Lecture 1 - Introduction to System Development

Subject: Modern System Analysis and Design

Objectives:

At the end of the lesson, students should be able to:

1. Describe the purpose of systems analysis and design when developing information systems
2. Explain the purpose of the system development life cycle (SDLC) and identify its six core processes
3. Explain how information system methodologies provide guidelines for completing the six core processes of the SDLC
4. Describe the characteristics of Agile methodologies and iterative system development

Software Development and Systems Analysis and Design

You have grown up in a world of ubiquitous computing, where computers are everywhere and are increasingly characterized by mobility, communication, and connectivity. You use smartphones, laptops, notepads, and wearable devices throughout the day. Some of you have already developed your own application software or have friends who have written applications for these devices. Some of you have taken programming classes; others have taught yourself how to write computer application programs. In one way or another, you are certainly interested in building computer applications and information systems.

An ***information system*** is a set of interrelated components that collect, process, store, and provide as output the information needed to complete business tasks. The information system always includes people who operate the system and carry out some of the work.

More recently, another term has been used to refer to an information system—a ***computer application***. A computer application is a computer software program that executes on a computing device to carry out a specific function or set of related functions. Sometimes, computer application is shortened to ***app*** (such as an iPhone app or an Android app). Many people use the terms information system and computer application interchangeably but remember that an information system includes people and their manual procedures, and an application usually refers just to the software. Fig. 1 shows variety of devices connected to a university information system.

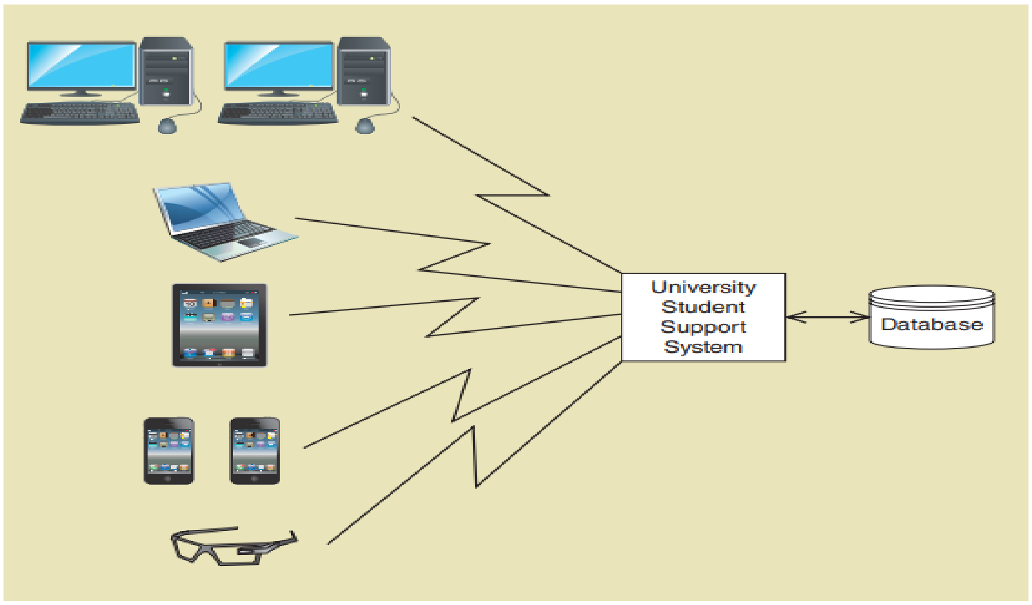


Fig. 1. Different devices connected to a university information system

Systems analysis consists of those activities that enable a person to understand and specify what the new system should accomplish. The operative words here are understanding and specifying. Systems analysis is more than a brief statement of the problem. Systems analysis describes in detail what a system must do to satisfy the need or solve the problem.

Systems design consists of those activities that enable a person to describe in detail how the information system will be implemented to provide the needed solution. In other words, systems design describes how the system will work. It specifies in detail all the components of the solution system and how they work together. See Fig. 2 to help distinguish between analysis and design.

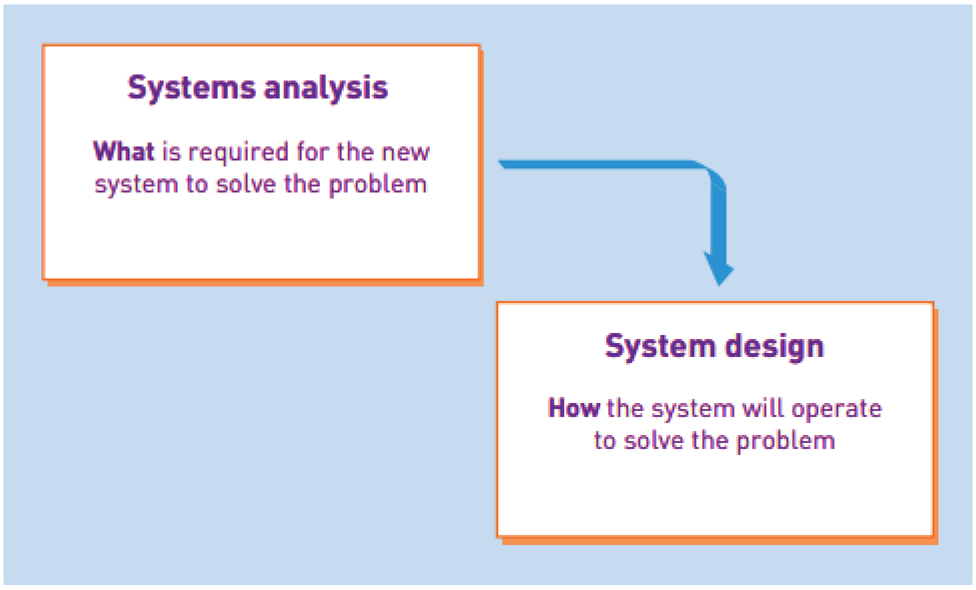

Systems analysis
What is required for the new system to solve the problem
System design
How the system will operate to solve the problem

Fig. 2 Systems analysis versus Systems design

Systems analysis and design plays an integral role in the development of information systems. To illustrate, consider an analogous situation: the art and science of creating a new building. In this scenario, there is the owner of the land who has the vision, the builder who will construct the building, and the architect who serves as the bridge between the owner and the builder. The architect helps the owner develop the vision but must also communicate the building's specifications to the builder. In doing so, the architect uses various tools to first capture the vision from the owner and to then provide the builder with instructions—including line drawings, blueprints, to-scale models, detail specifications, and even on-site inspection reports. Just as a builder doesn't start construction without plans, programmers don't just sit down and start writing program code. They need someone (maybe themselves) to function like an architect—planning, capturing the vision, understanding details, specifying needs—before designing and writing the code that satisfies the vision. Usually, we call this person a ***systems analyst***. In situations where you are the programmer as well as the analyst (often called a programmer-analyst), it might be possible to keep track of the details without many formal notes.

