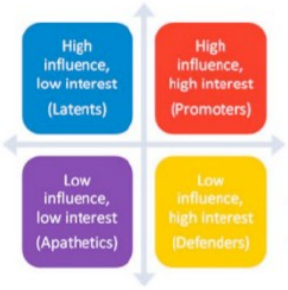
Stakeholder

- **Stakeholder -** person who have an interest in the successful implementation of the system
- **Internal Stakeholder -** person within the organization
- **External Stakeholder -** person outside the organization
- **Operational Stakeholder -** person who regularly interact with the system
- **Executive Stakeholder -** person who don't directly interact, but use the information or have financial interest

Stakeholders of a comprehensive accounting system for public company

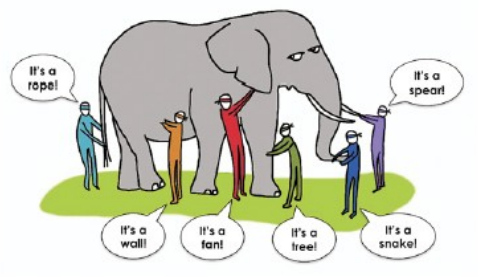
Prioritise and understand your stakeholders

- Someone's position on the grid shows the actions you have to take with them



- You need to understand how they feel about the project
- Determines how you engage /communicate with them

Gathering requirements ...



... now we know

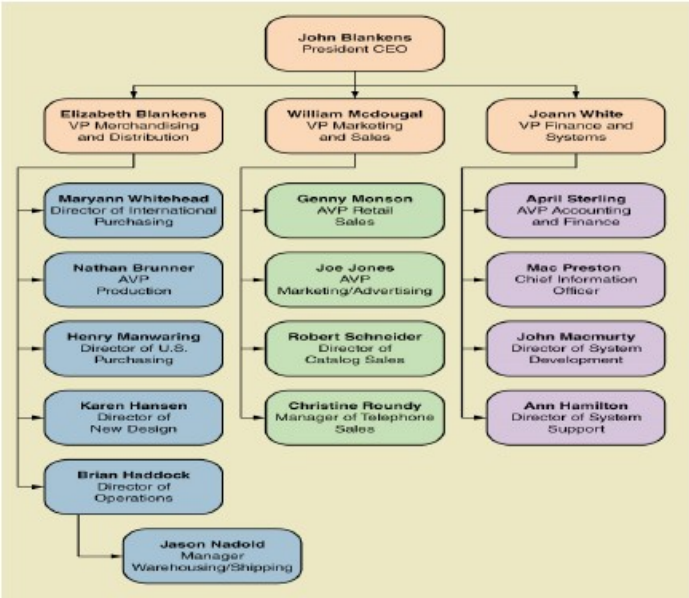
- Why we want to gather requirements?
- What we need to investigate?
- Who we need to gather these requirements from?

So ... how do we find out what the stakeholders want?

Stakeholder for RMO CSMS Project

- Phone/mail sales order clerk
- Warehouse and shipping personnel
- Marketing personnel who maintain online catalogue information
- Marketing, sales, accounting and financial manager
- Senior executive
- Customer
- External shippers

RMO Internal Stakeholders



Information Gathering Techniques

- Interviewing user and other stakeholders
- Distributing and collecting questionnaires
- Reviewing inputs, outputs and documentation
- Observing and documenting business procedures
- Researching vendor solutions
- Collecting active user comments and suggestion

Interviewing Users and Other Stakeholders

- Prep detailed question
- Meet with individuals or groups of users
- Obtain and discuss answers to questions
- Document answers
- Follow up as needed in the future meetings or interviews

Themes for Information Gathering Questions

Theme	Questions to users
What are the business operations and processes?	What do you do?
How should those operations be performed?	How do you do it? What steps do you follow? How could they be done differently?
What information is needed to perform those operations?	What information do you use? What inputs do you use? What outputs do you produce?

Preparing for the Interview

Checklist for Conducting an Interview

Before

- ☐ Establish the objective for the interview.
- ☐ Determine correct user(s) to be involved.
- ☐ Determine project team members to participate.
- ☐ Build a list of questions and issues to be discussed.
- ☐ Review related documents and materials.
- ☐ Set the time and location.
- ☐ Inform all participants of objective, time, and locations.

During

- ☐ Arrive on time.
- ☐ Look for exception and error conditions.
- ☐ Probe for details.
- ☐ Take thorough notes.
- ☐ Identify and document unanswered items or open questions.

After

- ☐ Review notes for accuracy, completeness, and understanding.
- ☐ Transfer information to appropriate models and documents.
- ☐ Identify areas needing further clarification.
- ☐ Thank the participants.
- ☐ Follow up on open and unanswered questions.

Interview Session Agenda

Discussion and Interview Agenda

Setting

Objective of Interview
Determine processing rules for sales commission rates

Date, Time, and Location
April 21, 2016, at 9:00 a.m. in William McDougal's office

User Participants (names and titles/positions)
William McDougal, vice president of marketing and sales, and several of his staff

Project Team Participants
Mary Ellen Green and Jim Williams

Interview/Discussion

1. *Who is eligible for sales commissions?*
2. *What is the basis for commissions? What rates are paid?*
3. *How is commission for returns handled?*
4. *Are there special incentives? Contests? Programs based on time?*
5. *Is there a variable scale for commissions? Are there quotas?*
6. *What are the exceptions?*

Follow-Up

Important decisions or answers to questions
See attached write-up on commission policies

Open items not resolved with assignments for solution
See item numbers 2 and 3 on open items list

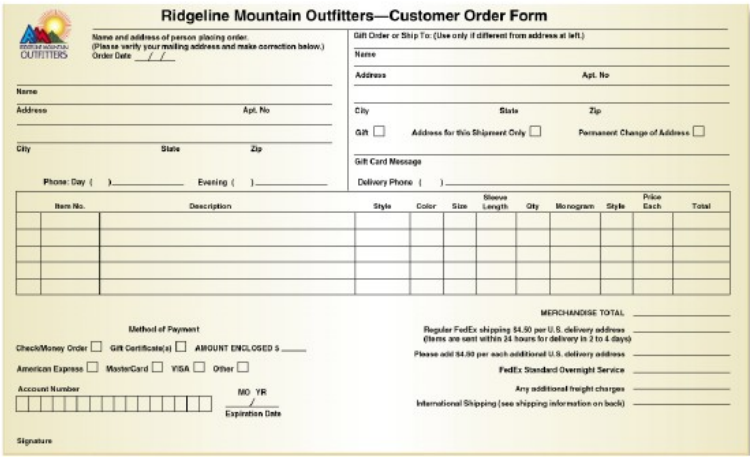
Date and time of next meeting or follow-up session
April 28, 2016, at 9:00 a.m.

Keeping an Open Items List

ID	Issue title	Date identified	Target end date	Responsible project person	User contact	Comments
1	Partial shipments	6-12-2016	7-15-2016	Jim Williams	Jason Nadold	Ship partials or wait for full shipment?
2	Returns and commissions	7-01-2016	9-01-2016	Jim Williams	William McDougal	Are commissions recouped on returns?
3	Extra commissions	7-01-2016	8-01-2016	Mary Ellen Green	William McDougal	How to handle commissions on special promotions?

Review Inputs, Outputs, and Procedures

ollect Questionnaires



Ridgeline Mountain Outfitters—Customer Order Form

Name and address of person placing order. (Please verify your mailing address and make correction below.)
Order Date ___/___/___

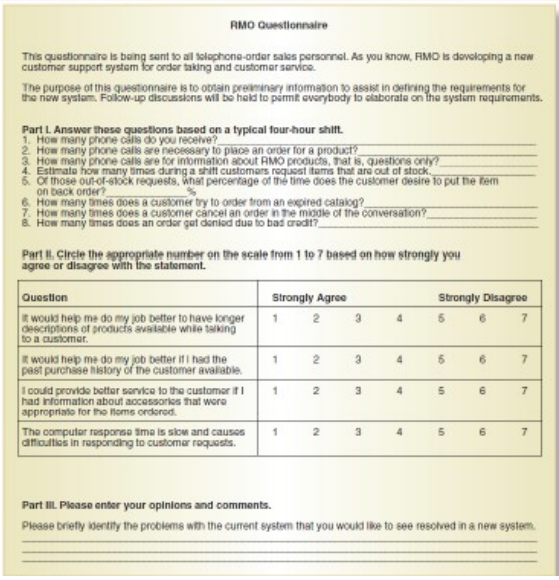
Name
Address Apt. No
City State Zip
Phone: Day ( ) Evening ( )

Gift Order or Ship To: (Use only if different from address at left.)
Name
Address Apt. No
City State Zip
Gift ☐ Address for this Shipment Only ☐ Permanent Change of Address ☐
Gift Card Message
Delivery Phone ( )

Item No.	Description	Style	Color	Size	Sleeve Length	Qty	Monogram	Style	Price Each	Total

Method of Payment
Check/Money Order ☐ Gift Certificate(s) ☐ AMOUNT ENCLOSED $ ____
American Express ☐ MasterCard ☐ VISA ☐ Other ☐
Account Number
MO YR
___/___
Expiration Date
Signature

MERCHANDISE TOTAL
Regular FedEx shipping $4.50 per U.S. delivery address (Items are sent within 24 hours for delivery in 2 to 4 days)
Please add $4.00 per each additional U.S. delivery address
FedEx Standard Overnight Service
Any additional freight charges
International Shipping (see shipping information on back)



RMO Questionnaire

This questionnaire is being sent to all telephone-order sales personnel. As you know, RMO is developing a new customer support system for order taking and customer service.

The purpose of this questionnaire is to obtain preliminary information to assist in defining the requirements for the new system. Follow-up discussions will be held to permit everybody to elaborate on the system requirements.

Part I. Answer these questions based on a typical four-hour shift.

1. How many phone calls do you receive? _____
2. How many phone calls are necessary to place an order for a product? _____
3. How many phone calls are for information about RMO products, that is, questions only? _____
4. Estimate how many times during a shift customers request items that are out of stock. _____
5. Of those out-of-stock requests, what percentage of the time does the customer desire to put the item on back order? _____ %
6. How many times does a customer try to order from an expired catalog? _____
7. How many times does a customer cancel an order in the middle of the conversation? _____
8. How many times does an order get denied due to bad credit? _____

Part II. Circle the appropriate number on the scale from 1 to 7 based on how strongly you agree or disagree with the statement.

Question	Strongly Agree						Strongly Disagree
It would help me do my job better to have longer descriptions of products available while talking to a customer.	1	2	3	4	5	6	7
It would help me do my job better if I had the past purchase history of the customer available.	1	2	3	4	5	6	7
I could provide better service to the customer if I had information about accessories that were appropriate for the items ordered.	1	2	3	4	5	6	7
The computer response time is slow and causes difficulties in responding to customer requests.	1	2	3	4	5	6	7

Part III. Please enter your opinions and comments.

Please briefly identify the problems with the current system that you would like to see resolved in a new system.

Interview Follow up

- As documentation created after the interview should be reviewed for accuracy as soon as possible after the interview
- Follow up interviews are required to explain and verify the models with the interview participants, and ask further questions
- You will have unresolved issues
- They should be tracked and resolved

<u>Additional techniques</u>

- Obeerve and Document Business Processes
  - Watch and learn
  - Document with activity diagram
- Research vendor solution
  - See what others have done for similar situation
  - White papers, vendor literature, competitors
- Collect active user comments and suggestions
  - Feedback on modes and tests
  - Users know it when they see it

Workshops

- Sometimes referred to as JAD sessions:
  - Joint application development
- Get all stakeholder in a room for a couple of days and facilitate discussion
  - Build models with everyone
  - Share ideas
- Requires a conference room
- Can be difficult to organise
- Once initial session is complete, all notes must be written up into document
- This is then distributed for feedback

Aim

- Determine clear purpose
  - Agenda
- Determine participants
  - Possibly a mixture of workshop type based on constituents
- Distribute material early
  - Must provide time for people to read and think about prior to workshop
- Facilities
- Personnel
  - Facilitator
  - Scribe
  - System 'driver'
- Promptly distribute material afterwards - keep motivation
  - Always get feedback for correction, misunderstanding

Tools

- Whiteboard
- CASE Tools
  - Diagramming support, data dictionary creation, versioning and repository function, even schema construction and reverse-engineering
  - LIMITATIONS;
    - Often tied to specific notation
    - May force logical design artefacts onto conceptual model by enforcing notation rules
    - Don't allow for working with draft or incomplete models
    - Widespread edits
    - Handling changes to logical models once physical model has been deployed
  - Pen, paper, basic drawing package (Visio)

Survey and Questionnaire

- Online, paper based, email
- **Advantage**
  - Cover a wide spectrum of people
- **Disadvantage**
  - Low response rate
  - Questions may not be well understood

Subject-Matter Experts

- Sometimes referred to as Domain Expert
- Not users of the system, but have relevant knowledge of the business domain
- Should be used to confirm:
  - All assumptions/assertions
  - Business rules
- Usually involved in JAD sessions/workshops
- Definitely part of the review and refine process to turn the initial model into a final working draft

Riding the trucks

- Observation
- Don't just rely on interview and workshop to understand business process
- Get out and see the business process in action
    - Talk to frontline staff, watch what they do
    - See how variation to the official way of doing things
    - See how people interpret and use same data in different ways
    - Look at how people actually use systems
    - Get a feel for data quality

Review Existing Reports, Forms, and Procedure Description

- Existing internal business documents and procedure descriptions
    - Identify business rules, discrepancies, redundancies
    - Be cautious of outdated material
    - obtain preliminary understanding of processes
    - Use as guidelines/ visual cues to guide interview
- External industry wide professional org and trade publication

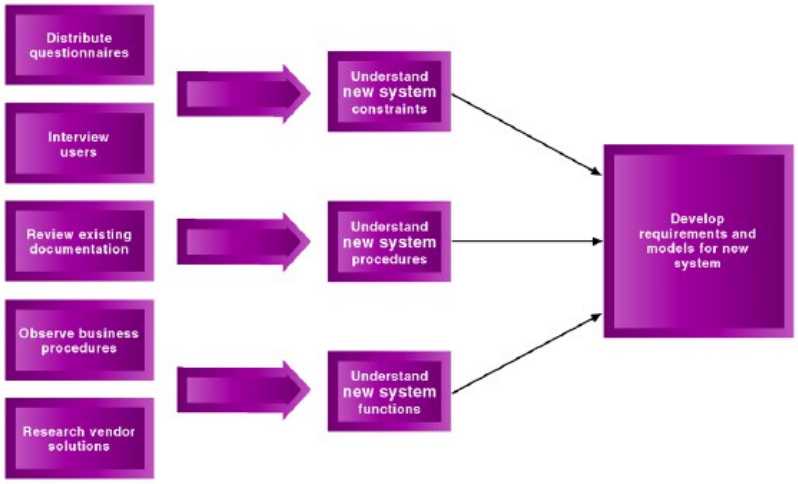
Research Vendor Solution

- Positive contribution of vendor solutions
    - Frequently provide new ideas
    - May be start of the art
    - Cheaper and less risky
- Danger
    - May purchase solutions before understanding problem
    - May no address all requirements
    - Vendor support, upgrade issues

<u>Useful techniques in vendor research</u>

- Technical specifications from vendors
- Demo or trial system
- References of existing clients
- On-site visits
- Printout of screens and reports
- RFP - request for proposal

Information Gathering and Model Building

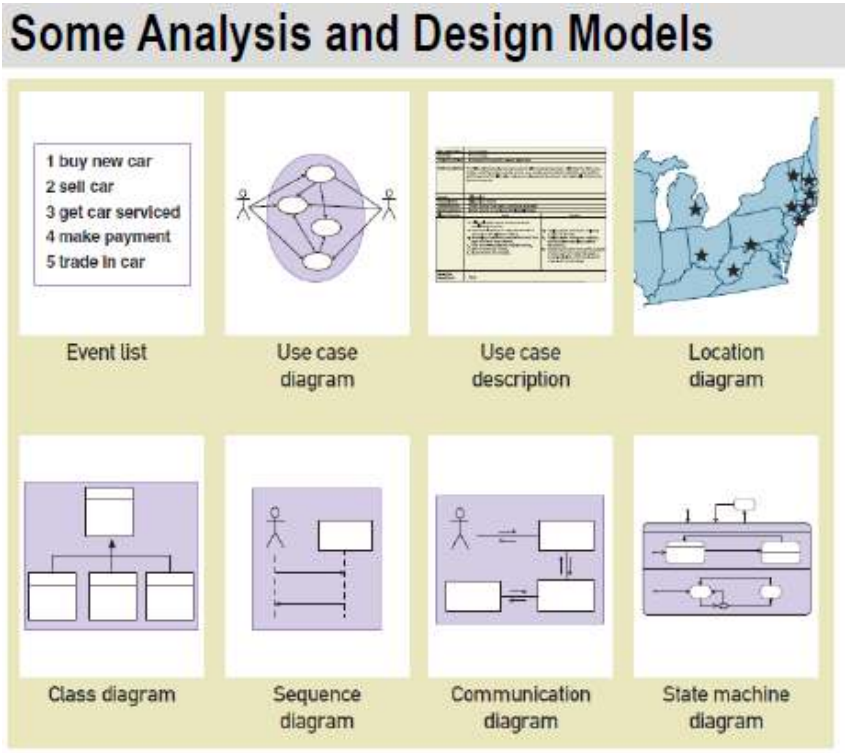


Models and Modelling

- Model - a representation of some aspect of the system being built
- Types of Models
  - Textual model - something written down
  - Graphical model - diagram, schematic
  - Math model - formula, stats, algorithms
- Unified Modelling Language (UML)
  - Standard graphical modelling symbols/terminology used for IS

