Requirements Analysis

CIS 375

Bruce R. Maxim
UM-Dearborn

Requirements Analysis

- Software engineering task bridging the gap between system requirements engineering and software design.
- Provides software designer with a model of:
  - system information
  - function
  - behavior
- Model can be translated to data, architectural, and component-level designs.
- Expect to do a little bit of design during analysis and a little bit of analysis during design.

Analysis Objectives

- Identify customer's needs.
- Evaluate system for feasibility.
- Perform economic and technical analysis.
- Allocate functions to system elements.
- Establish schedule and constraints.
- Create system definitions.

Software Requirements Analysis Phases

- Problem recognition
- Evaluation and synthesis
  - focus is on what not how
- Modeling
- Specification
- Review

Management Questions

- How much effort put towards analysis?
- Who does the analysis?
- Why is it so difficult?
- Bottom line - who pays for it?

Feasibility Study

- Economic feasibility
  - cost/benefit analysis
- Technical feasibility
  - hardware/software/people, etc.
- Legal feasibility
- Alternatives
  - there is always more than one way to do it

System Specification

- Introduction.
- Functional data description.
- Subsystem description.
- System modeling and simulation results.
- Products.
- Appendices.

Requirements

- Requirement
  - features of system or system function used to fulfill system purpose.
- Focus on customer's needs and problem, not on solutions:
  - Requirements definition document
    (written for customer).
  - Requirements specification document
    (written for programmer; technical staff).

Types of Requirements - 1

- Functional requirements:
  - input/output
  - processing.
  - error handling.
- Non-functional requirements:
  - Physical environment (equipment locations, multiple sites, etc.).
  - Interfaces (data medium etc.).
  - User & human factors (who are the users, their skill level etc.).

Types of Requirements - 2

- Non-functional requirements (continued):
  - Performance (how well is system functioning).
  - Documentation.
  - Data (qualitative stuff).
  - Resources (finding, physical space).
  - Security (backup, firewall).
  - Quality assurance (max. down time, MTBF, etc.).

Requirement Validation

- Correct?
- Consistent?
- Complete?
  - Externally - all desired properties are present.
  - Internally - no undefined references.
- Each requirement describes something actually needed by the customer.
- Requirements are verifiable (testable)?
- Requirements are traceable.

Requirements Definition Document

- General purpose of document.
- System background and objectives.
- Description of approach.
- Detailed characteristics of proposed system (data & functionality).
- Description of operating environment.

Software Requirements Elicitation

- Customer meetings are the most commonly used technique.
- Use context free questions to find out
  - customer's goals and benefits
  - identify stakeholders
  - gain understanding of problem
  - determine customer reactions to proposed solutions
  - assess meeting effectiveness
- Interview cross section of users when many users are anticipated.

F.A.S.T. - 1

- Facilitated application specification technique
- Meeting between customers and developers at a neutral site (no home advantage).
- Goals
  - identify the problem
  - propose elements of solution
  - negotiate different approaches
  - specify preliminary set of requirements

F.A.S.T. - 2

- Rules for participation and preparation established ahead of time.
- Agenda suggested
  - brainstorming encouraged
- Facilitator appointed.
- Definition mechanism
  - sheets, flipcharts, wallboards, stickers, etc.

Q.F.D. - 1

- Quality Function Deployment
- Customer's needs imply technical requirements:
  - Normal requirements
    (minimal functional & performance).
  - Expected requirements
    (important implicit requirements, i.e. ease of use).
  - Exciting requirements
    (may become normal requirements in the future, highly prized & valued).

Q.F.D. - 2

- Function Deployment:
  - Determines value of required function.
- Information Deployment:
  - Focuses on data objects and events produced or consumed by the system.
- Task Deployment:
  - product behavior and implied operating environment.

Q.F.D. - 3

- *Value Analysis* makes use of:
  - Customer interviews.
  - Observations.
  - Surveys.
  - Historical data.
- to create
  - Customer Voice Table
  - extract expected requirements
  - derive exciting req.