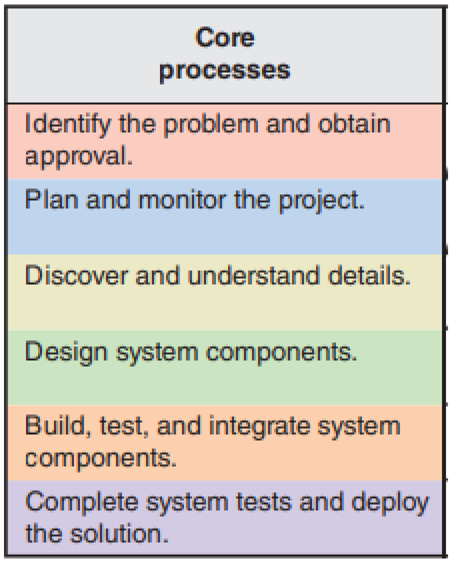
In a nutshell, systems analysis and design provides the tools and techniques you need as an information system developer to complete the development process:

1. Understand the need (business need).
2. Capture the vision.
3. Define a solution.
4. Communicate the vision and the solution.
5. Build the solution or direct others in building the solution.
6. Confirm that the solution meets the need.
7. Launch the solution application.

The System Development Life Cycle SDLC)

A ***project*** is a planned undertaking that has a beginning and an end and produces some result. This means that the activities required to develop a new system are identified, planned, organized, and monitored. Some projects are very formal, whereas others are informal, usually depending on the project size.

To manage a project with analysis, design, and other development activities, you need a project management framework to guide and coordinate the work of the project team. The ***system development life cycle (SDLC)*** is a framework that identifies all the activities required to research, build, deploy, and often maintain an information system. Normally, the SDLC includes all activities needed for the planning, systems analysis, systems design, programming, testing, and user training stages of information systems development, as well as other project management activities that are required to successfully deploy the new information system. There are many approaches to the SDLC, including variations specific to certain types of projects. However, every SDLC includes some core processes that are always required, though many different names are used. Here are the six core processes required in the development of any information system (see Fig.3):



Core processes
Identify the problem and obtain approval.
Plan and monitor the project.
Discover and understand details.
Design system components.
Build, test, and integrate system components.
Complete system tests and deploy the solution.

Fig. 3. SDLC Core Processes

Most information systems you will develop are conceived and built to solve complex organizational problems, which are usually very complex, thus making it difficult to plan and manage a system development project. Fortunately, there are many ways to implement the six core processes of the SDLC to handle each project's complexity***. Information systems development methodology*** is a set of comprehensive guidelines for carrying out all the activities of each core process of the SDLC. An overall system development process is a more recent term for methodology. Each develop methodology prescribes a way of carrying out the development project, and every organization develops its own system development methodology over time to suit its needs.

Agile Development

The basic philosophy of Agile development is that neither team members nor the users completely understand the problems and complexities of a new system, so the project plan and the execution of the project must be responsive to unanticipated issues. The plan must be agile and flexible. It must have procedures in place to allow for, anticipate, and even embrace changes and new requirements that come up during the development process. The six core processes are still involved in Agile development, but they are carried out iteratively.

Iterative Development

Iterative development is an approach to system development in which the system is "grown" in an almost organic fashion. Core components are developed first, and then additional components are added. It is called iterative because the six core development processes are repeated for each component. In other words, there is one big project, which consists of a series of mini-projects, and the information system is grown piece by piece during these mini-projects. Iterative development makes Agile development possible, although Agile development includes additional techniques that help with project flexibility, too.

Fig. 4 illustrates how an iterative project might be managed. Across the figure, you see six iterations as columns. Each iteration involves all six core processes, shown as rows in the table. At the end of each iteration, a working part of the system is completed and evaluated. An iteration lasts a fixed period, usually two to four weeks. The rounded mounds inside the graph represent the relative amount of effort for that core process during that iteration. For example, in Figure 1–5, Iteration 1 appears to primarily focus on identifying the problem and planning the project. Lesser amounts of discovery, design, and build/test may also be done. For this iteration, nothing is done about deploying the system. In Iteration 2, there is less effort for identifying the problem and planning the project and more effort for discovery, design, and build/test. By Iteration 3, build/test gets the most effort, but all six core processes are still involved, including the beginnings of completing and deploying the system.

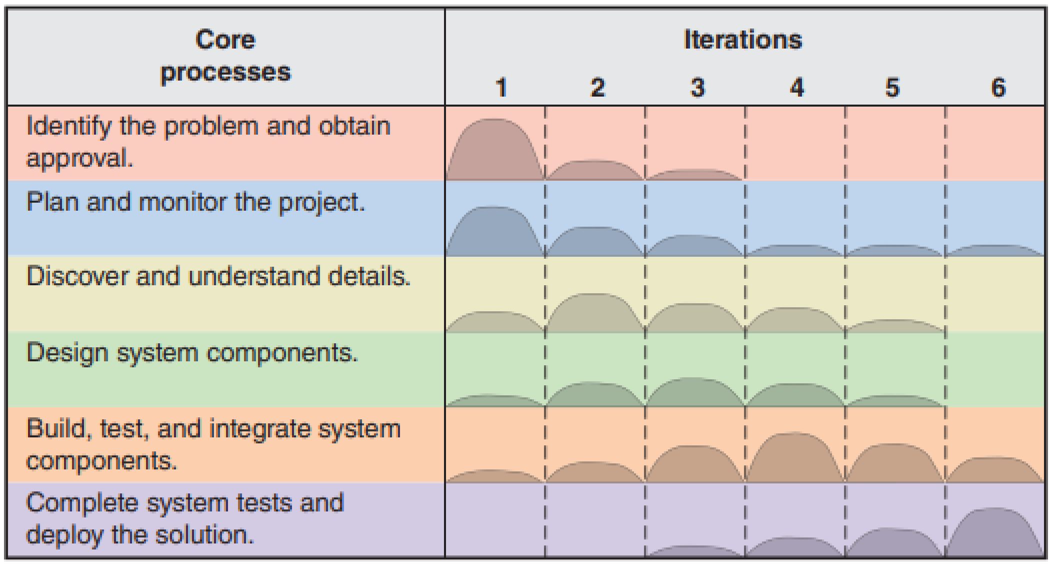


Fig. 4. Core processes and its iterations

A key element of iterative development is dividing system components into pieces that can be completed in two to four weeks. During one iteration, all the core development processes are involved, including programming and systemwide testing, so the result is a working part of the system, even though it may only have a portion of the functionality that is ultimately required. Developers choose components for each iteration based on priority, either the components most needed or riskiest to implement.