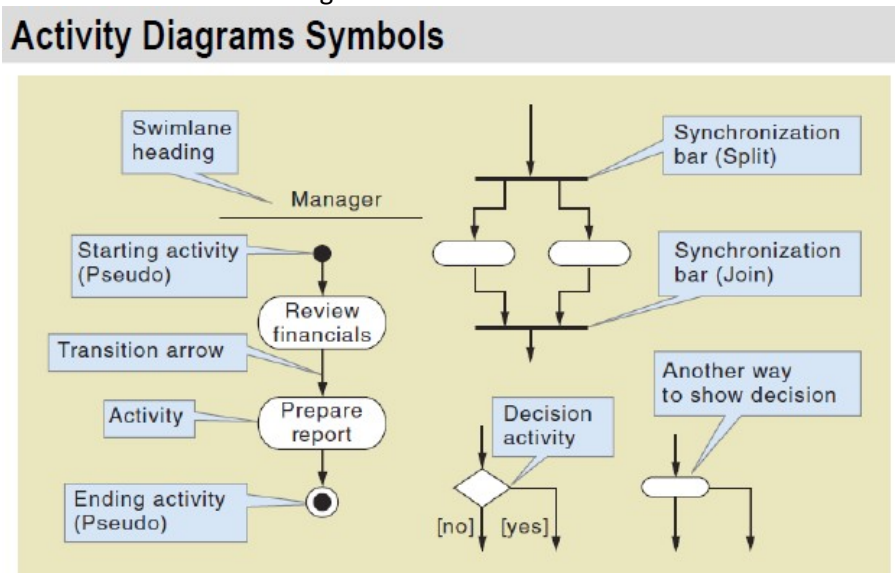
Reason for Modelling

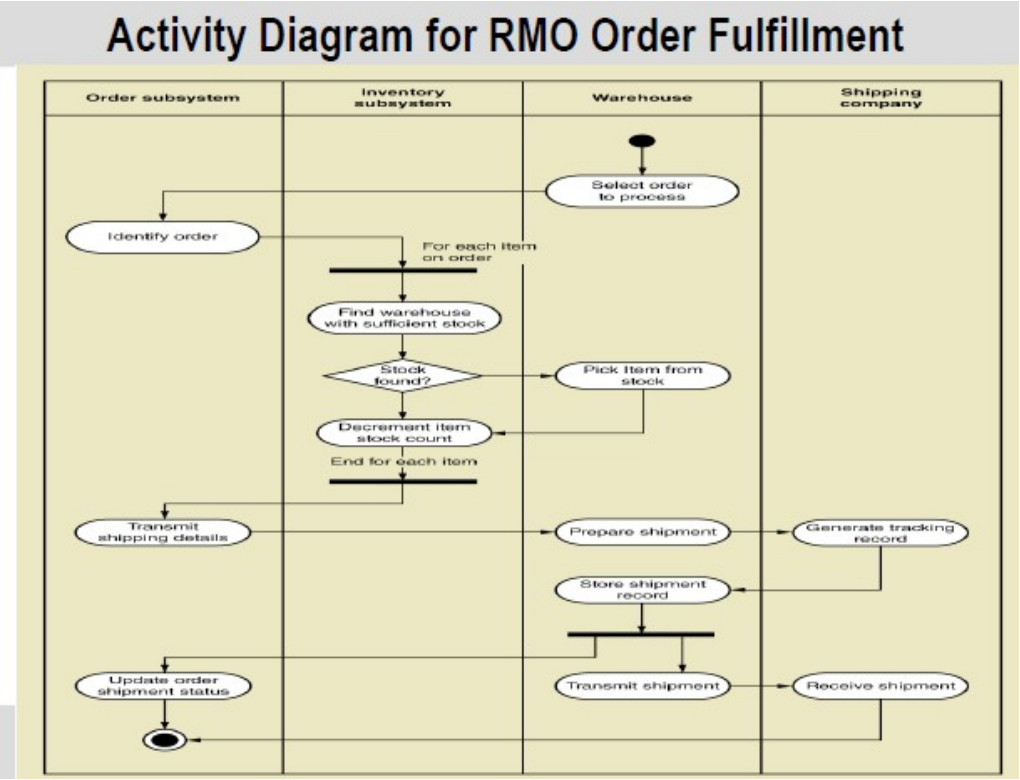
- Learning from modelling process
- Reducing complexity by abstraction
- Remembering all the details
- Communicating with other development team members
- Communicating with a variety of users and stakeholders
- Documenting what was done for future maintenance/enhancement

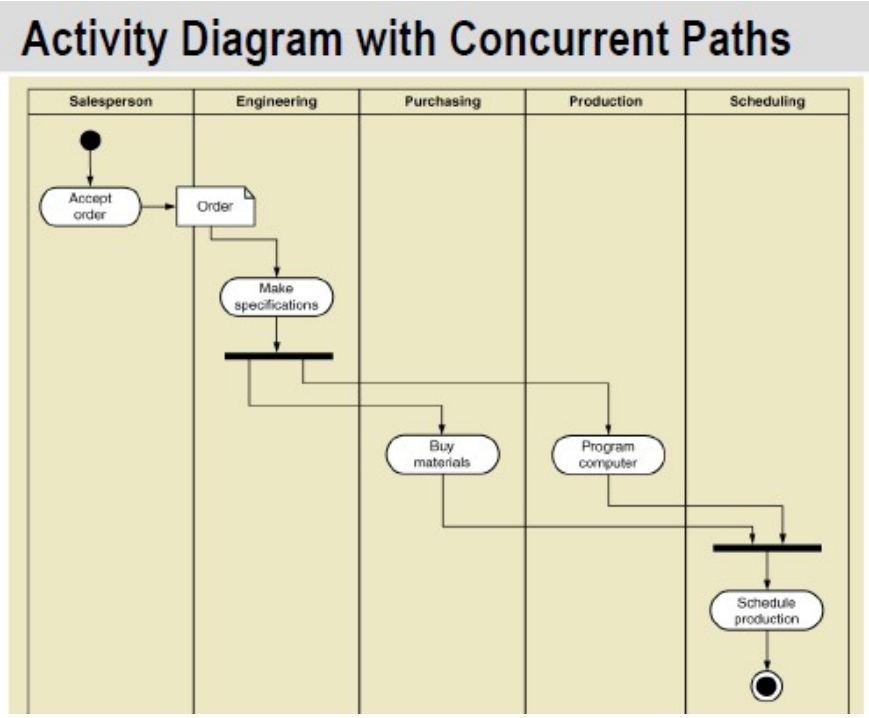


Documenting Workflow and Activity Diagrams

- **Workflow -** sequence of processing steps that completely handles one business transaction or customer request
- **Activity Diagram -** describes user activities, the person who does each activity, and the sequential flow of these activities
    - Useful for showing a graphical model of a workflow
    - A UML Diagram







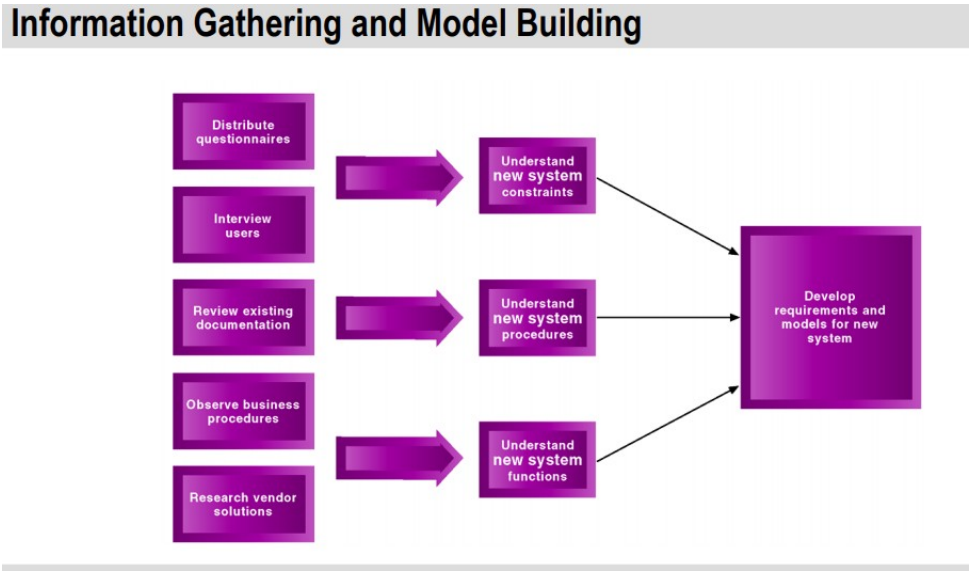
SUMMARY

- System analysis involves defining system requirements - functional and non-functional
- Analysis activities include;
  - Gather detailed info
  - Define requirements
  - Prioritize requirements

    - Develop user-interface design
    - Evaluate requirements with users
- FURPS + is acronym for functional, usability, reliability, performance, and security requirements
- Stakeholders are the people who have an interest in the success of the project
- Internal vs external and operational vs executive stakeholders
- Information gathering techniques are used to collect information about the project
- UML activity diagram is used to document (model) workflow after collecting information
- Models and modelling are used to explore and document requirements
- Unified Modelling Language is the standard set of notations and terminology for Information Systems Models

Week 4 – Identifying User Stories and Use Case

Identifying User Stories and Use Cases



Models

- All designers use some kind of modelling technique
- Models simulate some aspect of the artefact being designed
    - Allows focus on that part of the artefact
- Essentially a communication tool

Why?

- Cheaper than building real thing
- Allows designer to try out idea without having to construct
- Can be used to document and describe:
  - An existing system
  - A design for a proposed system
- Very effective mechanisms for eliciting and communicating business requirements
  - Not always done for system dev
- The key means by which stakeholders can provide feedback on and verify design ideas

What formats?

- **Mathematical -** formulas that describe technical aspect of the system, pseudo-code algorithms
- **Descriptive -** narrative memos, reports, or lists that describe aspects of the system, prototypes, screenshots
- **Graphical -** Diagrams and schematic representations of some aspect of the system

What Type?

- Essentially three different views of an info system:
  - Processes and affordances (functions)
  - Data and info (and metadata)
  - Behaviour (a combo of the other two - object oriented models)

Model Content

Regardless of the modelling technique or type of model, three sets of models can be produced fo each design project:

- Conceptual
- Logical
- Physical

Implementation Independent
Deep structure only

Implementation Specific
Deep, surface and physical structures

IMPORTANT: These 3 types of models apply to data models, process models or behaviour models

Conceptual Model

- Initial model developed with clients and subject matter experts
- Business oriented
- Most creative, design oriented model
- Not constrained by anything other than most basic notations
- Aims to specific what the system should do, independent of how it does
- Main audience: client, users and experts

Logical Model

- Result of modifying conceptual model
  - Complete , detailed specification of functionality and info structure

        - Some designers treat conceptual and logical as same thing
    - Still tech independent
    - Think ad conceptual as first draft, logical as detailed, complete and technically correct final draft
    - Main Audience: client, users, experts, developers, designers

Physical Model

- Technology specific model
- Results from modifying the logical model to take into account how the system will do what it is intending to do
- Technically oriented
- Models surface and physical structure as well as the deep structure
- Model is tuned to take into account performance requirements, efficiency in processing, limitations of hardware, architecture
- Main Audience: System Development and Maintainers

Why we do this?

- Having separate logical and physical models allows us to fine tune the implementation, upgrade components, without impacting the fundamental design of the system
- Protects business rules and requirements from technical, engineering considerations
- Conceptual models allow us more freedom to communicate with clients - less complexity

Who should be involved

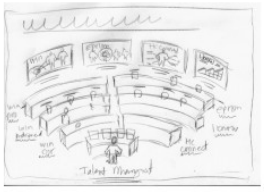
- Designer
- System owner, user, project sponsor
- Subject matter experts
- IT department

What they look for?

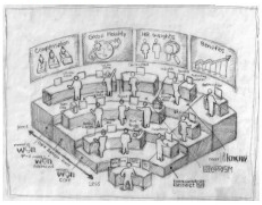
- Yourself
- Other designers
- System developers and maintainers

Models

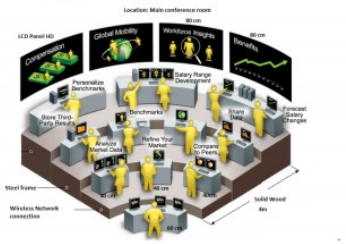
Conceptual model



Logical model



Physical model



Centred on business concepts

Used to discuss with a business audience (client, subject matter experts, etc.)

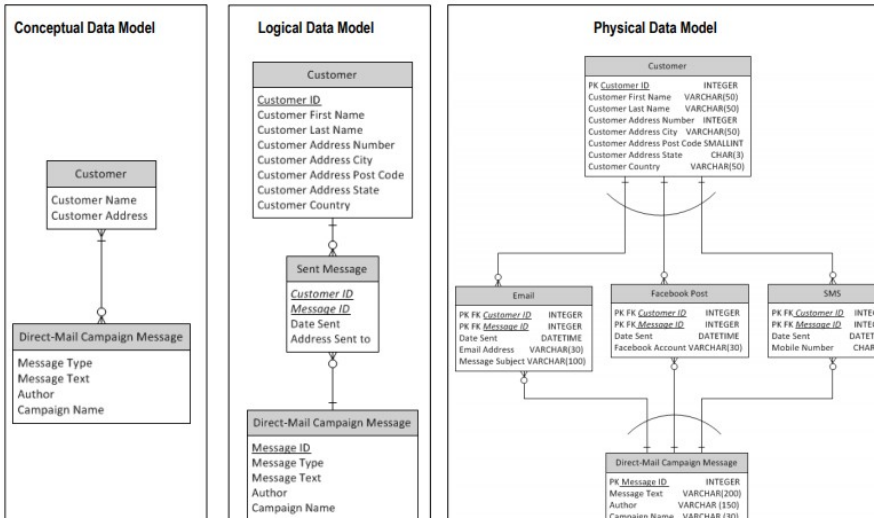
Centred on business concepts

Used to discuss with a business audience (client, subject matter experts, etc.)

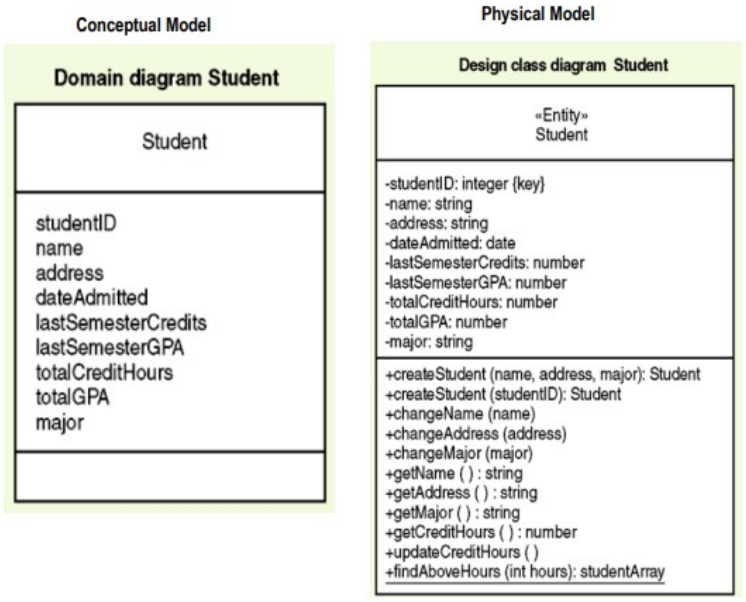
Centred on business concepts plus technical specifications

Used to discuss with a technical audience (Engineers, data base admins, network admins, etc.)