Use Cases

- Scenarios that describe how the product will be used in specific situations.
- Written narratives that describe the role of an actor (user or device) as it interacts with the system.
- Use-cases designed from the actor's point of view.
- Not all actors can be identified during the first iteration of requirements elicitation, but it is important to identify the primary actors before developing the use-cases.

User Profile - Example

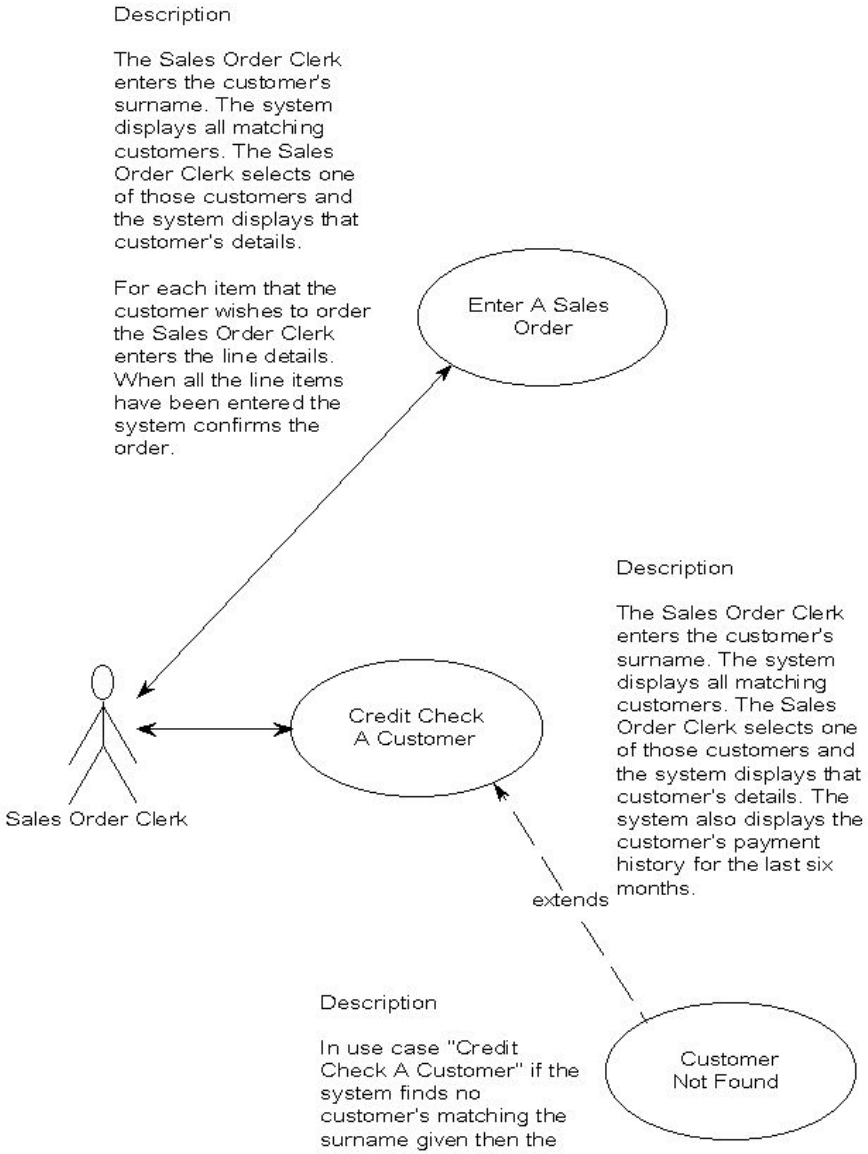
- Full Control (Administrator)
- Read/Write/Modify All (Manager)
- Read/Write/Modify Own (Inspector)
- Read Only (General Public)