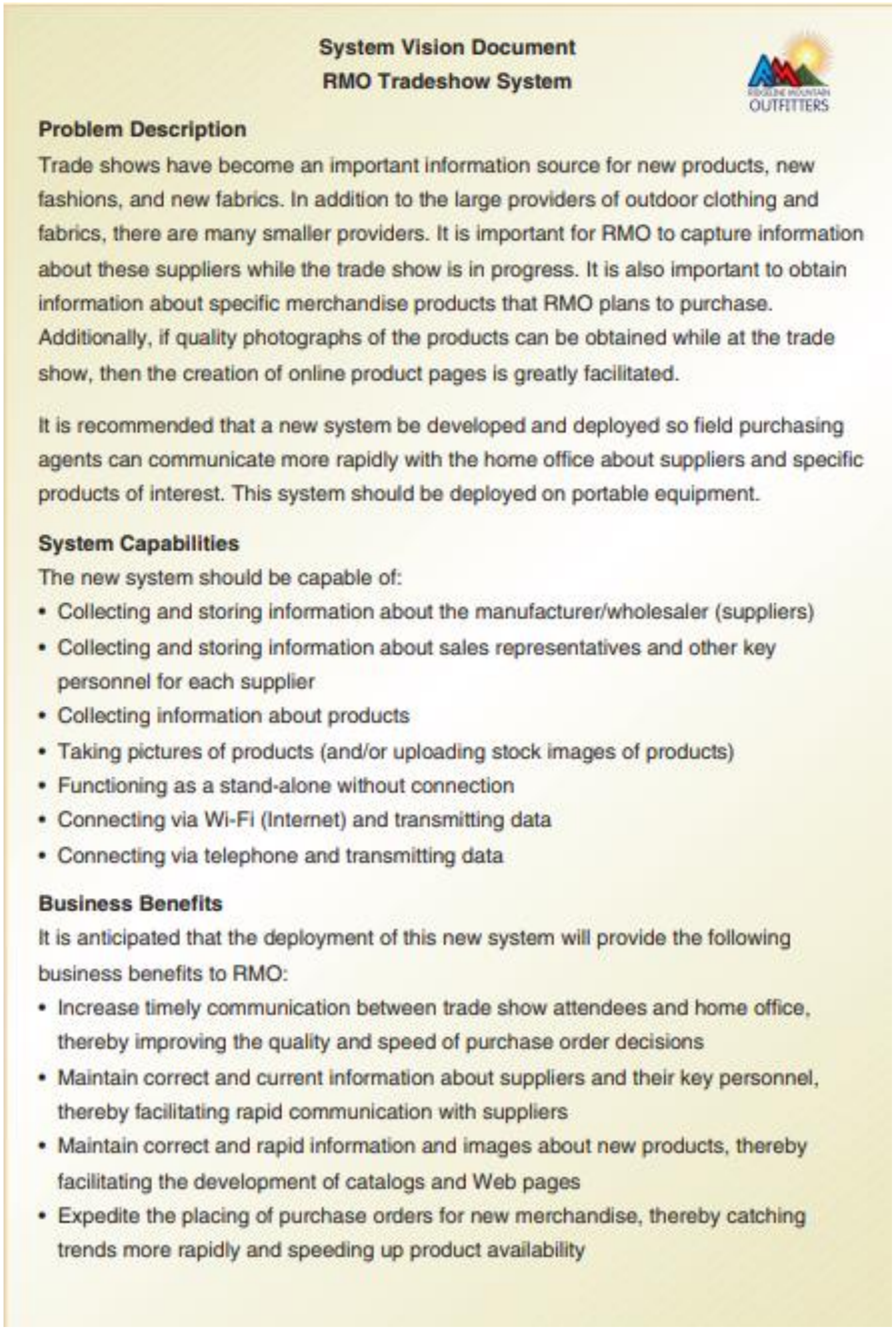
Case Study: Ridgeline Mountain Outfitters (RMO)

Day 1 Activities:

Core 1:

- **Identify the problem and document the objective of the solution system.**

Problem Description of RMO:



System Vision Document
RMO Tradeshow System

Problem Description

Trade shows have become an important information source for new products, new fashions, and new fabrics. In addition to the large providers of outdoor clothing and fabrics, there are many smaller providers. It is important for RMO to capture information about these suppliers while the trade show is in progress. It is also important to obtain information about specific merchandise products that RMO plans to purchase. Additionally, if quality photographs of the products can be obtained while at the trade show, then the creation of online product pages is greatly facilitated.

It is recommended that a new system be developed and deployed so field purchasing agents can communicate more rapidly with the home office about suppliers and specific products of interest. This system should be deployed on portable equipment.

System Capabilities

The new system should be capable of:

- Collecting and storing information about the manufacturer/wholesaler (suppliers)
- Collecting and storing information about sales representatives and other key personnel for each supplier
- Collecting information about products
- Taking pictures of products (and/or uploading stock images of products)
- Functioning as a stand-alone without connection
- Connecting via Wi-Fi (Internet) and transmitting data
- Connecting via telephone and transmitting data

Business Benefits

It is anticipated that the deployment of this new system will provide the following business benefits to RMO:

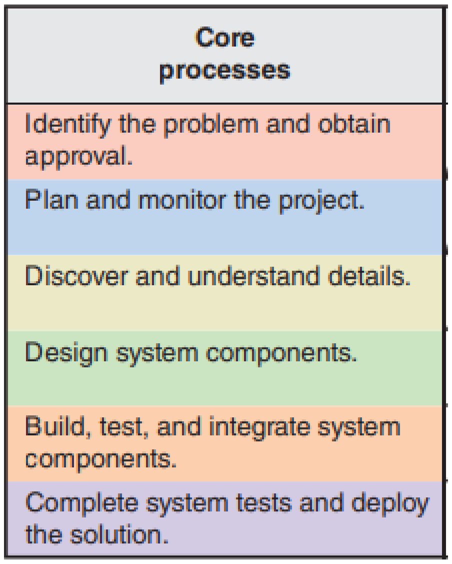
- Increase timely communication between trade show attendees and home office, thereby improving the quality and speed of purchase order decisions
- Maintain correct and current information about suppliers and their key personnel, thereby facilitating rapid communication with suppliers
- Maintain correct and rapid information and images about new products, thereby facilitating the development of catalogs and Web pages
- Expedite the placing of purchase orders for new merchandise, thereby catching trends more rapidly and speeding up product availability

Fig. 5. System Vision Document of RMO

Question:

What are the major problems of the company and how did they solve them?

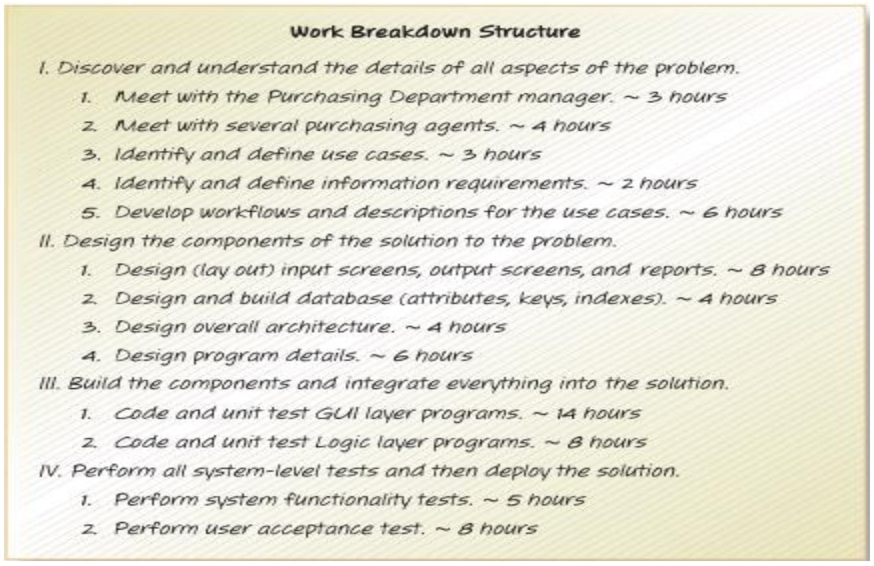
Core 2: Plan the project



Core processes
Identify the problem and obtain approval.
Plan and monitor the project.
Discover and understand details.
Design system components.
Build, test, and integrate system components.
Complete system tests and deploy the solution.