Example : Entity-Relationship model



Example: UML Class Diagrams

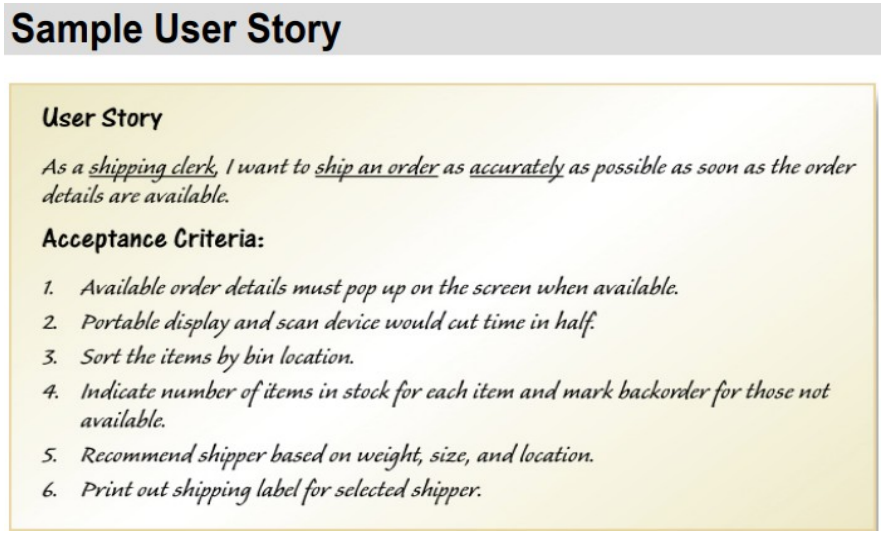


RMO Case so far …..

- Last week, we provided an overview of systems analysis activities, functional and non-functional requirements, modelling, and information gathering techniques
- This lecture focuses on identifying and modelling the key aspect of functional requirements– **use cases**
- In the RMO Tradeshow System from Lecture 1, some use cases are:
  - *Look up supplier*
  - *Enter/update product information*
  - *Enter/Update contact information*

USER STORIES

- Is a one sentence description of a work-related task done by a user to achieve some goal or result
- Acceptance Criteria identify the features that must be present at the completion of the task
- The template for a user story description is
  - AS A <ROLE> I WANT TO <GOAL> SO THAT <BENEFIT>



USE CASE

- Use case - an activity that the system performs, usually in response to a request by a user
- Use case defines functional requirements
- Analysts decompose the system into a set of use cases (functional description)
- Two techniques for identifying use cases
  - User goal techniques
  - Event decomposition techniques
- Name each use case using **verb-noun**

User Goal techniques

- Identify all the potential categories of user of the system
- Interview and ask them to describe the tasks the computer can help them with
- Probe further to refine tasks into specific user groups
  - I need to Ship items, track a shipment, create a return

User Goal Technique

Some RMO CSMS Users and Goals

User	User goal and resulting use case
Potential customer	Search for item Fill shopping cart View product rating and comments
Marketing manager	Add/update product information Add/update promotion Produce sales history report
Shipping personnel	Ship items Track shipment Create item return

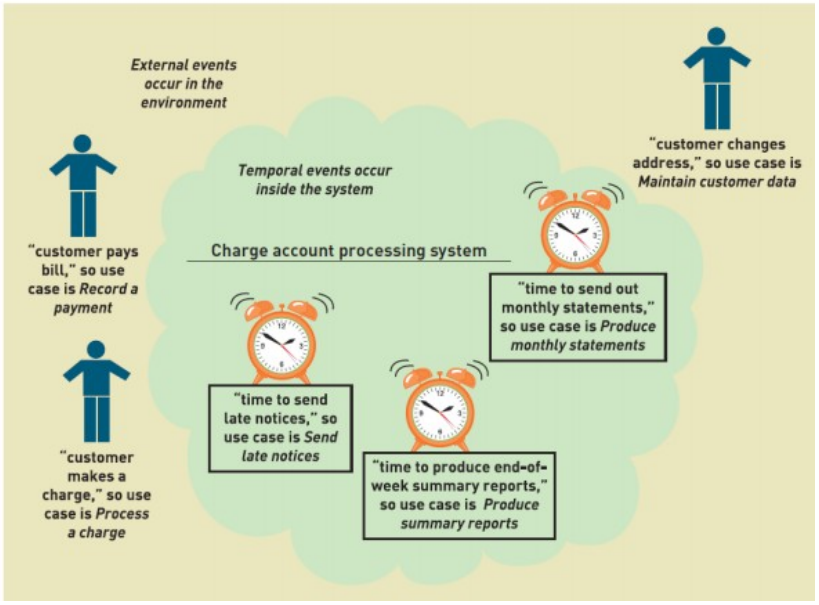
User Goals Technique: Specific Steps

1. Identify all the potential users for the new system
2. Classify the potential users in terms of their functional role (sales, shipping)
3. Further classify potential users by organizational level (executive, management)
4. For each type of user, interview them to find a list of specific goals they will have when using new systems
5. Create a list of preliminary use cases organized by type of user
6. Look for duplicates with similar use case names and resolve inconsistencies
7. Identify where different types of users need the same user cases
8. Review the completed list with each type of user and then with interested stakeholders

Event Decomposition technique

- More comprehensive and complete technique
  - Identify the events that occur to which the system must respond
  - For each event, name a use case that describes what the system does when the system occurs
- Event something that occurs at a specific time and place, can be described and should be remembered by the system

Events and Use Cases



Types of Events

- External events
  - An event that occurs outside the system, usually initiated by an external agent or actor
- Temporal Event
  - An event that occurs as a result of reaching a point in time
- State Event
  - An event that occurs when something happens inside the system that triggers some process
  - Reorder point is reached for inventory item

<u>External Event Checklist</u>

- External agent or actor want something resulting in a transaction
  - Customer buys a product
- External people want some info
  - Customers want to know product details
- External data changed and needs to be updated
  - Customer has new address and phone
- Management wants some information
  - Sales manager wants updated on production plans

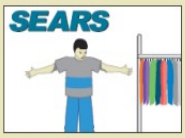
<u>Temporal Event Checklist</u>

- Internal outputs needed a point in time
  - Management reports (summary or exception)
  - Operational reports (detailed transaction
  - Internal statement and document (including payroll)
- External Outputs needed at point of time
  - Statements, status reports, bills and reminders

Finding the actual event that affects the system

Customer thinks about getting a new shirt

Customer drives to the mall



Customer tries on a shirt at Sears

Customer goes to Walmart

Customer tries on a shirt at Walmart

Customer buys a shirt
(the event that directly affects the system!)

Tracing a sequence of transactions resulting in many events

Customer requests a catalog

Customer wants to check item availability

Customer places an order

Customer changes or cancels an order

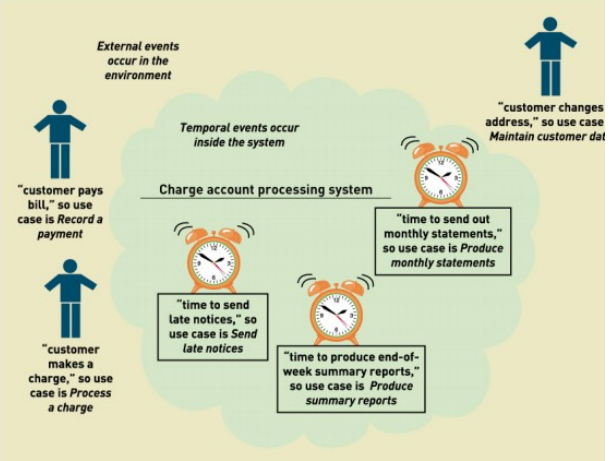
Customer wants to check order status

Customer updates account information

Customer returns the item

Event Decomposition Technique: Specific Steps

1. Consider the external events in the system environment that require a response from the system by using the checklist shown in Figure
2. For each external event, identify and name the use case that the system requires



Event Decomposition Technique: Specific Steps

1. Consider the temporal events that require a response from the system by using the checklists
2. For each temporal event, identify and name the use case that the system requires and then establish the point of time that will trigger the use case

External events to look for include:

√ External agent wants something resulting in a transaction
√ External agent wants some information
√ Data changed and needs to be updated
√ Management wants some information

Temporal events to look for include:

√ Internal outputs needed
  √ Management reports (summary or exception)
  √ Operational reports (detailed transactions)
  √ Internal statements and documents (including payroll)
√ External outputs needed
  √ Statements, status reports, bills, reminders

Event Decomposition Technique: Specific Steps (continued)

5. Consider the state events that the system might respond to, particularly if it is a real-time system in which devices or internal state changes trigger use cases.
6. For each state event, identify and name the use case that the system requires and then define the state change.
7. When events and use cases are defined, check to see if they are required by using the perfect technology assumption. Do not include events that involve such system controls as login, logout, change password, and backup or restore the database, as these are put in later.

Event Decomposition techniques: Benefits

- Events are broader than user goals: Capture temporal and state events
- Help decompose at the right level of analysis: an Elementary Business Process(EBP)
- EBP is a fundamental business process performed by one person, in one place, in response to a business event
- Uses perfect technology assumption to make sure functions that support the users work are identified and not additional functions for security and system controls

Waiters on call meal-delivery system

Waiters on Call is a restaurant meal-delivery service started in 2010 by Sue and Tom Bickford. The Bickfords worked for restaurants while in college and always dreamed of opening their own restaurant; unfortunately, the initial investment was always out of reach. The Bickfords noticed that many restaurants offer takeout food, and that some restaurants-primarily pizzerias-offer home-delivery service. However, many people they met seemed to want home delivery service with a wider food selection.

Sue and Tom conceived Waiters on Call as the best of both worlds: a restaurant service without the high initial investment. They contracted with a variety of well-known restaurants in town to accept orders from customers and to deliver the complete meals. After preparing the meal to order, the restaurant charges Waiters on Call a wholesale price, and the customer pays retail plus a service charge and tip when the meals are delivered. Waiters on Call started modestly, with only two restaurants and one delivery driver working the dinner shift. Business rapidly expanded, until the Bickfords realized they needed a custom computer system to support their operations. They hired a consultant, Sam Wells, to help them define what sort of system they needed.

Waiters on call meal-delivery system

"What events happen when you are running your business that make you want to reach for a computer?" asked Sam. "Tell me about what usually goes on."

"Well," answered Sue, "when a customer calls in wanting to order, I need to record it and get the information to the right restaurant. I need to know which drivers are available to pick up the order, so I need drivers to call in and tell me when they are free. Perhaps this could be included as a smartphone or iPad app. Sometimes, customers call back wanting to change their orders, so I need to find the original order and notify the restaurant to make the change."

"Okay, that's great. Now, how do you handle the money?" queried Sam.

Tom Jumped in. "The drivers get a copy of the bill showing the retail price directly from the restaurant when they pick up the meal. The bill should agree with our calculations. The drivers collect that amount plus a service charge. When drivers report in at closing, we add up the money they have and compare it with the records we have. After all drivers report in, we need to create a deposit slip for the bank for the day's total receipts. At the end of each week, we calculate what we owe each restaurant at the agreed-to wholesale price and send them a statement and check."

"What other information do you need to get from the system?" continued Sam.

"It would be great to have some information at the end of each week about orders by restaurant and orders by area of town – things like that," added Sue. "That would help us decide about advertising and restaurant contracts. Then, we need monthly statements for our accountant."

User Goals

- Produces an end-of-day deposit slip
- Produces end-of-week restaurants payment
- Produces weekly sales reports
- Produces monthly financial reports

Events

- A customer calls to place an order - *Record an order*
- A driver finished with a delivery - *Record delivery completion*
- A customer calls back to change an order - *Update an order*
- A driver reports for work - *Sign in the driver*
- A driver submits the day's receipts: *Reconcile driver receipts*

Use Cases and Brief Use Case Descriptions

- Brief use case description is often a one sentence description showing the main steps in a use case

Use case	Brief use case description
Create customer account	User/actor enters new customer account data, and the system assigns account number, creates a customer record, and creates an account record.
Look up customer	User/actor enters customer account number, and the system retrieves and displays customer and account data.
Process account adjustment	User/actor enters order number, and the system retrieves customer and order data; actor enters adjustment amount, and the system creates a transaction record for the adjustment.

RMO CSMS Project Use Cases

CSMS Sales Subsystem	
Use cases	**Users/actors**
Search for item	Customer, customer service representative, store sales representative
View product comments and ratings	Customer, customer service representative, store sales representative
View accessory combinations	Customer, customer service representative, store sales representative
Fill shopping cart	Customer
Empty shopping cart	Customer
Check out shopping cart	Customer
Fill reserve cart	Customer
Empty reserve cart	Customer
Convert reserve cart	Customer
Create phone sale	Customer service representative
Create store sale	Store sales representative

RMO CSMS Project Use Cases

CSMS Order Fulfillment Subsystem	
Use cases	**Users/actors**
Ship items	Shipping
Manage shippers	Shipping
Create backorder	Shipping
Create item return	Shipping, customer
Look up order status	Shipping, customer, management
Track shipment	Shipping, customer, marketing
Rate and comment on product	Customer
Provide suggestion	Customer
Review suggestions	Management

CSMS Customer Account Subsystem	
Use cases	**Users/actors**
Create/update customer account	Customer, customer service representative, store sales representative
Process account adjustment	Management
Send message	Customer
Browse messages	Customer
Request friend linkup	Customer
Reply to linkup request	Customer
Send/receive partner credits	Customer
View "mountain bucks"	Customer
Transfer "mountain bucks"	Customer

RMO CSMS Project Use Cases

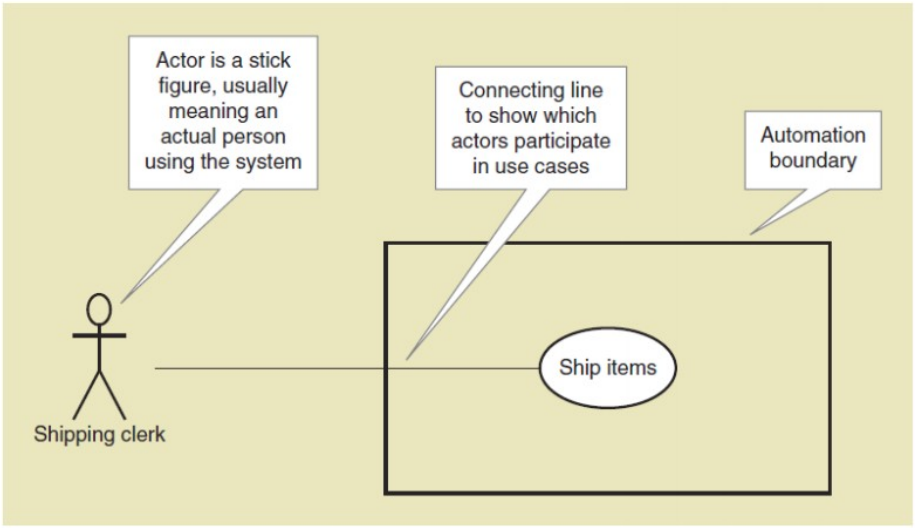
CSMS Marketing Subsystem	
Use cases	**Users/actors**
Add/update product information	Merchandising, marketing
Add/update promotion	Marketing
Add/update accessory package	Merchandising
Add/update business partner link	Marketing

CSMS Reporting Subsystem	
Use cases	**Users/actors**
Produce daily transaction summary report	Management
Produce sales history report	Management, marketing
Produce sales trends report	Marketing
Produce customer usage report	Marketing
Produce shipment history report	Management, shipping
Produce promotion impact report	Marketing
Produce promotional partner activity report	Management, marketing

Use Case Diagram

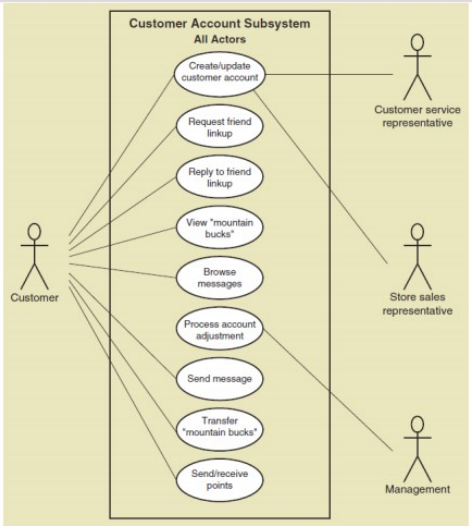
- A UML model used to graphically show uses cases and their relationship to actors
- Recall UML is Unified Modelling Language, the standard for diagrams and terminology for developing info systems
- Actor is the UML name for end user
- **Automation Boundary -** the boundary between the computerized portion of the application and the user who operates the application