- Determine the major components ((functional areas) that are needed
    - Supplier information system
    - Product information system
- Define the iterations and assign each function to an iteration
    - Decide to do supplier subsystem first
    - Plan one iteration as it is small and straight forward
- Determine team members and responsibilities

In identifying all individual tasks that needed to be done, you can develop the Work Breakdown Structure (WBS). WBS describes the works and covers core processes 3, 4, 5, and 6. Fig. 6 shows RMO's WBS.



Work Breakdown Structure

I. Discover and understand the details of all aspects of the problem.
1. Meet with the Purchasing Department manager. ~ 3 hours
2. Meet with several purchasing agents. ~ 4 hours
3. Identify and define use cases. ~ 3 hours
4. Identify and define information requirements. ~ 2 hours
5. Develop workflows and descriptions for the use cases. ~ 6 hours

II. Design the components of the solution to the problem.
1. Design (lay out) input screens, output screens, and reports. ~ 8 hours
2. Design and build database (attributes, keys, indexes). ~ 4 hours
3. Design overall architecture. ~ 4 hours
4. Design program details. ~ 6 hours

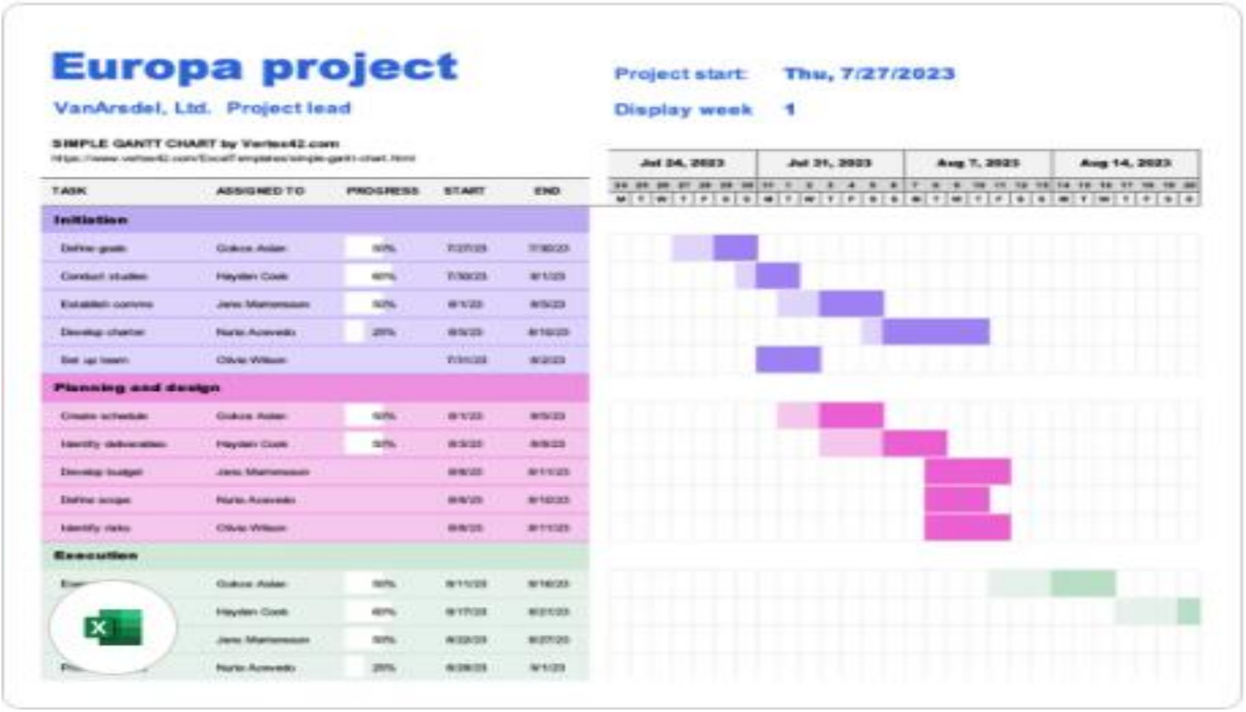
III. Build the components and integrate everything into the solution.
1. Code and unit test GUI layer programs. ~ 14 hours
2. Code and unit test Logic layer programs. ~ 8 hours

IV. Perform all system-level tests and then deploy the solution.
1. Perform system functionality tests. ~ 5 hours
2. Perform user acceptance test. ~ 8 hours

Fig. 6. RMO's WBS

You may also use a Gantt chart to organize your Work Breakdown Structure.

Example of a Gantt Chart:



Europa project

VanArsdel, Ltd. Project lead

Project start: Thu, 7/27/2023

Display week: 1

SIMPLE GANTT CHART by Vertex42.com
https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html

TASK	ASSIGNED TO	PROGRESS	START	END
Initiation				
Define goals	Gokce Aslan	50%	7/27/23	7/30/23
Conduct studies	Hayden Cook	60%	7/30/23	8/1/23
Establish comms	Jens Martensson	50%	8/1/23	8/5/23
Develop charter	Nuria Acevedo	25%	8/5/23	8/10/23
Set up team	Olivia Wilson		7/31/23	8/2/23
Planning and design				
Create schedule	Gokce Aslan	50%	8/1/23	8/5/23
Identify deliverables	Hayden Cook	50%	8/3/23	8/8/23
Develop budget	Jens Martensson		8/8/23	8/11/23
Define scope	Nuria Acevedo		8/8/23	8/10/23
Identify risks	Olivia Wilson		8/8/23	8/11/23
Execution				
[illegible]	Gokce Aslan	50%	8/11/23	8/16/23
[illegible]	Hayden Cook	40%	8/17/23	8/21/23
[illegible]	Jens Martensson	50%	8/22/23	8/27/23
[illegible]	Nuria Acevedo	25%	8/28/23	9/1/23

Day 2 Activities:

Core 3 Process – Discover and understand details

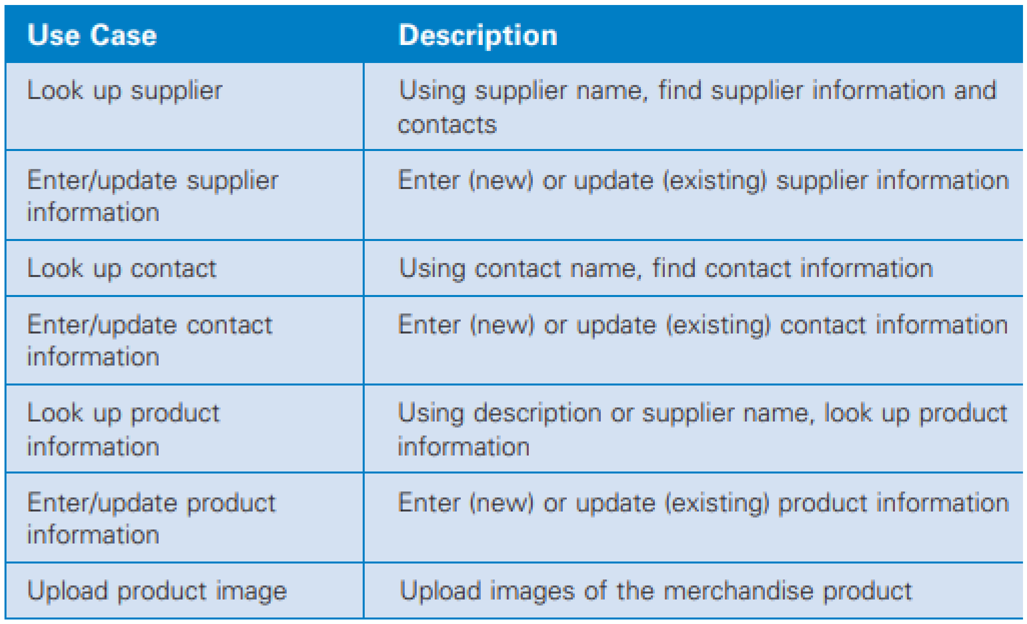
Core processes
Identify the problem and obtain approval.
Plan and monitor the project.
Discover and understand details.
Design system components.
Build, test, and integrate system components.
Complete system tests and deploy the solution.

- Do preliminary fact-finding to understand requirements
- Develop a preliminary list of use cases and a use case diagram
- Develop a preliminary list of classes and a class diagram

Use Case – a case or situation where the system is used.

1. **Identify Use Cases**

RMO System's Use Cases

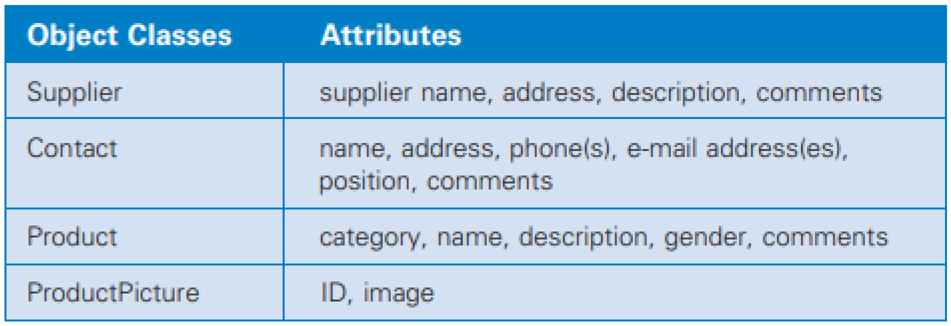


Use Case	Description
Look up supplier	Using supplier name, find supplier information and contacts
Enter/update supplier information	Enter (new) or update (existing) supplier information
Look up contact	Using contact name, find contact information
Enter/update contact information	Enter (new) or update (existing) contact information
Look up product information	Using description or supplier name, look up product information
Enter/update product information	Enter (new) or update (existing) product information
Upload product image	Upload images of the merchandise product

2. **Identify Object Classes**

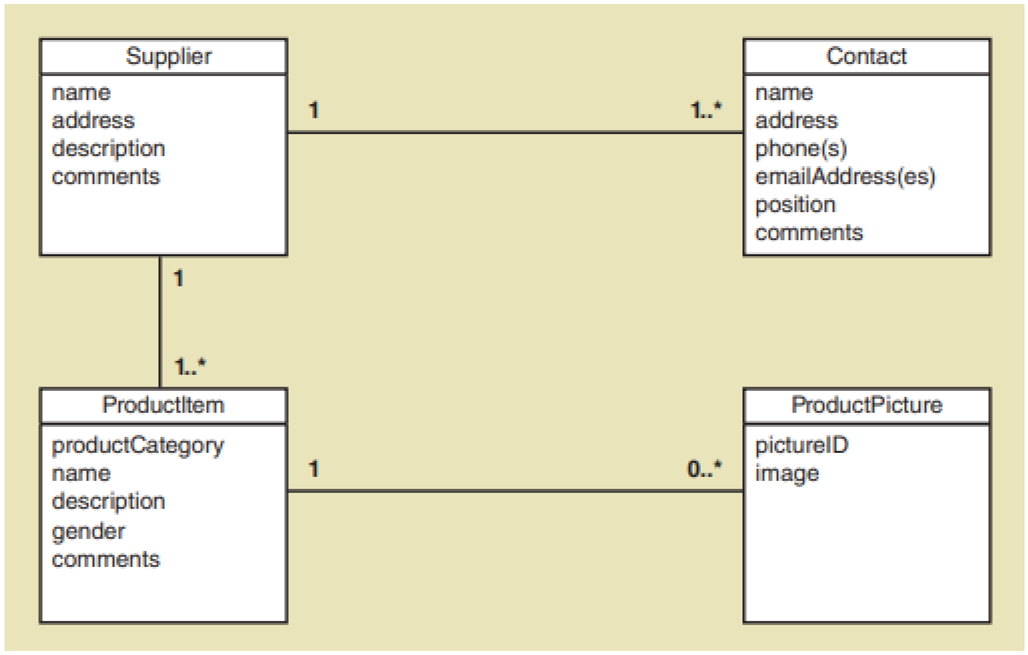
Domain classes identify those things in the real world that the system needs to know about and keep track of. To find domain classes, we look for all objects, or things, that the system uses or captures. Domain classes are identified during the discussions with purchasing agents by looking for the nouns that describe categories of things. For example, the agents will often talk about suppliers, merchandise products, or inventory items. These nouns become the domain classes, and each domain class has attributes (like contact information, product, or business location) that detail the information you need to store about the domain class.

RMO System's Object Classes



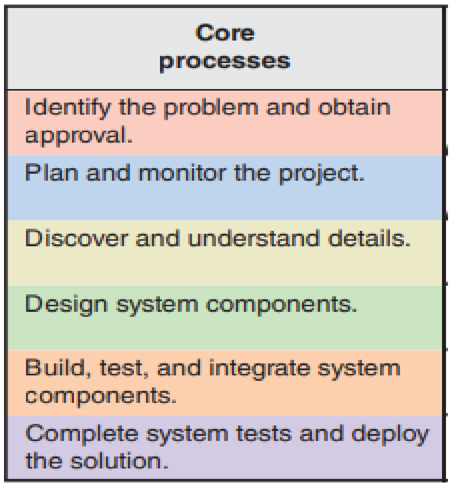
Object Classes	Attributes
Supplier	supplier name, address, description, comments
Contact	name, address, phone(s), e-mail address(es), position, comments
Product	category, name, description, gender, comments
ProductPicture	ID, image

Class Diagram of the Trade Show System:



Each box is a class and can be thought of as a particular set of objects that are important to the system. Important attributes of each class are also included in each box. These represent the detailed information about each object that will be maintained by the system. Note that some classes have lines connecting them. These represent associations between the classes that need to be remembered by the system. For example, a contact is a person who works for a particular supplier. A specific example might be that Bill Williams is the contact person for the South Pacific Sportswear Company. Thus, the system needs to associate Bill Williams and the South Pacific Sportswear Company. The association line documents that requirement. Domain class diagrams are a powerful and frequently used way to understand and document the information requirements of a system.

Day 3 Activities: Core 3 and 4 Processes



Core processes
Identify the problem and obtain approval.
Plan and monitor the project.
Discover and understand details.
Design system components.
Build, test, and integrate system components.
Complete system tests and deploy the solution.