Use Case Diagrams Symbols



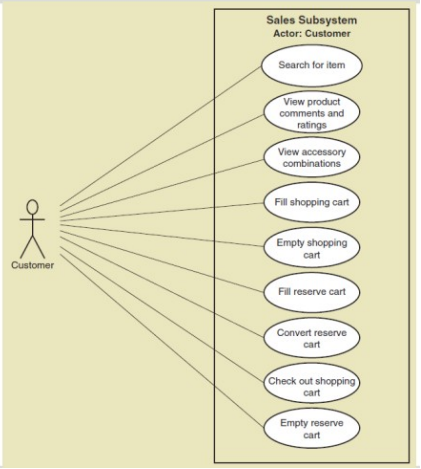
Use Case Diagrams

Draw for each subsystem



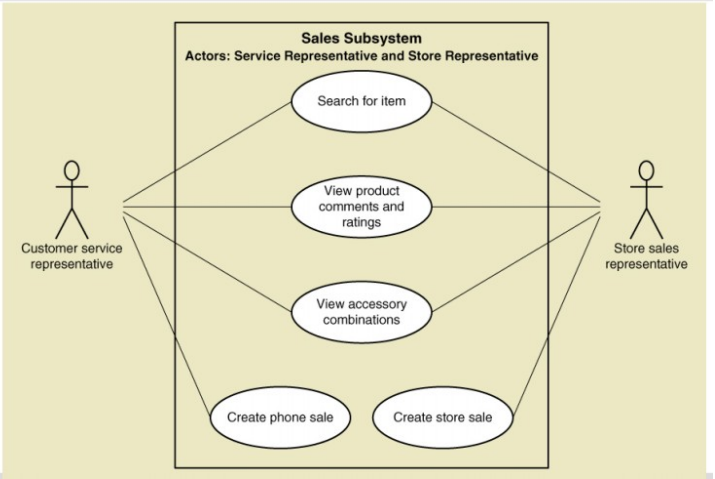
Use Case Diagrams

Draw for a single actor, such as customer



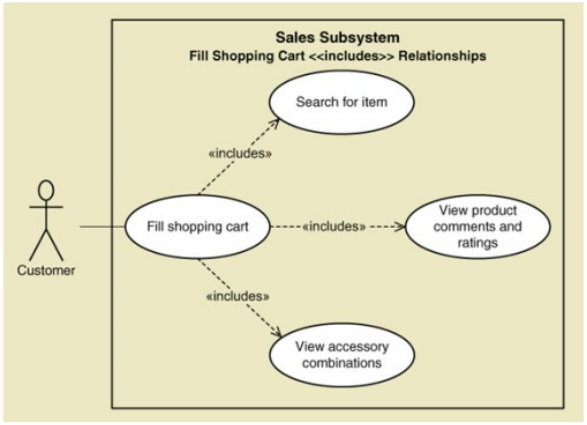
Use Case Diagrams

Draw for internal RMO actors



Use Case Diagrams— The <<Includes>> relationship

- A relationship between use cases where one use case is stereotypically included within the other use case— like a called subroutine. Arrow points to subroutine



Use Case Diagram: Steps

1. Identify all stakeholders and users who would benefit by seeing a use cae diagram
2. Determine what each stakeholder or user needs to review in a use case diagram: each subsystem, for each type of user, for use cases that are of interest
3. For each potential communication need, select the use cases and actors to show and draw the use case diagram.
4. Carefully name each use case diagram and then note how and when the diagram should be used to review use cases with stakeholders and users

SUMMARY

- Use cases are the functions identified, the activities the system carrier out in response to a user request
- Two techniques for identifying use cases are the user goal technique and the even decomposition technique
- The user goal technique begins with identifying end users called actors and asking what specific goals they have when interacting with the system
- The even decomposition technique begins by identifying events that occur that require the system to respond
- Three types of events include external ,temporal and state events
- Brief use case description are written for use cases
- The use case diagram is the UML diagram to show the use cases and the actors
- The use case diagram shows the actors, automation boundary, the use cases that involves each actor, and the <<includes>> relationship
- A variety of use case diagram are drawn depending on the presentation needs of the analysis

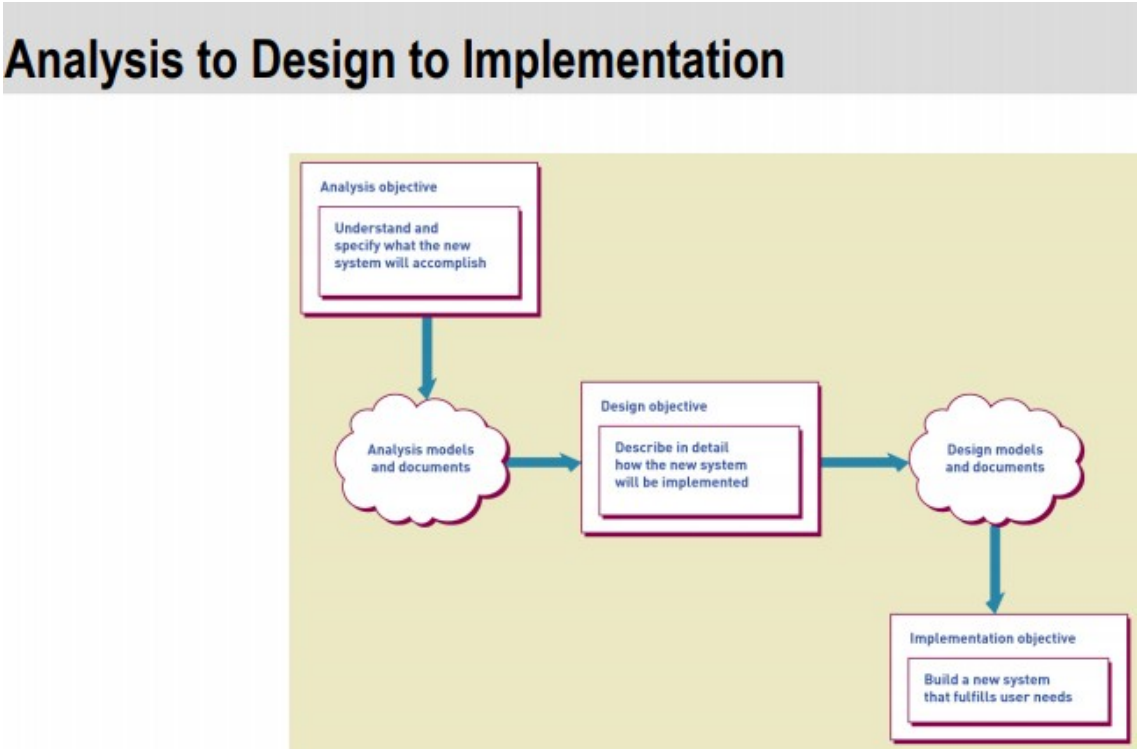
Week 5 – Domain Modelling

Week 6 – System Design Fundamentals

Systems Design Fundamentals

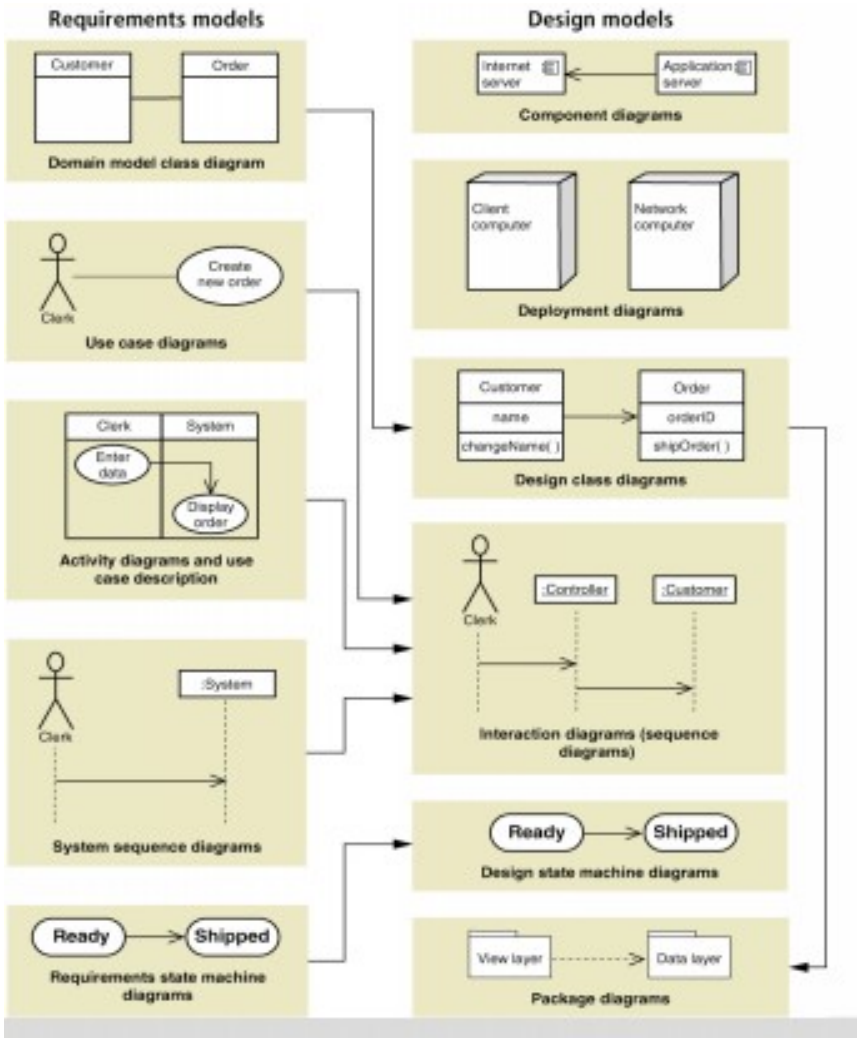
What is System Design?

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



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 solution and may require network



Design Activity

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

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?

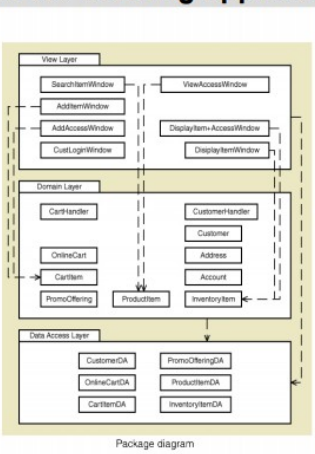
Describe the Environment

- Two key elements in the environment
  - Communication with external system
    - Message formats
    - Web and networks
    - Communication protocols
    - Security methods
    - Error detection and recovery
  - Conforming to an existing Technology Architecture
    - Discover and describe existing architecture

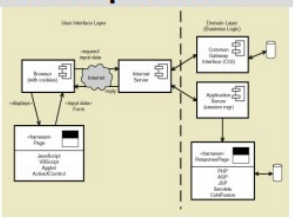
Design the Application Components

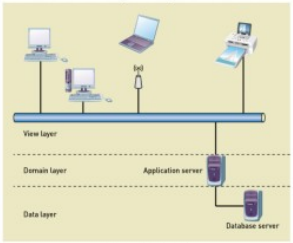
- App component is a well-defined unit of software that performs some functions
- Issues involve how to package components including
  - Scope and size - What are the functions, boundaries, interfaces?
  - Programming language - What are the accepted languages?
  - Build or buy - Is an acceptable version available to purchase?

Typical models for defining application components



Package diagram

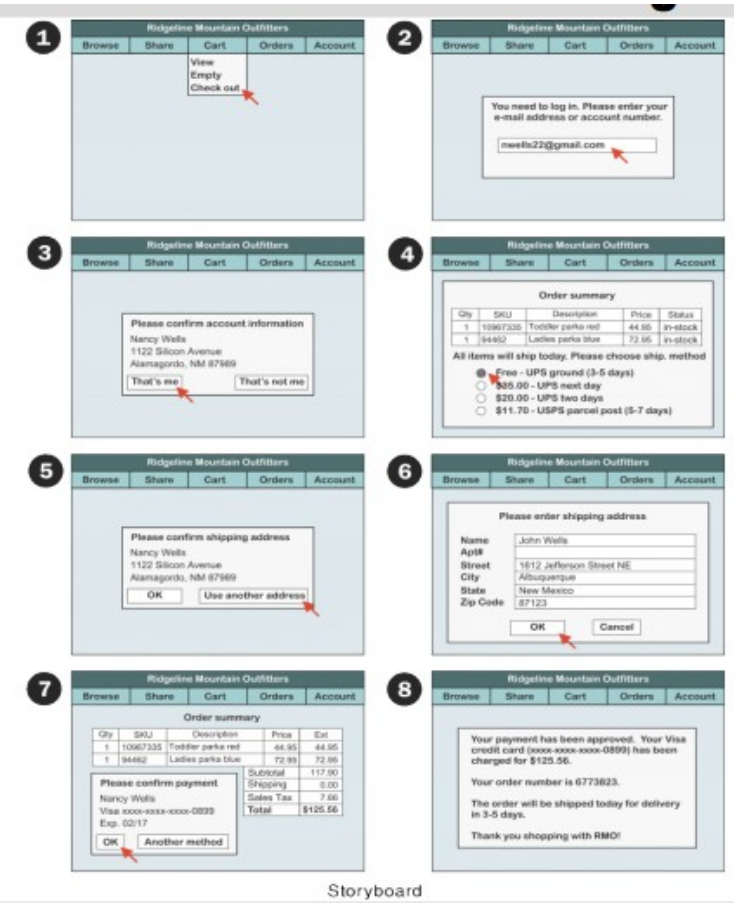

Component diagram

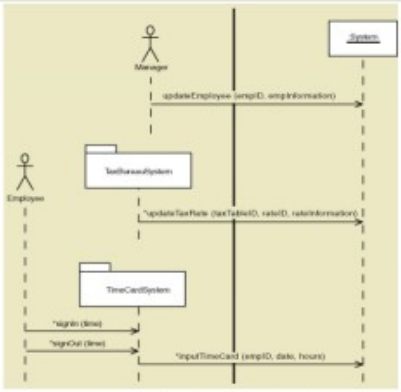


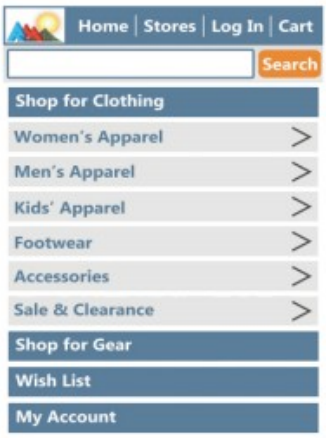
Deployment diagram

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 environment







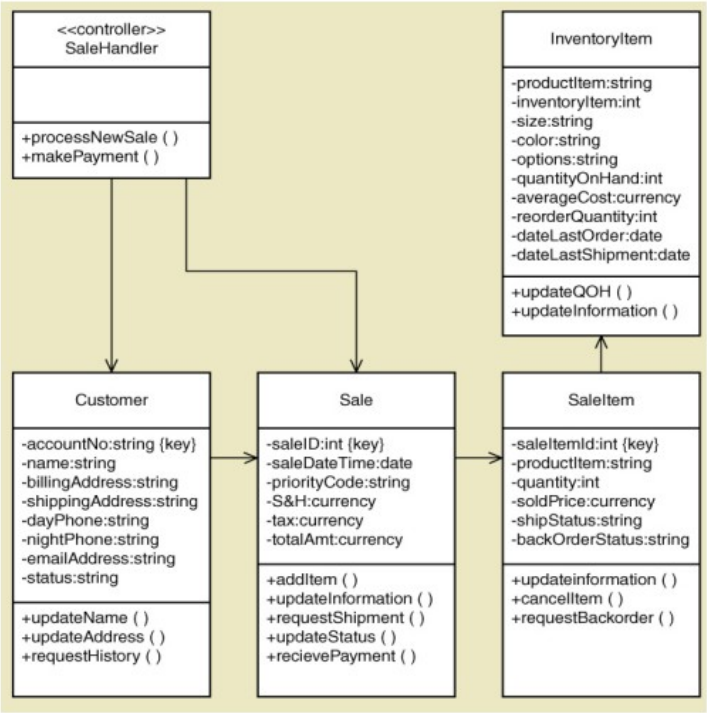
Design the Database

- Requires converting the data model to a rational database
- Requires addressing of many other technical issues
  - Throughput and response time
  - Security

Design Software Classes and Methods

- AKA detailed Design
- Model building activity
  - Design class diagram
  - Sequence diagram
  - State-machine Diagram

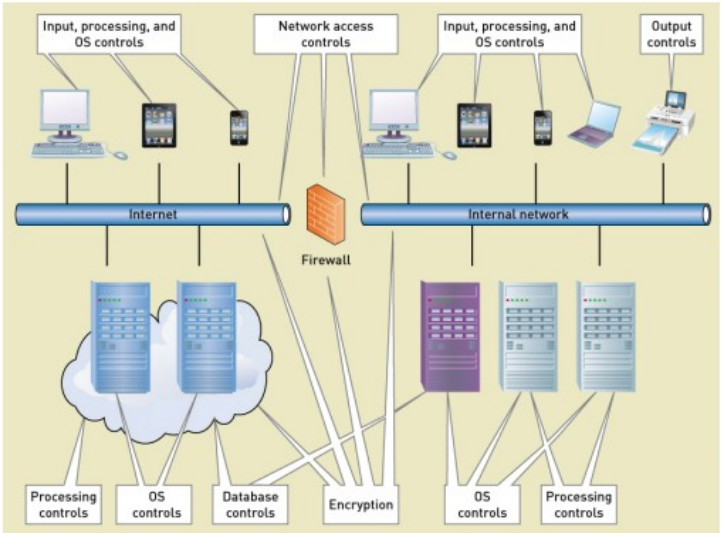
Typical Design Class Diagram with attributes and methods



System Controls and Security

- Integrity Controls
  - Controls that maintain integrity of inputs, outputs and data and programs
- Security Controls
  - Controls that protect the assets from threats, internal and external

Integrity and Security Controls



Designing Integrity Control

- Integrated into application programs and DMBS
- Objectives of integrity controls
    - Ensure that only appropriate and correct business transaction are accepted
    - Ensure that transactions are recorded and processed correctly
    - To protect and safeguard assets such as the database

Input Controls

- Prevent invalid and erroneous data from entering the system
- Value limit controls
    - Check the range of inputs for reasonableness
- Completeness controls
    - Ensure that all the data has been entered
- Data validation controls
    - Ensure that specific data values are correct
- Field combination controls
    - Ensure data is correct based on relationship between fields

Outputs Controls

- To ensure that output arrives at proper destination and is accurate, current and complete
- e.g.
    - Physical access to printer 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

- Protect data and systems from danger
    - Databases
    - Hardware
    - Software app
    - Network
- On-site vs off-site copy

Fraud Prevention

- Critical to prevent internal fraud, embezzlement, or loss
- Fraud Triangle
    - Opportunity
    - Motive
    - Rationalization

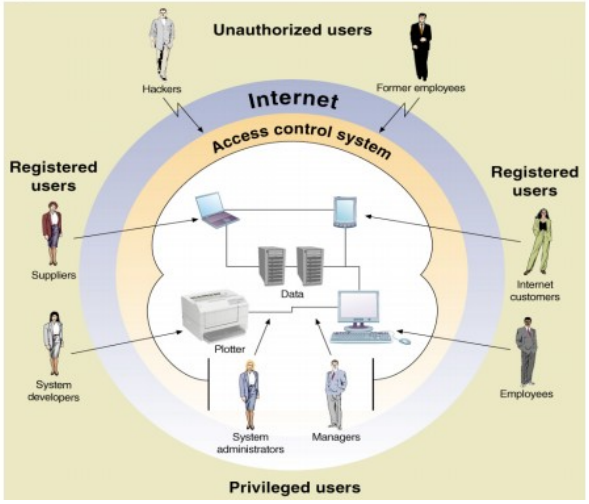
Fraud Risk – Factors and Techniques

Factors affecting fraud risk	Risk-reduction techniques
Separation of duties	Design systems so those with asset custody have limited access to related records. Also, ensure that no one has sufficient system access to commit and cover up a fraud.
Records and audit trails	Record all transactions and changes in asset status. Log all changes to records and databases, and restrict log access to a few trusted persons.
Monitoring	Incorporate regular and systematic procedures to review records and logs for unusual transactions, accesses, and other patterns.
Asset control and reconciliation	Limit physical access to valuable assets, such as inventory, and periodically reconcile physical asset counts with related records.
Security	Design security features into individual systems and supporting infrastructure. Review and test security features frequently. Use outside consultants to conduct penetration testing attack and fraud vectors from external and internal sources.

Designing Security Controls

- Protect all assets against external threats
- Other objectives
  - Protect and maintain a stable, functioning operating environment 24/7
  - Protect info and transaction during transmission across networks and internet
- Access Control - Limit a person's ability to access servers, files, data, applications
  - Authentication - to identify users
    - Multifactor authentication
  - Access control list - list of valid users
  - Authorization - authenticated user's list of permission level of each resource
- Registered User - those with authorization
- Unauthorized Users - anyone not registered
- Privileged Users - those that maintain lists and systems

Types of users

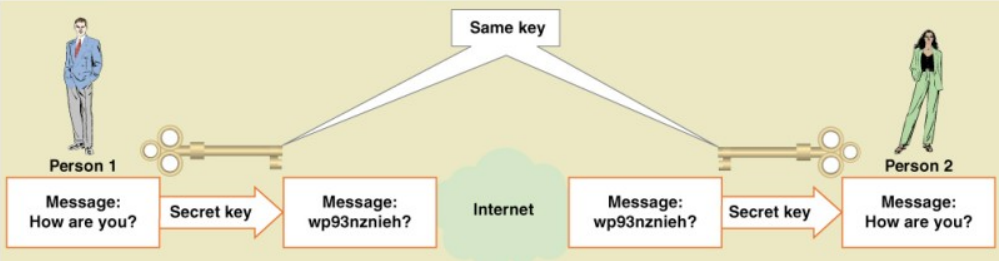


Data Encryption

- 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 - Math transformation of the data
- Encryption Key - Long data string that allows the same algorithm to produce unique encryption

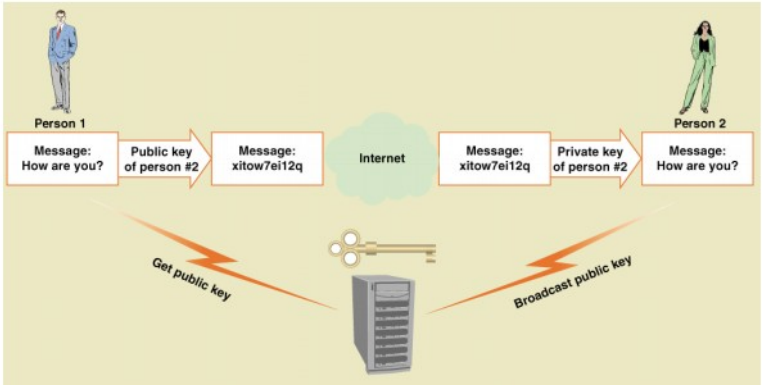
Symmetric Key Encryption

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



Asymmetric Key Encryption

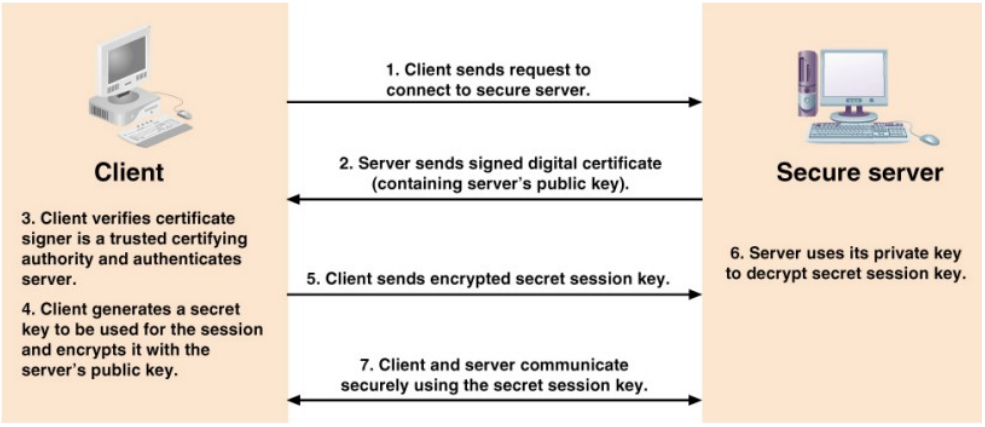
- Encryption method that uses different keys to encrypt and decrypt
  – AKA Public Key Encryption



Digital Signatures and Certificates

- Digital Signature - technique where a document is encrypted using a private key
  - Document is encrypted with private key, but then can only be decrypted with correct public key
- Digital certificate - An organization name and public that is encrypted and certified by an authorized third party
- Certifying Authority - the authorized third party

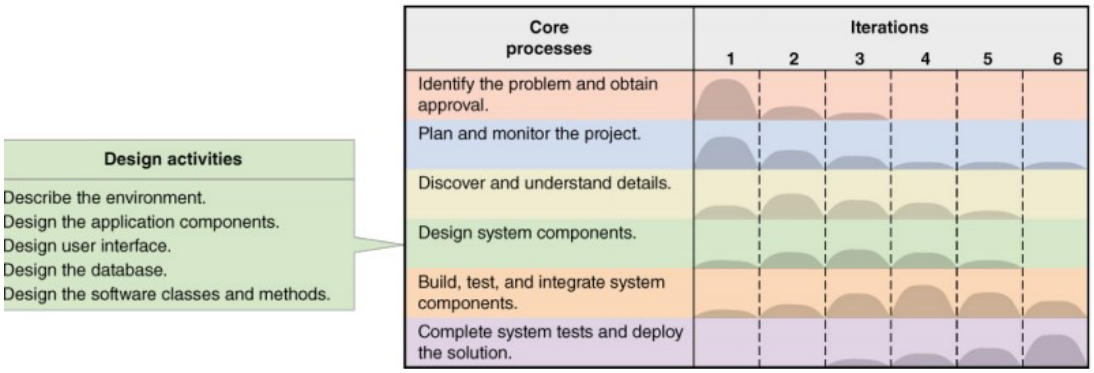
How a Digital Certificate is used



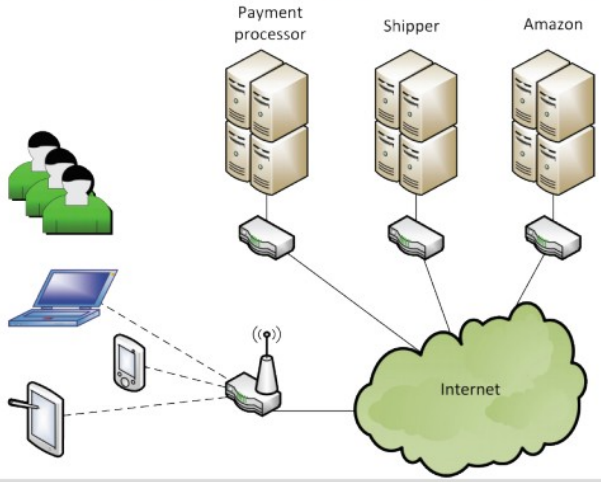
Secure Transaction

- Secure Sockets Layer (SSL) - standard set of protocols for authentication and authorization
- Transport layer security (TLS) - an internet standard equal to SSL
- IP Security (IPSec) - Internet Security protocol at a low-level transmission
- Hypertext Transfer Protocol Secure (HTTPS) - Internet standard to transmit Web pages

Activities of "Design System Components"



Simplified architecture for application (Amazon.com)

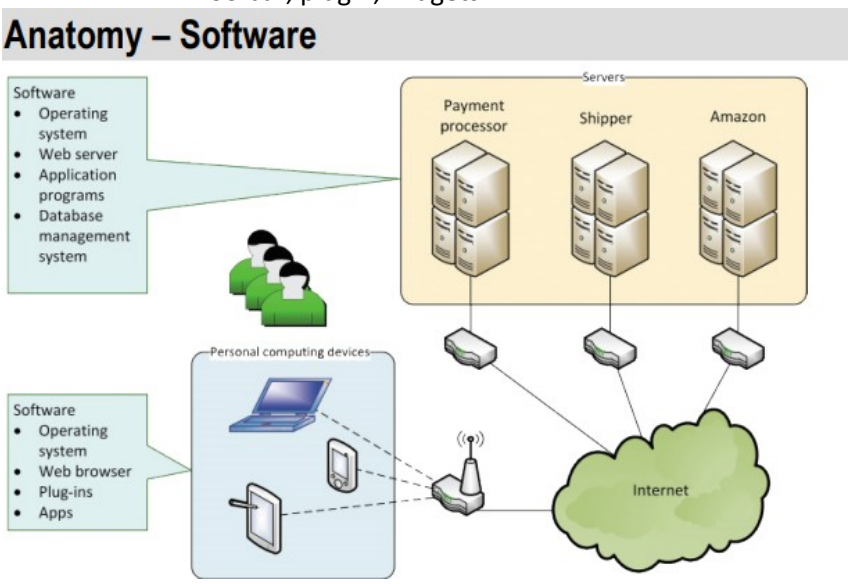


Anatomy - Networks

- Computer networks - hardware, software, transmission media
- Internet backbone
  - High capacity with high bandwidth trunk lines and large high-speed computers
  - Owned by government and telecom companies
- Local Area networks (LAN)
  - Small network for a single site
- WWW
  - All interconnected resource accessed through the internet
- Uniform Resource Locator (URL)
  - Identifier for the web to locate a particular resource
- Hyperlink
  - The URL of a resource embedded within another resource

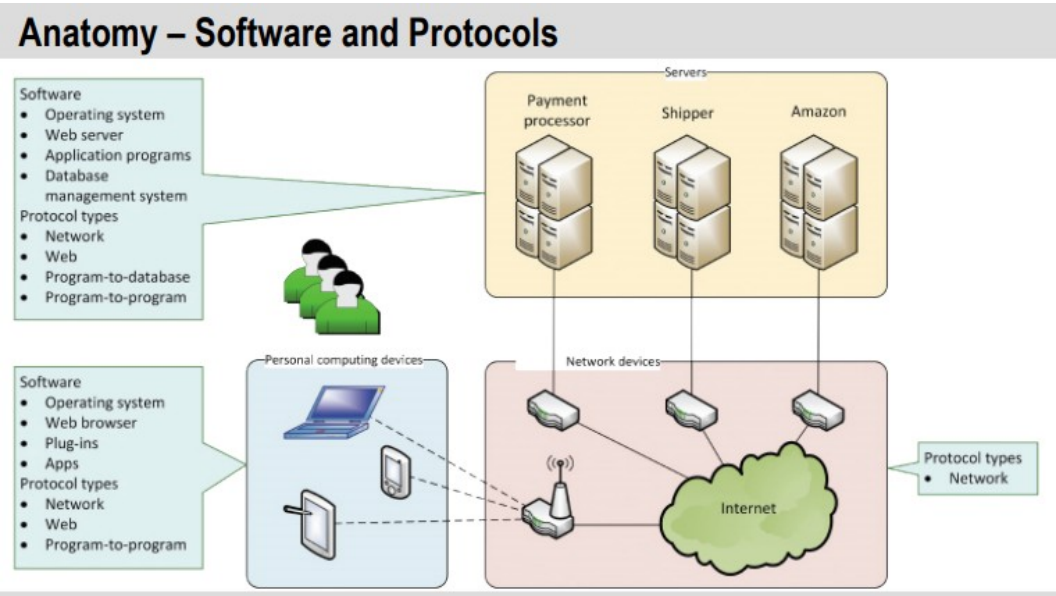
Anatomy - Software

- App Software - program that perform work for user
  - Either a custom app or web-based app
- App
  - A custom program usually for a laptop or smartphone
- System Software
  - Behind the scene software, works as glue to hold everything together
- Web-based App
  - Uses a web browser
  - Accessed through a URL
  - Resides on a web server
  - Uses standard IP protocols
- Embedded Software
  - Software app or function embedded with another app, such as within a browser or O/S
  - Toolbar, plugin, widgets



Anatomy - Protocols

- Protocols
  - Set of languages and rules to ensure communication and data exchange between hardware and software
- Network protocols
  - VPN
    - Creates a private network but on the internet by using secure tech and encryption



Anatomy - Web Protocols

- HTML
  - Protocols for the structure and content of a Web Page
- XML (Extensible mark-up language)
  - An extension of HTML that enables defining semantic of tags
- HTTP (Hypertext transfer protocol)
  - Defines format and content for transfer of Web Document
- HTTPS (Hypertext transfer protocol secure)
  - Encrypted and secure http transfer

Architectural Concepts

- Technology Architecture
  - Computer, network computer and hardware, and system software
- Application Architecture
  - The software program and their configuration

Software as a Service (SaaS)

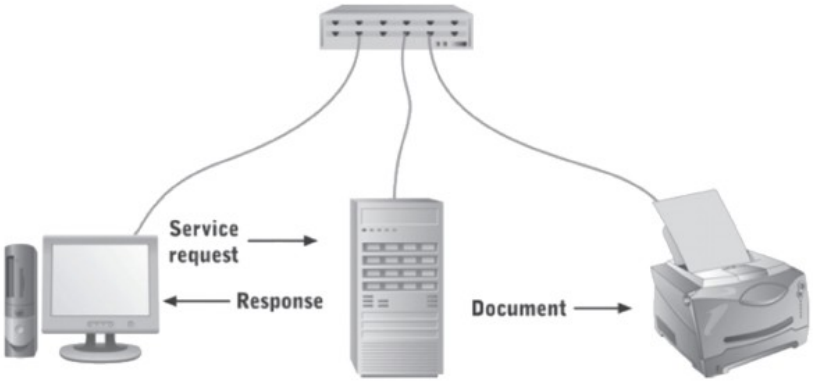
- No software is installed on the users device
- App services is remotely accessed
- User data is isolated and stored on common servers

Web Services

- 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

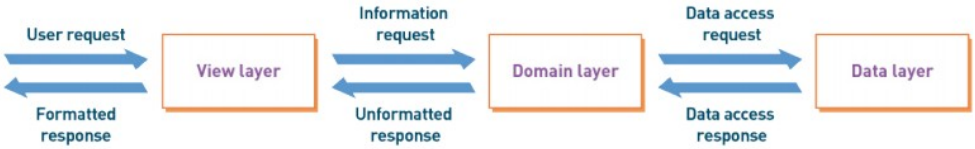
Distributed Architectures

- Client/Server architecture
  - Software design with part of the application on a server and part on the client