Core Process 3: Discover and Understand Details

- Do in-depth fact-finding to understand requirements
- Understand and document the detailed workflow of each use case

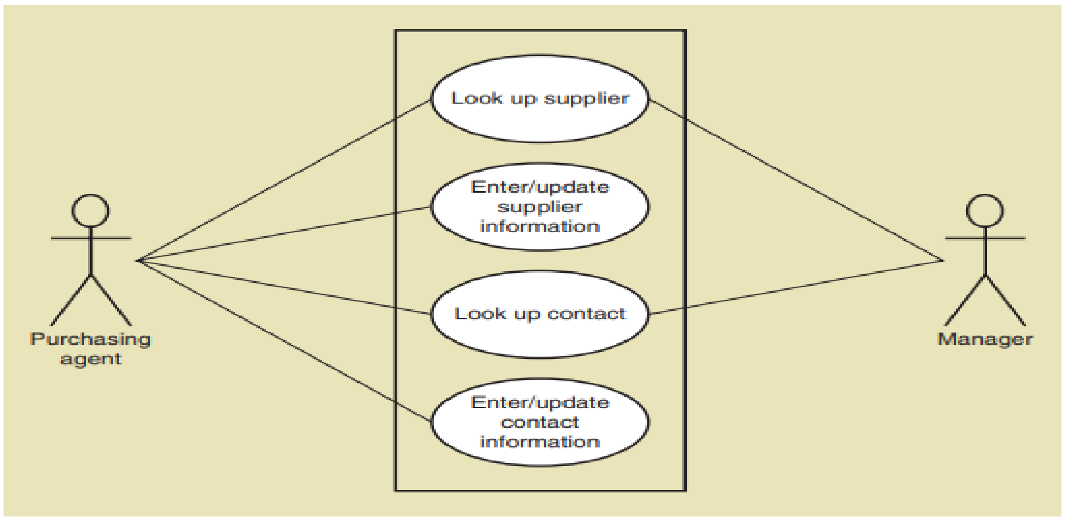
Core Process 4: Design System Components

- Define the user experience with screens and report sketches

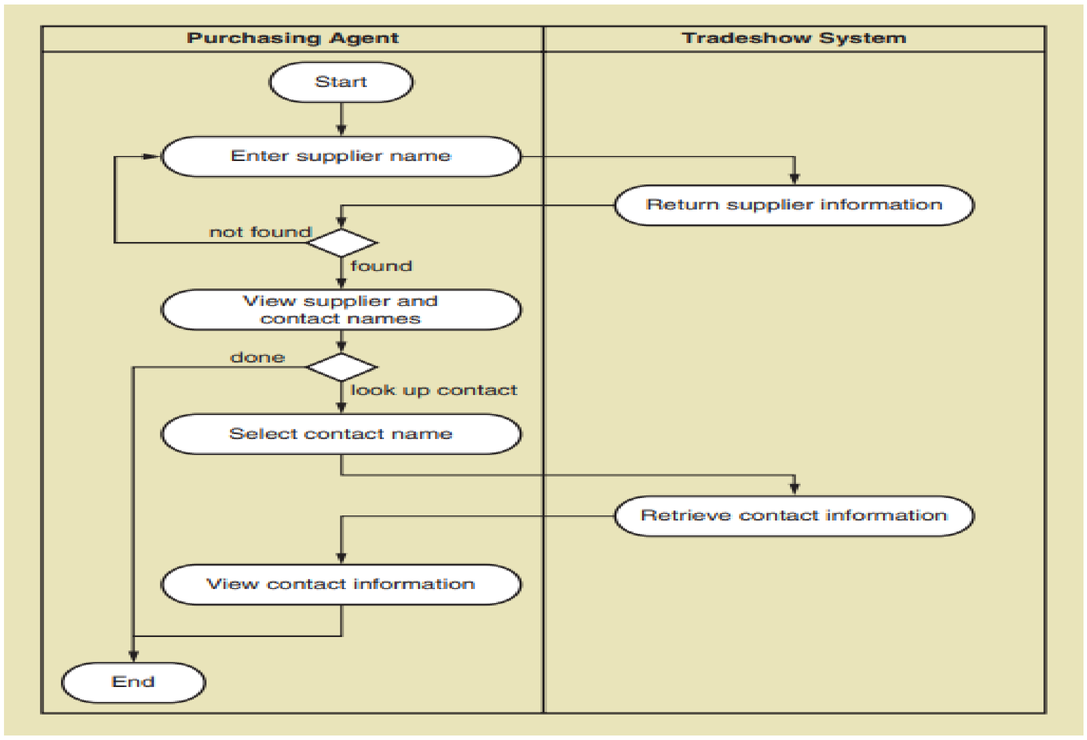
Supplier Information Subsystem
Use cases:

- Lookup supplier
- Enter/update supplier information
- Lookup contact information
- Enter/update contact information

Use case diagram for the Supplier Information Subsystem:

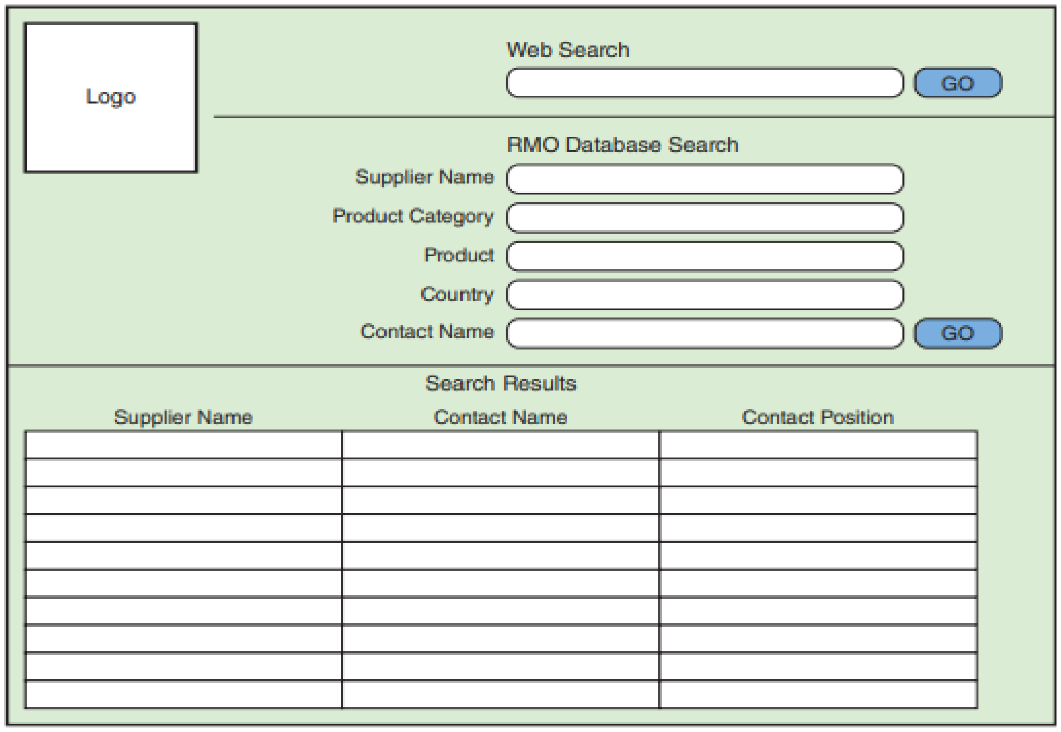

Look up supplier
Enter/update supplier information
Look up contact
Enter/update contact information
Purchasing agent
Manager