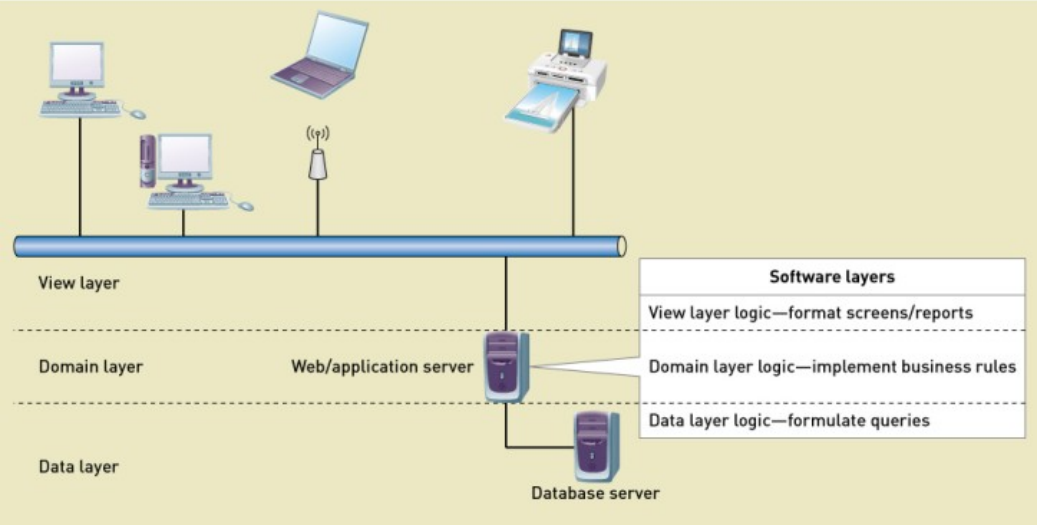


Distributed Architecture

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



Three Layer Architecture



Interoperability

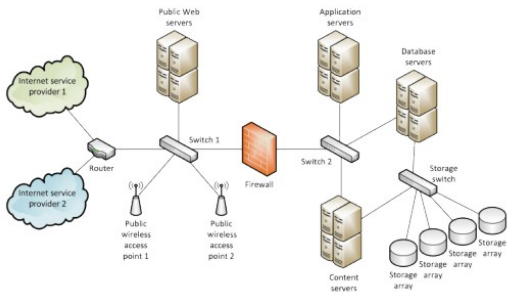
- The ability of an app to interact with other software
- Important characteristic is current development project
  - Understand the environment
  - Reuse software existing components
  - Build components considering interoperability
  - Combine all components into a solution system

Diagrams for System Architectures

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

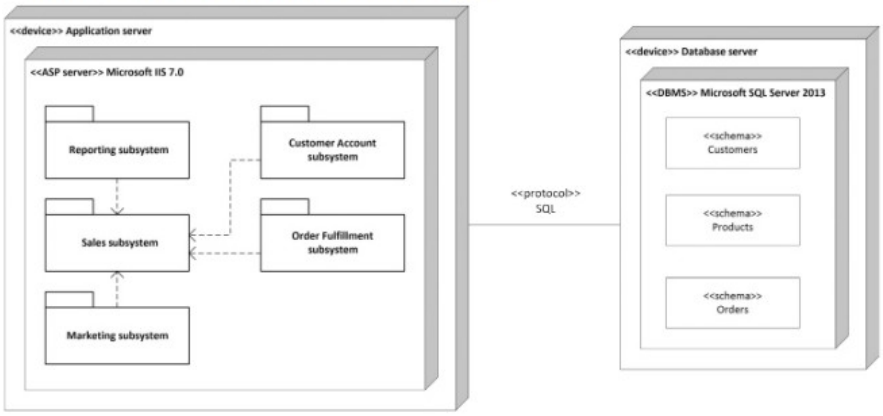
Diagrams for System Architecture

- Network Diagrams
  - How the application software is deployed across the hardware and system software



Diagrams for System Architecture

- Deployment Diagrams
  - How the components of a network are interconnected



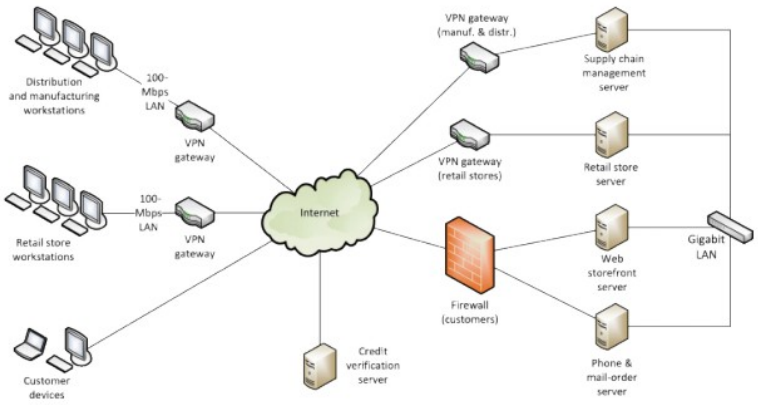
Describing the Environment

- Key questions to help describe accurately
  - What are the key features of existing or new environment
    - O?S, system software, networks, tools
  - What are the external systems or DBMS
    - What kind of interaction
    - What is the data
    - What are the protocols
    - What kind of security
  - What devices will be required
    - Protocols for devices
    - Security
    - What API
  - What user interface technology will be used
    - Where and who are users, and what will be used
    - What hardware and devices
    - What client O/A will be used
    - Security requirements
    - What API are needed

RMO CASE

RMO Environment - Existing

Current environment prior to new development

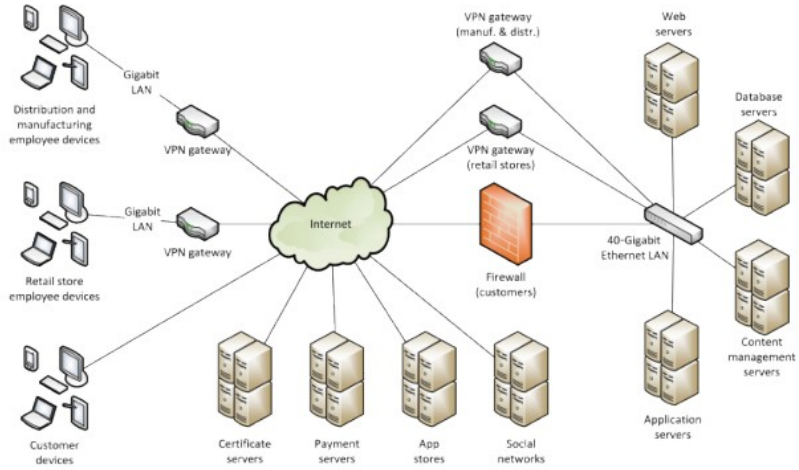


MONASH 64

RMO ENVIRONMENT - PROPOSEED

- More mobile device and app
- Web app software and content
- Social networking app
- Security issue
- External hosting of portions

RMO Environment - Proposed



Designing Application Components

- Application Component Boundaries
  - Which component perform which function
  - How to group functions 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 Architecture

Grouping by customer actor – part 1

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	

RMO CSMS Application Architecture

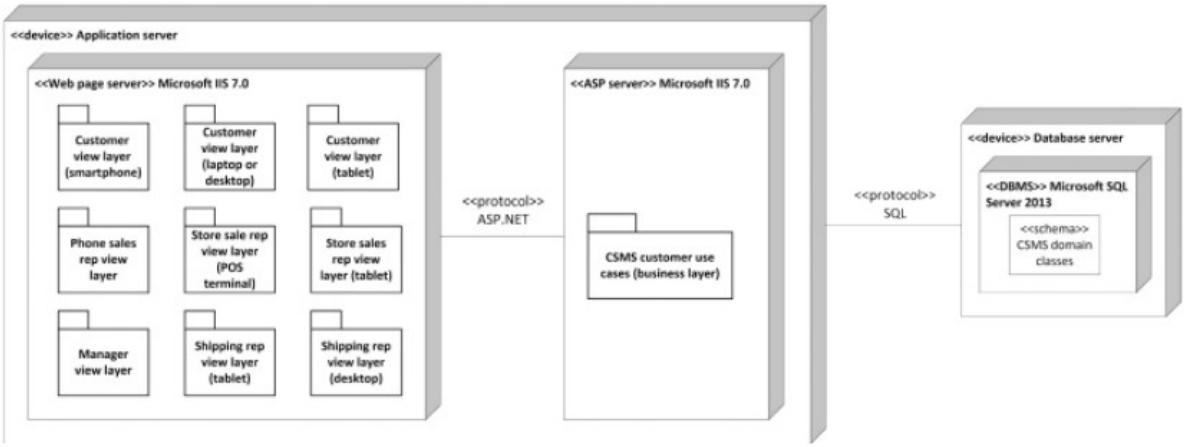
Grouping by customer actor – part 2

Use case	User/actor	Domain class(es)	Event(s)	Group
Fill shopping cart	Customer	ProductItem, InventoryItem, CartItem, OnlineCart	Customer request, usually after sale completed	D
Empty shopping cart	Customer	ProductItem, InventoryItem, CartItem, OnlineCart	Customer request while shopping online	
Check out shopping cart	Customer	ProductItem, InventoryItem, CartItem, OnlineCart, SaleItem, Sale, SaleTrans	Customer request while shopping online	
Fill reserve cart	Customer	ProductItem, InventoryItem, CartItem, OnlineCart	Customer request while shopping online	
Empty reserve cart	Customer	ProductItem, InventoryItem, CartItem, OnlineCart	Customer request while shopping online	
Convert reserve cart	Customer	ProductItem, InventoryItem, CartItem, OnlineCart	Customer request while shopping online	
Rate and comment on product	Customer	Customer, ProductComment, ProductItem	Customer request, usually after sale completed	E
Provide suggestion	Customer	Customer, Suggestion	Customer request while shopping online	
Send message	Customer	Customer, Message	Customer request while shopping online	
Browse messages	Customer	Customer, Message	Customer request while shopping online	
Request friend linkup	Customer	Customer, FriendLink	Customer request while shopping online	
Reply to linkup request	Customer	Customer, FriendLink	Customer request while shopping online	
Send/receive partner credits	Customer	Customer, CustPartnerCredit, PromoPartner	Customer request while shopping online	
View "mountain bucks"	Customer	Customer, Sale	Customer request while shopping online	
Transfer "mountain bucks"	Customer	Customer, Sale	Customer request while shopping online	

MONASH

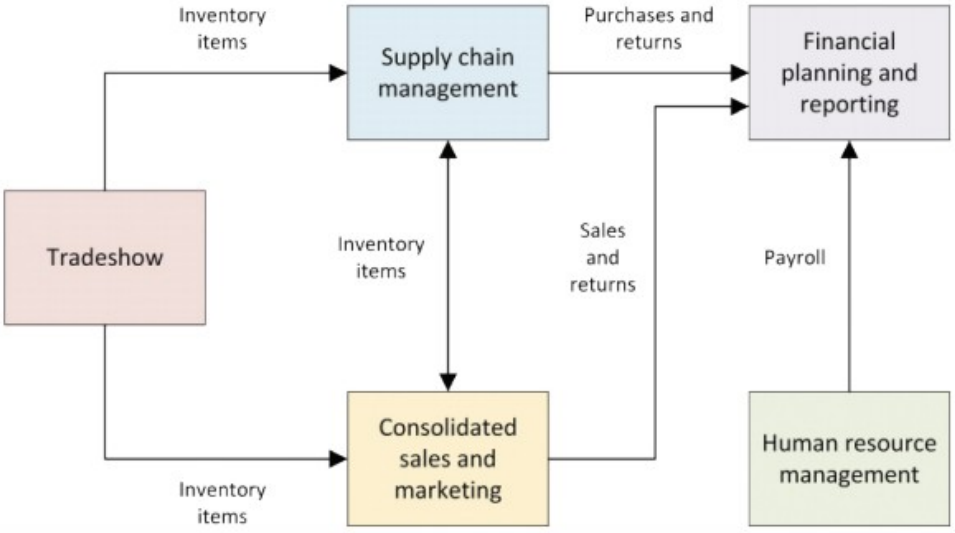
RMO CSMS Deployment Diagram

Three-layer design with user components grouped by user functions



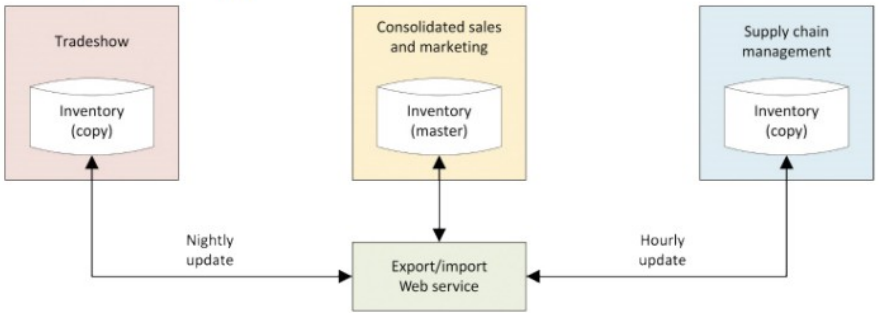
RMO CSMS Component Integration

Subsystem integration and data flows



RMO CSMS Data Ownership

- Who "owns" the data
  - System of record
    - What system is responsible to maintain the data
    - What system has a copy or can access the data



SUMMARY

- Anatomy of a Modern Info System
- Architectural Concepts
  - SaaS
  - Web Services
  - Distributed Architectures
    - Client/server and three-layer architecture
- Interoperability
  - Getting all the components to work together
- Architectural Diagrams
  - Location
  - Network
  - Deployment
- Describing the environment
  - External system
  - Technology architecture
  - Key question requiring answers
- Designing application components
  - Application components boundaries
  - Grouping functions into components
  - System of record - who owns the data

Week 7 – Object-Oriented Design and UML Modelling

Week 8 – Object-Oriented Design and UML Modelling (Part 2)

Designing the User Interface

User Experience

- UI - inputs and outputs that directly involve human user/actor
- UI design must focus on User Experience
- Called User-Centred Design
  - Focus early on user and their work
  - Evaluate designs to ensure usability
  - Use iterative development
- Usability is the objective

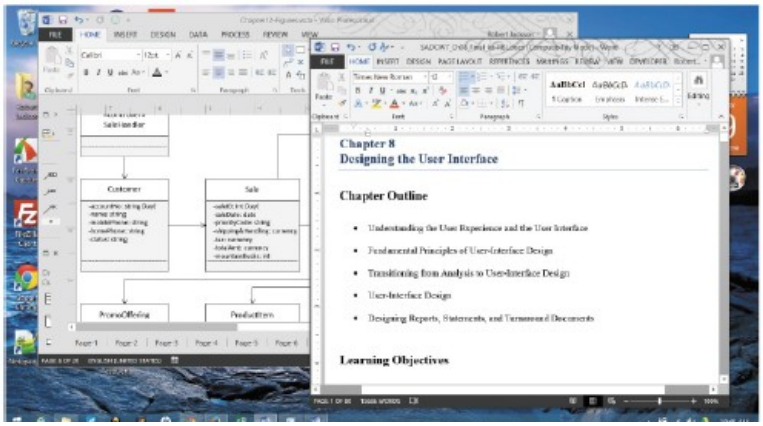
Metaphors of Human Computer Interaction

- **Direct Manipulation**
  - Metaphor in which objects on a display are manipulated to look like physical objects or graphic symbols that represent them (icons)
- **Desktop**
  - Metaphor in which the visual display is organized into distinct regions, with al arge empty workspace in the middle and a collection of tool icons around the perimeter
- **Document**
  - Metaphor in which data is visually represented as paper pages or forms
- **Dialog**
  - Metaphor in which user and computer accomplish a task by engaging in a convo or dialog via text, voice or tools such as labelled buttons

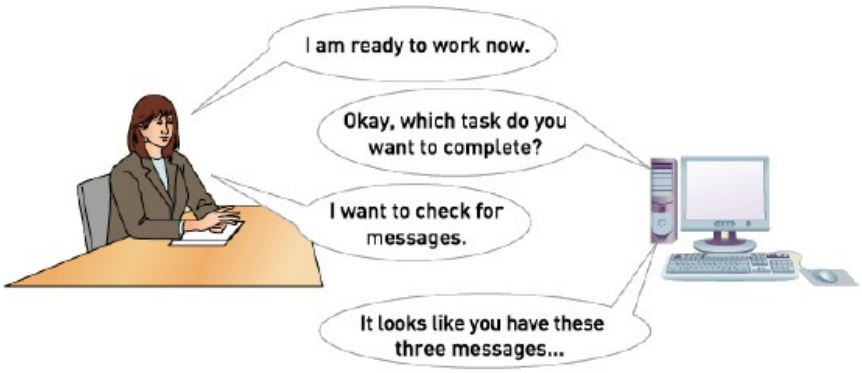
Metaphor Details

Metaphor	Description	Example
Direct manipulation	Manipulating objects on a display that look like physical objects (pictures) or that represent them (icons)	The user drags a folder icon to an image of a recycle bin or trash can to delete a collection of files.
Desktop	Organizing visual display into distinct regions, with a large empty workspace in the middle and a collection of tool icons around the perimeter	At computer startup, a Windows user sees a desktop, with icons for a clock, calendar, notepad, inbox and sticky notes (the computer interface version of a physical Post-It note).
Document	Visually representing the data in files as paper pages or forms; these pages can be linked together by references (hyperlinks)	The user fills in a form field for a product he or she owns, and the manufacturer's Web site finds and displays the product's manual as an Adobe Acrobat file, which contains a hyperlinked table of contents and embedded links to related documents.
Dialog	The user and computer accomplishing a task by engaging in a conversation or dialog by using text, voice, or tools, such as labeled buttons	The user clicks a button labeled "troubleshoot" because the printer isn't working. The computer prints questions on the display, and the user responds by typing answers or selecting responses from a printed list.

Direct Manipulation, Desktop, and Document Metaphors On One Screen



Dialog Metaphor



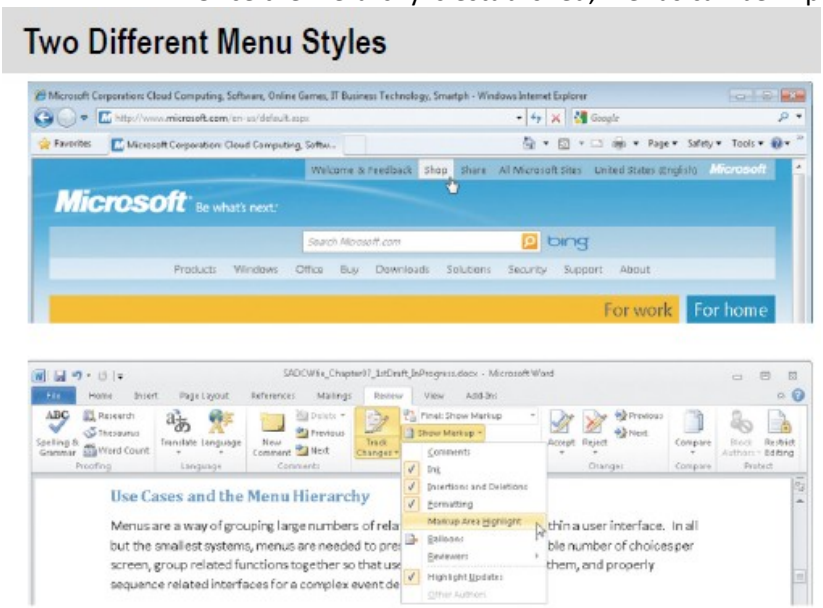
Principles of UI Design

- Human Interface Objects
  - Affordance - the appearance of the object suggests it's functions
  - Visible with feedback
  - Both visible on the display and provides a response to a user action (feedback)
  - Good example is radio buttons and check boxes (ac/heater button, click and graphic change)
  - Consistency
    - Across platforms
    - Within a suite of applications
    - Within a particular app
  - Continuity
    - Consistency across releases over time
  - Discoverability
    - To help users discover hidden features or objects
    - Active discovery - mouse hovers, pop-ups, tool tips
    - Visual diagram to guide users

- Closure
  - Closure on dialogues - end of a series of actions
  - Protect users work - at end and for partially complete work
  - Provide undo to reverse action
- Readability and navigation
  - Readable text for all users
  - Clear navigation
  - Reverse navigation - way out
- Usability and Efficiency
  - Shortcut keys
  - Meaningful error message
  - Simplicity - KISS

Transitioning from Analysis to UI Design

- Use case and the menu hierarchy
  - We design use case by use case
  - Menus are typically way to organize access to use case functionality
  - Different types of users might have different menus
  - Useful to design an overall menu hierarchy and then subset for different users
  - Once the hierarchy is established, menus can be implemented in a variety of ways



Some RMO Use Cases

- Grouped by Actor and Subsystem

Subsystem	Use case	Users/actors
Sales	Search for item	Customer, customer service representative, store sales representative
Sales	View product comments and ratings	Customer, customer service representative, store sales representative
Sales	View accessory combinations	Customer, customer service representative, store sales representative
Sales	Fill shopping cart	Customer
Sales	Empty shopping cart	Customer
Sales	Check out shopping cart	Customer
Sales	Fill reserve cart	Customer
Sales	Empty reserve cart	Customer
Sales	Convert reserve cart	Customer
Sales	Create phone sale	Customer service representative
Sales	Create store sale	Store sales representative
Order fulfillment	Ship items	Shipping
Order fulfillment	Manage shippers	Shipping
Order fulfillment	Create backorder	Shipping
Order fulfillment	Create item return	Shipping, customer
Order fulfillment	Look up order status	Shipping, customer, management
Order fulfillment	Track shipment	Shipping, customer, marketing
Order fulfillment	Rate and comment on product	Customer
Order fulfillment	Provide suggestion	Customer

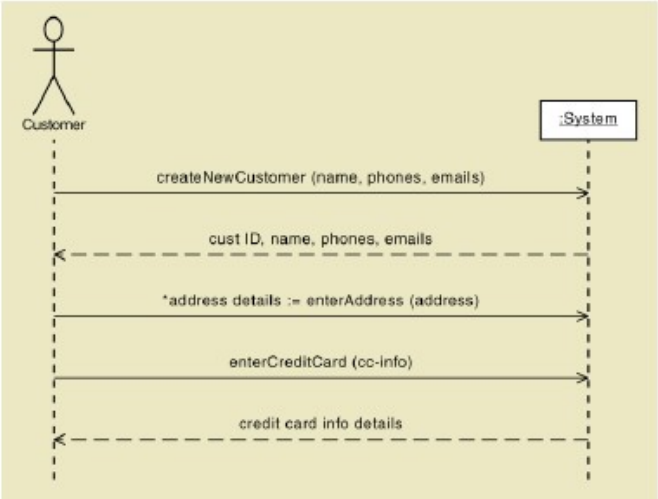
RMO Use Cases

- Grouped into First Cut Menu Hierarchy

Menu description	Menu choices (use cases)	Intended user(s)
Shopping cart functions (primary or reserve)	Search for item View product comments and ratings View accessory combinations Switch carts (primary to reserve or vice versa) Fill shopping cart Empty shopping cart Check out shopping cart	Customer
Sale creation	Search for item View product comments and ratings View accessory combinations Create sale	Customer service and store sales representatives
Order shipment	Ship items Manage shippers Create backorder Create item return Look up order status Track shipment	Customer service and store sales representatives
Customer order control	Look up order status Track shipment Create item return Rate and comment on product Provide suggestion	Customer

Analysis Models and Input Forms

- SSD defines input messages, which indicates what for



Sample Customer Form

- First draft of RMO Customer Form from SSD information

Dialogs and Storyboards

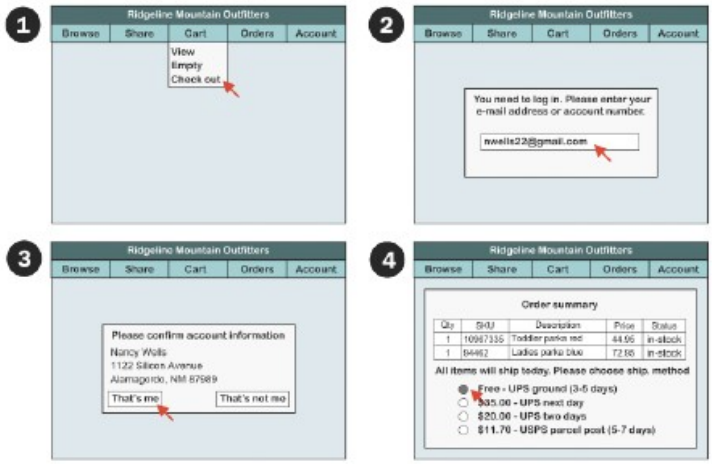
- For each use case, think of the natural flow of a dialog between user and computer
  - Based on the flow of activities in use case description and/or the SSD
  - Use natural language to emphasize feedback to user
  - Create storyboard of the dialog, showing the sequence of sketches of the screen each step of the dialog
  - Review the storyboard with user

From Dialog to Storyboard (part 1)

Use case *Check out shopping cart*

SYSTEM:	What would you like to do?
USER:	I'd like to check out.
SYSTEM:	Okay. What is your e-mail address or account number?
USER:	My e-mail address is nwells22@gmail.com.
SYSTEM:	Fine. You are Nancy Wells at 1122 Silicon Avenue. Correct?
USER:	Yes.
SYSTEM:	All items in your cart are in stock and can be shipped today. Because your order subtotal is over $100, you qualify for free UPS ground shipping (3–5 days). Other shipping options include next day ($35.00), two day ($20.00), and USPS parcel post ($11.70, 5–7 days). How would you like your items to be shipped?

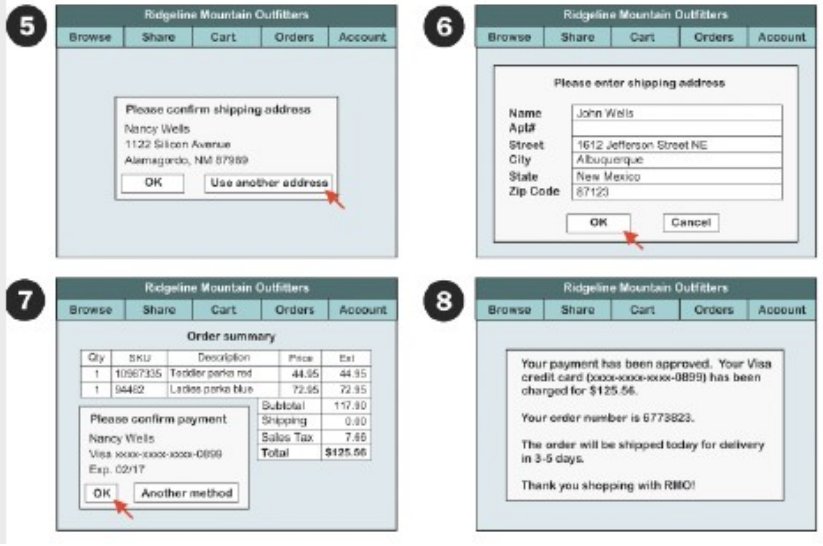
From Dialog to Storyboard (part 1)



From Dialog to Storyboard (part 2)

USER:	Free UPS ground shipping.
SYSTEM:	We have a shipping address on file for you (1122 Silicon Avenue). Do you want your order shipped to that address?
USER:	No.
SYSTEM:	To what address would you like the order shipped?
USER:	John Wells, 1612 Jefferson Street NE, Albuquerque, NM 87123.
SYSTEM:	Okay, the total charge is $125.56 ($117.90 item subtotal plus $7.66 sales tax). Shall I charge that amount to your credit card on file (a Visa with an account number ending in 0899)?
USER:	Yes.
SYSTEM:	Your payment has been approved, and your order is being prepared for shipment. A confirmation e-mail has been sent to you, and another will be sent with a shipment tracking number when the order is shipped later today. Can I help you with anything else?
USER:	No.

From Dialog to Storyboard (part 2)



UI Design

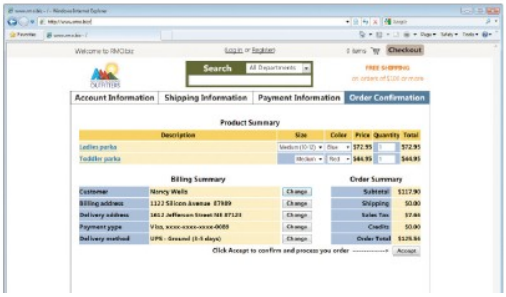
- Is the system a custom app or browser based?
- What kind of devices will the UI need to support?
- What OS will the UI run on?
- Desktop and Laptop UI
  - Layout and formatting
    - Purposeful design, location and grouping, no sloppiness or error
  - Data Entry
    - Text box, list box, combo box, radio button, check box
    - Include online editing to minimize errors
  - Navigation and visibility
    - Minimize, maximize, close, scroll bar, resize

Consideration for Web-Based App

- Layout and formatting
  - Various browser default setting
  - Impact of online advertising
- Data entry and user actions - client side programming
- Navigation and visibility - complete yet simple

RMO Checkout Page

- RMO shopping cart checkout
  - Simple, easy to read



Smartphones and small mobile devices

- Challenges
  - Small screen, keyboard, limited network capacity
- Layout and formatting
  - Rotating view, resizing, visible navigation, scrolling
- Data entry and user action
  - Fat fingers and accidental touches
- Navigation and visibility
  - Show site map
  - Use action bar
  - Visual clues
  - Back buttons

Designing Reports, Statements and Turnaround Documents

- Report Types
  - Detailed reports - contain specific info on business transaction
  - Summary report - summarize detail or recap periodic activity
  - Exception report - provide detail or summary info about transaction or operation results that fall outside a predefined normal range of values
  - Executive report - assess overall org health and performance
  - Electronic Report
    - Drill down - to view additional detail related to an item
    - Linking report to other reports
    - View data grouped various categories
  - Graphical and multimedia reports
    - Charting and graphic of data

SUMMARY

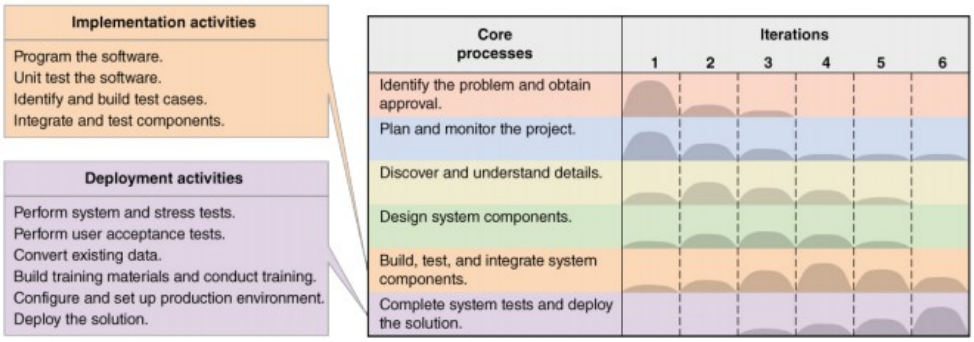
- UI involves direct user interaction with the system
- Design of UI has long history as HCI and relies on user-centred design, which focuses early on users, evaluated design to ensure usability and uses iterative development
- Metaphors are used to think about the nature of the UI, and they include direct, desktop, document and dialog
- Key UI concepts include affordance and visibility for controls
- Other key principles include consistency, shortcuts, feedbacks, dialog closure, error handling, and reversal of action
- Use cases are organized into one or more menu hierarchies to arrange functionality for users
- Dialog and storyboard are used to design the interaction for each use case based on use case flow of activities and SSD
- Guidelines are available for designing Windows, Web Browser and handheld devices
- Designing inputs involves identifying devices and mechanisms, identifying inputs and the data content, and determining the controls necessary
- Designing outputs includes designing detailed reports, summary report, exception report and executive reports
- Electronic reports and other outputs can include drill down, graphic and multimedia

Week 9 – Designing the User Interface

Week 10 – Deploying the New System

Deploying the New System

Implementation and Deployment Activities



Testing Concepts

- Testing
  - The process of examining a component, subsystem, or system to determine its operational characteristics and whether it contains any defect
- Test Case
  - Formal description of a starting state, one or more events to which the software must response or ending state
    - Defined based on well understood functional and non-functional requirements
    - Must test all normal and exception situation
- Test Data
  - A set of starting state and events used to test a module, group of modules, or entire system
    - Data will be used for a test case

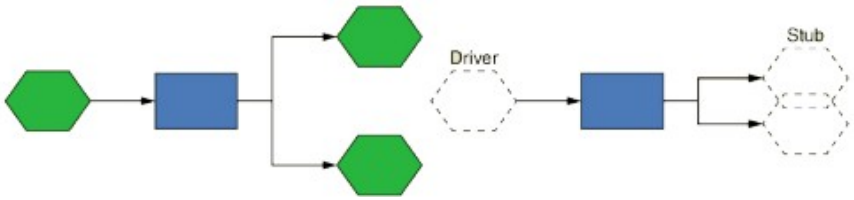
Most common types of tests

Test type	Core process	Need and purpose
Unit testing	Implementation	Software components must perform to the defined requirements and specifications when tested in isolation—for example, a component that incorrectly calculates sales tax amounts in different locations is unacceptable.
Integration testing	Implementation	Software components that perform correctly in isolation must also perform correctly when executed in combination with other components. They must communicate correctly with other components in the system. For example a sales tax component that calculates incorrectly when receiving money amounts in foreign currencies is unacceptable .
System and stress testing	Deployment	A system or subsystem must meet both functional and non-functional requirements. For example an item lookup function in a Sales subsystems retrieves data within 2 seconds when running in isolation, but requires 30 seconds when running within the complete system with a live database.
User acceptance testing	Deployment	Software must not only operate correctly, but must also satisfy the business need and meet all user "ease of use" and "completeness" requirements—for example, a commission system that fails to handle special promotions or a data-entry function with a poorly designed sequence of forms is unacceptable.