Activity diagram for the Supplier Information Subsystem:



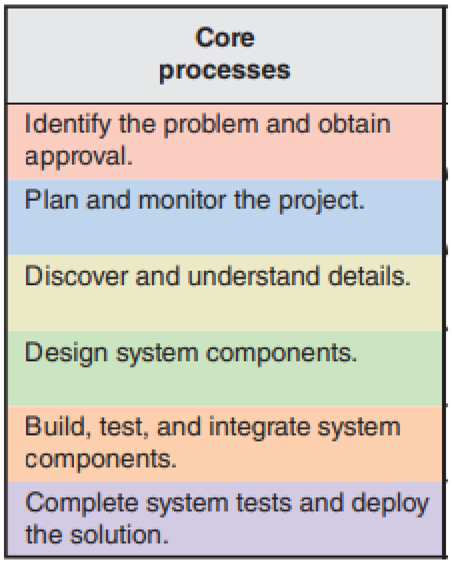
The arrows that cross the line between users (center) represent the interactions between the user and the system. These are critically important because they designate situations where the developers must provide a screen or Web page that either captures or displays information. These situations become part of the user interface. In the activity diagram above, the top arrow indicates that the supplier's name is entered into the system first. Thus, we infer that the use case must have an online lookup page with a field available to enter the supplier's name. The next arrow indicates the application requires a page that displays all the details for an individual supplier, including a list of existing contacts. The user may also want to see more details about a specific contact person for this supplier, so the application must provide detailed information request fields for a particular contact. Because the user will need to select one of the displayed results, we must design the page so each entry on the list is either a hot link or can be selected.

Initial page layout for the Look up use case:



Day 4 Activities:

The primary focus of Day 4 is to design the various components of the solution system, corresponding to Core Process 4: Design System Components.



Core processes
Identify the problem and obtain approval.
Plan and monitor the project.
Discover and understand details.
Design system components.
Build, test, and integrate system components.
Complete system tests and deploy the solution.

Core Process 4: Design System Components

- Design the database (schema)
- Design the system's high-level structure
    - Browser, windows, or smart phone
    - Architectural configuration (components)
    - Design class diagram
    - Subsystem architectural design

Database design is a straightforward activity that uses the domain class diagram and develops the detailed database schema that can be directly implemented by a database management system. Such elements as table design, key and index identification, attribute types, and other efficiency decisions are made during this activity.

Database Schema:

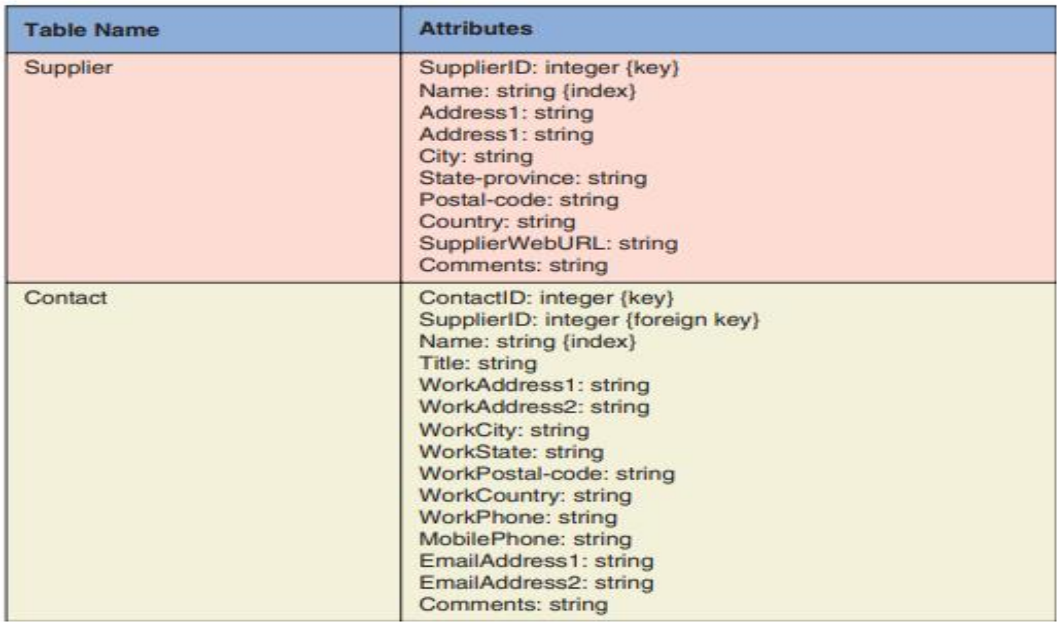
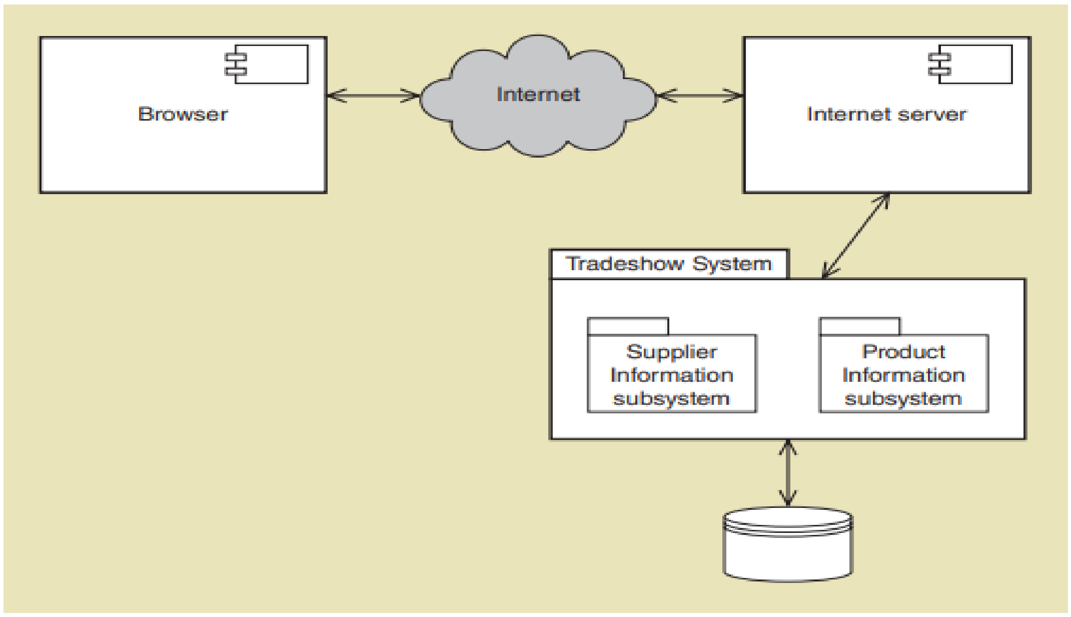