Unit Testing

- Test if an individual method, class or component before it is integrated with other software
- Driver
  - A method or class developed for unit testing that simulates the behaviour of a method that sends a message to the method being tested
- Stub
  - Method or class developed for unit testing that simulates behaviour of a method invoked that hasn't yet been written.

- Driver and stub components



Integration Testing

- Test behaviour of a group of methods, classes or components
  - Interface combability - E,G - one method passes a parameter of the wrong data type to another method
  - Parameter values - A method is passed or returns a value that was unexpected, such as a negative number for a price
  - Run-time expectations - A method generates an error, such as 'out of memory' or 'file ready in use', due to conflicting resource needs
  - Unexpected state interactions - the state of two or more objects interact to cause complex failures, as when an OlineCert method operates correctly for all possible Customer object states expect one

- Integration testing of OO software is very complex because an OO program consists of a set of interacting objects
  - Methods can be called by many other methods, and the calling methods may be distributed across many classes
  - Classes may inherit methods to be called is dynamically determined at run time based on the number and type of message parameters
  - Objects can retain internal variables values between call. The response to two identical calls may be different due to state changes that result from the first call or occurs between calls

- Required Procedures
  - Build and unit test the components to be integrated
  - Create test data - detailed test data, must be coordinated between deployers
  - Conduct the integration test - Assign resource and responsibilities. Plan frequency and procedures
  - Evaluate the test results - identify valid and invalid reponses
  - Log the test results - log valid test runs

- Correct the code and retest

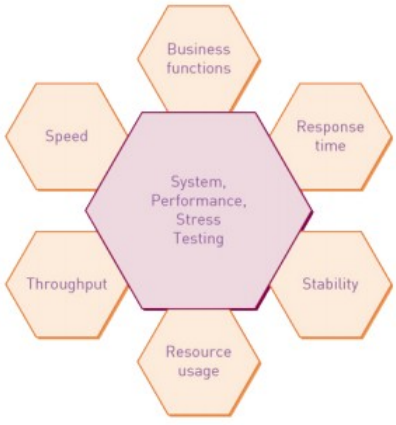
System, Performance, and Stress Testing

- System test – an integration test of an entire system or independent subsystem
  - Can be performed at the end of each iteration
  - Can be performed more frequently
  - Build and smoke test – a system test that is performed daily or several times a week
    - The system is completely compiled and linked (built), and a battery of tests is executed to see whether anything malfunctions in an obvious way ("smokes")
    - Automated testing tools are used. Catches any problems that may have come up since the last system test

System, Performance, and Stress Testing

- Performance test or stress test – an integration and usability test that determines whether a system or subsystem can meet time-based performance criteria
  - Response time – the desired or maximum allowable time limit for software response to a query or update
  - Throughput – the desired or minimum number of queries and transactions that must be processed per minute or hour

System, Performance, and Stress Testing



User Acceptance Testing (UAT)

- A system test performed to determine whether the system fulfils user requirements
- Amy be performed near ending of project
- Very formal activity in most development projects
- Details of acceptance tests are sometimes included in the RFP and procurement contact
- Plan the UAT
  - Should be done early in project
  - Test cases for every use case and user story
  - Identify condition to verify that the system supports the use case accurately and completely
- Prep and Pre-UAT activities
  - Develop test data
  - Plan and schedule specific tests
  - Set up test environment
- Manage and execute the UAT
  - Assign responsibilities
  - Document and tract results
  - Rework the plan for re-testing

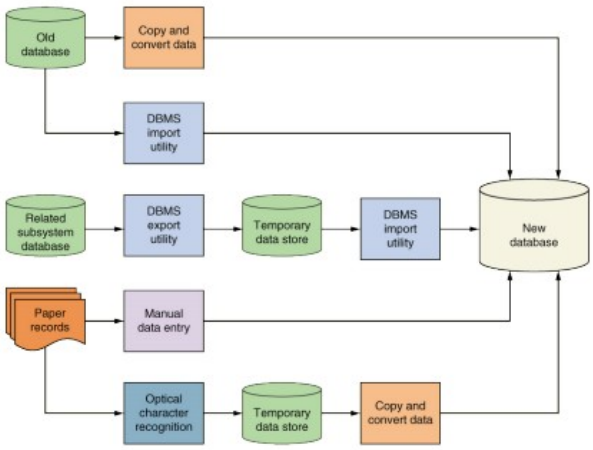
- Log and Results tracking list

Spec ID	Cross refer to use case	Short description	Test condition	Expected outcomes	Name of tester	Date executed	Acceptance criteria	Status	Outstanding issues
10	101	Maintain customer info	Add customer, update customer, delete not allowed	New customer with all fields, updated customer with selected fields	Mary Helper	7/15/2015	All expected outcomes, DB updated successfully	Accepted	None
11	201	Maintain sale info	Create sale, upate sale, finalize sale, pay for sale	New sale in DB, update selected fields, payment creates transaction	Mary Helper	7/15/2015	All expected outcomes, DB updated successfully	Pending	1005, 1006
12	202	Ship items	Display items, update status	Sale update, sale items updated, shipment created				Not started	

Converting and Initializing Data

- Data needed at system start up can be obtained from these sources:
  - Files or database of a system being replaced
  - Manual records
  - Files or databases from other system in the org
  - User feedback during normal system operation
- Reuse existing databases
  - Modify or update existing data
- Reload database
  - Copy and convert the data
  - Export and import data from distinct DBMS
  - Data entry from paper document

- Complex data conversion example



Training Users

- Training for end users must emphasize hands on use for specific business processes or functional, such as order entry
  - Widely varying skill and experience level call for at least of some hands on training
- System operators training can be must less formal when the operator arejt the end users

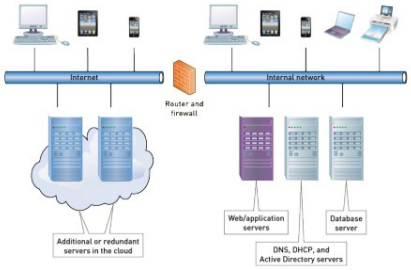
Training Users

End-user activities	System operator activities
Creating records or transactions	Starting or stopping the system
Modifying database contents	Querying system status
Generating reports	Backing up data to archive
Querying database	Recovering data from archive
Importing or exporting data	Installing or upgrading software

- System Documentation
  - Description of system requirement and architecture to help maintenance and update of the system
- User Decimation
  - How to interact will use the system for end user and system operator
- System Document
- User Documentation

Configuring the Production Environment

- Infrastructure and clients on a .net application



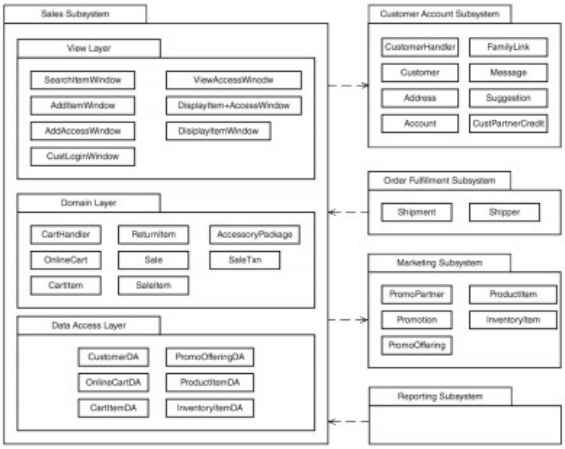
Planning and Managing

Implementation, Testing and Deployment

- Development Order
  - Input, Process, Output - A development order that implements input modules first, process module next and output last
- Top down development - a development order that implements top-level modules first
  - Use stubs for testing
- Bottom up development
- Use-case Driven - Select specific use cases and order the development based on selected use case

IPO Development

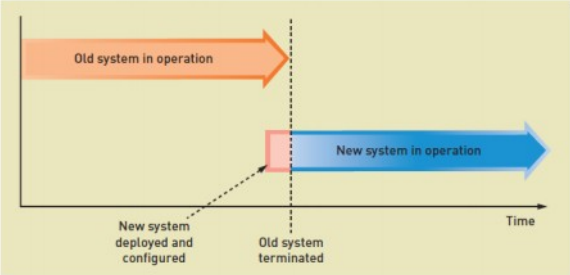
- Package diagram showing dependencies, which constrains development order



- Source code control
  - An automated tool for tracking source code files and controlling changes to those files
    - Programmer checks out a file in read mode without making changes
    - When a programmer needs to make change, check out the file in readwrite mode
    - The SCCS allows only one programmer at a time to check out a file in readwrite mode

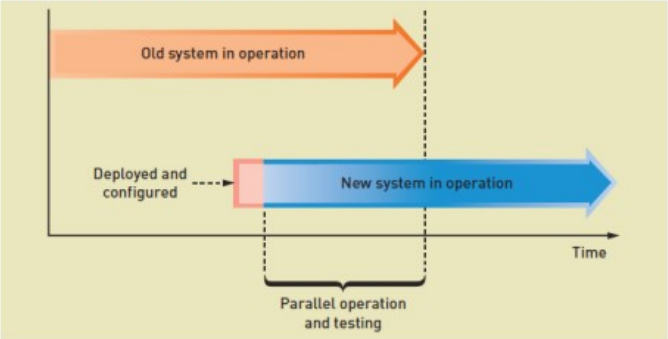
Planning and Managing
Implementation, Testing and Deployment

- Direct deployment – a deployment method that installs a new system, quickly makes it operational, and immediately turns off any overlapping systems
  - Higher risk, lower cost



Planning and Managing
Implementation, Testing and Deployment

- Parallel deployment – a deployment method that operates the old and the new systems for an extended time period
  - Lower risk, higher cost



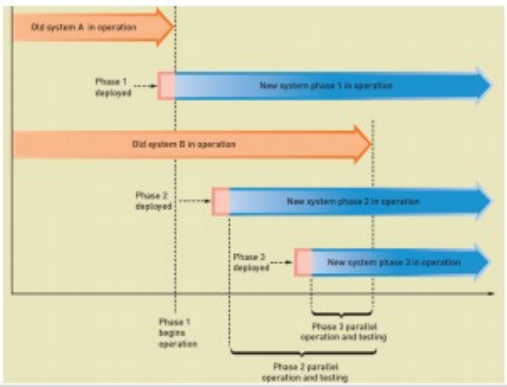
Planning and Managing
Implementation, Testing and Deployment

- Parallel deployment – Problems
  - Incompatible inputs (old and new)
  - Heavier load on equipment, may not have sufficient capacity for both
  - Heavier load on staff, may require overtime
- Partial parallel may be an option
  - Process only a subset of the data
  - Use only part of the system – only some functions

MONASH University

Planning and Managing
Implementation, Testing and Deployment

- Phased deployment
- a deployment method that installs a new system and makes it operational in a series of steps or phases



MONASH

Planning and Managing Implementation, Testing and Deployment

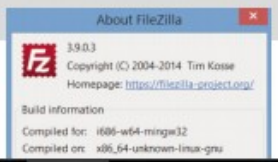
- Change and Version Control – tools and processes handle the complexity associated with testing and supporting a system through multiple versions
  - Alpha version – a test version that is incomplete but ready for some level of rigorous integration or usability testing
  - Beta version – a test version that is stable enough to be tested by end users over an extended period of time
  - Production version, release version, or production release – a system version that is formally distributed to users or made operational for long-term use
  - Maintenance release – a system update that provides bug fixes and small changes to existing features

Version Control

- Need for version control

Version Control

- About box showing version number



Planning and Managing Implementation, Testing and Deployment

- Submitting Error Reports and Change Requests
  - Standard reporting methods
  - Review of requests by a project manager or change control committee
  - For operational systems, extensive planning for design and implementation
- Implementing a Change
  - Identify what parts of the system must be changed
  - Secure resources (such as personnel) to implement the change
  - Schedule design and implementation activities
  - Develop test criteria and a testing plan for the changed system

RMO CSMS System Revisited

- Upgrade or Replace?
  - The current infrastructure is near capacity.
  - RMO expects to save money by having an external vendor host the CSMS
  - Existing CSS programs and Web interfaces are a hodgepodge developed over 15 years
  - Current system software is several versions out of date
  - Infrastructure that supports the current CSS can be repurposed to expand SCM capacity
- RMO decided to Replace

RMO CSMS System Revisited

- Phased Deployment to Minimize Risk
  - Deploy in two versions
- Database Development and Data Conversion
  - New database built and data migrated before deploying version 1, in iterations
- Development Order
  - Start with the higher risk Sales subsystem and customer facing Order fulfillment subsystem
- Documentation and Training
  - Spread throughout later iterations for both versions

RMO CCMS Iteration Plan (part 1)

Iteration	Description
1	Define business models and development/deployment environment. Define essential use cases and rough class diagram. Storyboard sales processing. Finalize deployment environment. Select and acquire network components, system software, hardware, and development tools. Create a CSS database copy with minimal data content as a starting point for CSMS database. Construct a simple prototype for adding a customer order (no database updates) and perform usability testing.
2	Define class, use case, sequence diagrams, and programs, concentrating on the key use cases (*Search for item, Fill shopping cart, Check out shopping cart, Look up customer*, and *Create customer account*). Deploy infrastructure components, including operating systems, Web/application servers, and DBMS by the middle of the iteration. Update database schema based on newly defined or revised classes and associations. Perform usability, unit, and integration testing to validate database design, customer/sales function set, and user interfaces.
3	Loop through iteration 2 use cases again and make all changes determined at the end of the previous iteration. Expand requirements and design to cover additional sales use cases and essential customer account and order-fulfillment use cases. Perform usability, unit, and integration testing.
4	Loop through iteration 3 use cases again and make all changes determined at the end of the previous iteration. Expand requirements and design to cover remaining Marketing subsystem use cases for products and promotions. Develop customer-oriented online help for all functions implemented in previous iterations. Prepare training materials and conduct training for phone and retail stores sales personnel. Finalize the new database and prepare it for data migration. Develop data migration (import) procedures. Test and refine data migration procedures by importing all data from the CSS database.

RMO CCMS Iteration Plan (part 2)

5	Loop through iteration 4 use cases again and make all changes determined at the end of the previous iteration. Continue training for phone and retail stores sales personnel. Conduct usability tests with a large number of actual or simulated customers. Make any needed changes to user interfaces, including online help. Conduct performance and stress testing and make any needed changes. Create a copy of the CSMS deployment environment at the Park City data center for use as a test system for version 2.0 development. Conduct user acceptance testing. Import all CSS database changes since the last import. Place version 1.0 into production.
6	Monitor system performance and user comments. Develop a change list and classify them as "ASAP" or "version 2.0." Implement ASAP changes. Expand requirements and design to cover essential use cases from the Reporting subsystem and those related to social networking. Migrate database updates from CSMS to CSS database twice per day. If no problems are encountered with CSMS, discontinue data migration and old system operation at the end of this iteration.
7	Loop through iteration 6 use cases again and make all changes determined at the end of the previous iteration. Expand requirements and design to cover all remaining use cases. Update database design as needed to support version 2.0 use cases. Program iteration 7 and use cases and conduct unit and integration testing.
8	Develop customer-oriented online help for all functions implemented in iterations 6 and 7. Prepare training materials and conduct training for sales, marketing, and management personnel. Conduct usability tests with a large number of actual or simulated customers. Make any needed changes to user interfaces, including online help. Update the production database with any structural changes in the test database.
9	Continue training for sales, marketing, and management personnel. Conduct performance and stress testing and make any needed changes. Conduct user acceptance testing. Place version 2.0 into production.

Summary

- Implementation and deployment are complex processes because they consist of so many interdependent activities
- Implementation activities include program the software, unit tests, building test cases, and integrate and test components
- Deployment activities include perform system and stress tests, perform acceptance tests, convert existing data, build training materials/conduct training, configure and set up the production environment, and deploy the solution
- Testing is a key activity of implementation and deployment and includes unit tests, integration tests, usability tests, system/performance/stress tests, and acceptance tests

Summary (continued)

- A program development plan is a trade-off among available resources, available time, and the desire to detect and correct errors prior to system deployment
- Configuration and change management activities track changes to models and software through multiple system versions, which enables developers to test and deploy a system in stages
- Versioning also improves post deployment support by enabling developers to track problem support to specific system versions
- Source code control systems enable development teams to coordinate their work

Summary (continued)

- Three options for deployment include direct deployment, parallel deployment and phased deployment
- Direct deployment is riskier but less expensive. Parallel deployment is less risky but more expensive
- For moderate to large projects, a phase deployment approach makes sense to get key parts of the system operational earlier

Week 11 –

Week 12 – Emerging Trends