Table Name	Attributes
Supplier	SupplierID: integer {key} Name: string {index} Address1: string Address1: string City: string State-province: string Postal-code: string Country: string SupplierWebURL: string Comments: string
Contact	ContactID: integer {key} SupplierID: integer {foreign key} Name: string {index} Title: string WorkAddress1: string WorkAddress2: string WorkCity: string WorkState: string WorkPostal-code: string WorkCountry: string WorkPhone: string MobilePhone: string EmailAddress1: string EmailAddress2: string Comments: string

Architectural Configuration Diagram:

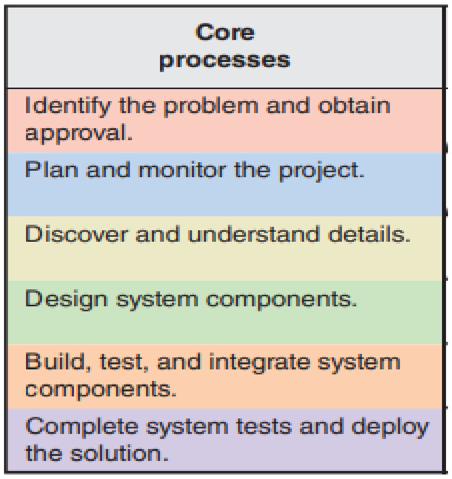


Include also Class Diagram Design and other subsystem's architectural design.

Managing the Project

Design is a complex activity with multiple perspectives—from high-level structural design to low-level detailed program design. In our project, we have separated the tasks for designing the overall system structure from detailed design of the programs themselves. However, these activities are often done concurrently. The basic high-level software architectural structure is defined first, but midlevel and low-level design are often done concurrently with programming. Detailed design and programming are quite time-consuming activities. A project manager must decide whether to extend the project or bring on additional programmers to help write the code. In our project, we have elected to insert a half-day of free time to bring in two additional programmers and train them.

Day 5 Activities:



Core processes
Identify the problem and obtain approval.
Plan and monitor the project.
Discover and understand details.
Design system components.
Build, test, and integrate system components.
Complete system tests and deploy the solution.

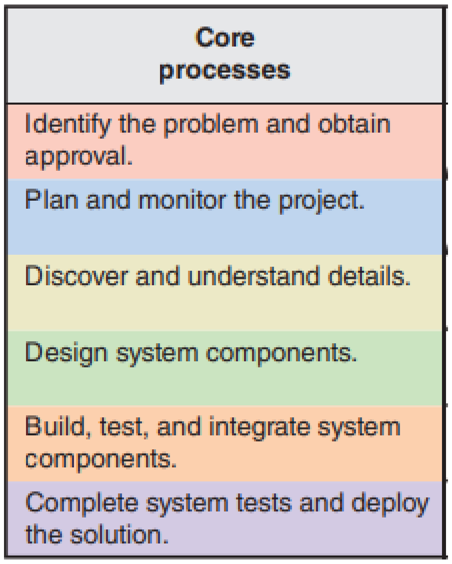
Core Process 4: Design System Components

- Continue with design details
- Proceed use case by use case

Core Process 5: Build, Test, and Integrate System

- Continue programming (build)
- Build use case by use case
- Perform unit and integration tests

Day 6 Activities:

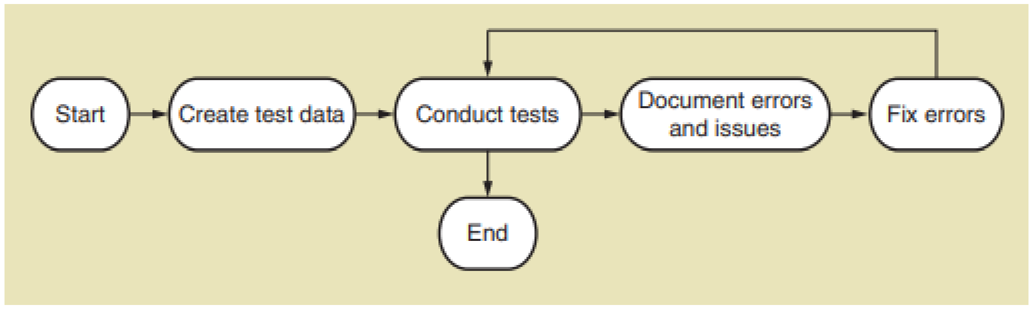


Core processes
Identify the problem and obtain approval.
Plan and monitor the project.
Discover and understand details.
Design system components.
Build, test, and integrate system components.
Complete system tests and deploy the solution.

Core Process 6: Complete System Testing and deploy the system

- Perform system functional testing
- Perform user acceptance testing
- possibly deploy part of system

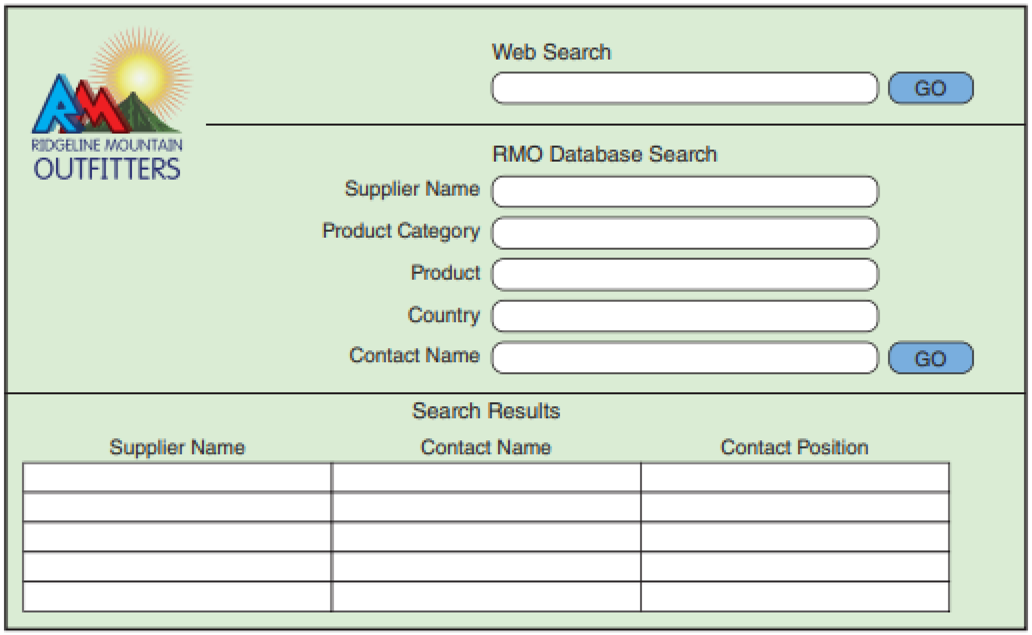
Flowchart for testing tasks:



First Iteration Recap:

- This is a 6-day iteration of small project
  - Most iterations are longer (2 to 4 weeks)
  - This project might be 2 iterations
  - Most projects have many more iterations
- End users need to be involved, particularly on day 1, 2, 3, and 6.
- Days 4 and 5 involved design and programming concurrently.

UI for page in ***Look up supplier*** use case for tablet:



Supplier Name	Contact Name	Contact Position

Text: Systems Analysis and Design in A Changing World
By: Satzinger, Jackson, and Burd