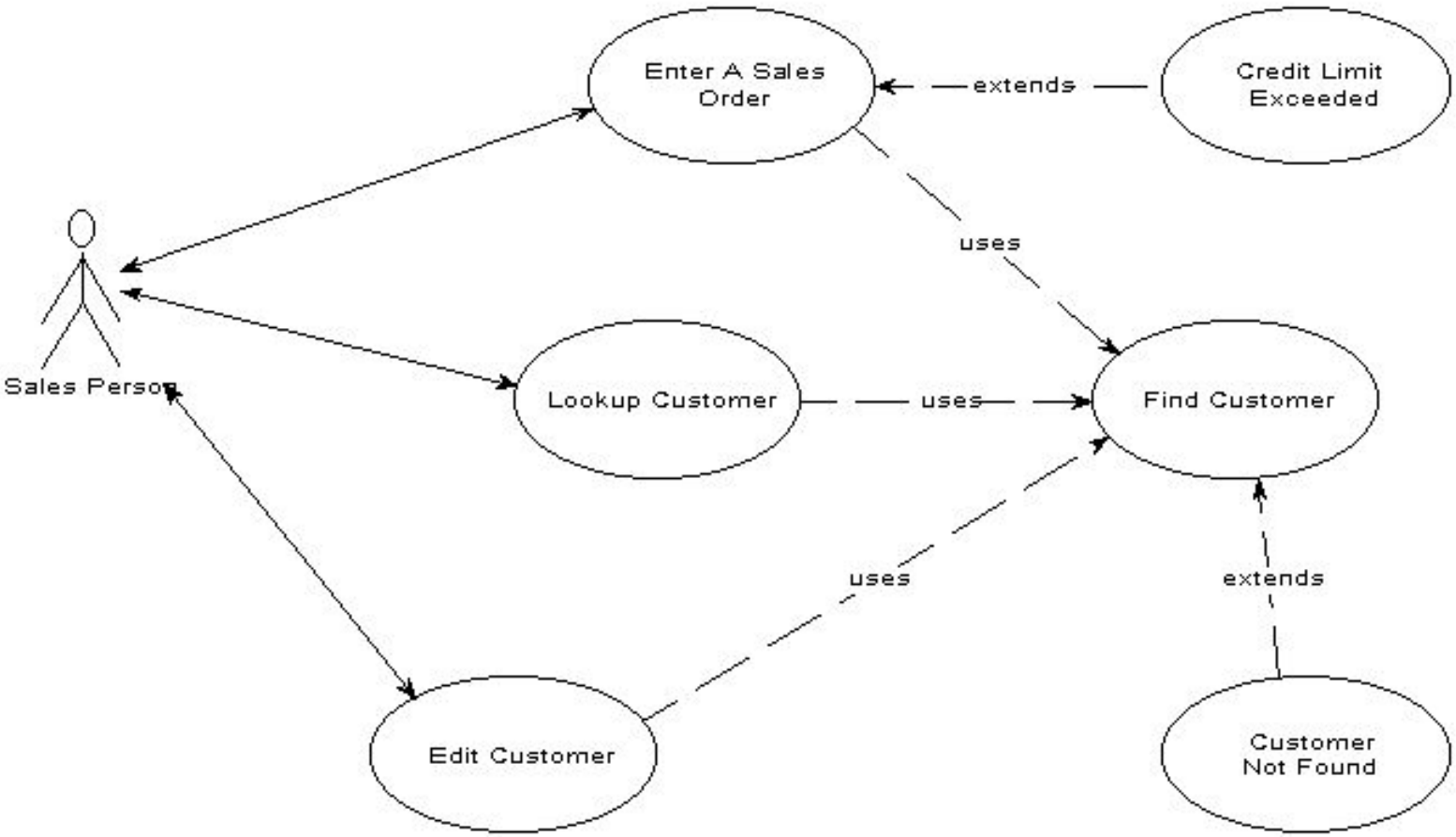
Use Case Example - 1

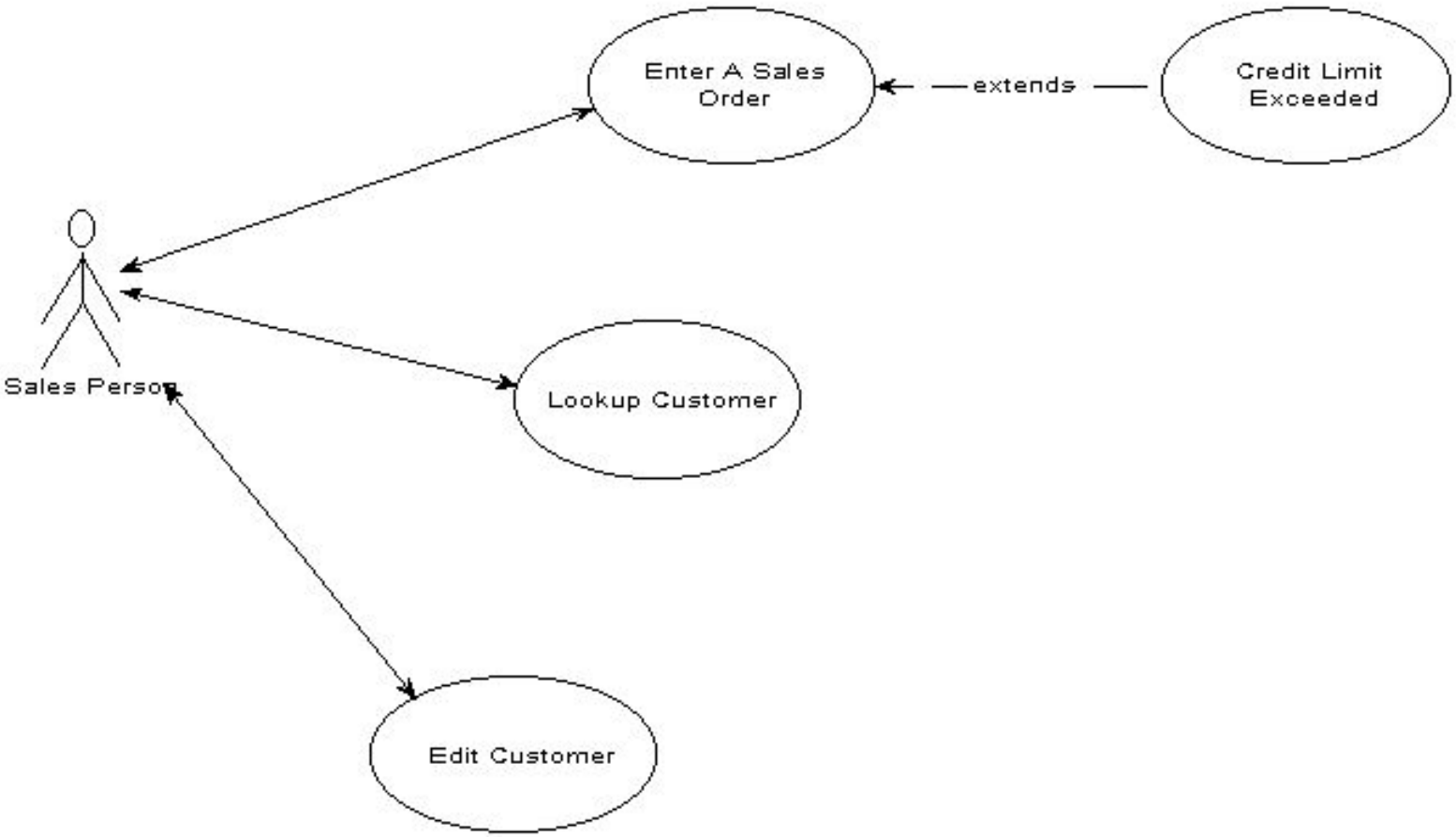
- **Read Only Users**
  - The read-only users will only read the database and cannot insert, delete or modify any records.
- **Read/Write/Modify Own Users**
  - This level of users will be able to insert new inspection details, facility information and generate letters. They will be also able to modify the entries they made in the past.

Use Case Example - 2

- **Read/Write/Modify All Users**
  - This level of users will be able to do all the record maintenance tasks. They will be able to modify any records created by any users.
- **Full Control Users**
  - This is the system administrative level which will be able to change any application settings, as well as maintaining user profiles.


Description
The Sales Order Clerk enters the customer's surname. The system displays all matching customers. The Sales Order Clerk selects one of those customers and the system displays that customer's details.
For each item that the customer wishes to order the Sales Order Clerk enters the line details. When all the line items have been entered the system confirms the order.
Enter A Sales Order
Description
The Sales Order Clerk enters the customer's surname. The system displays all matching customers. The Sales Order Clerk selects one of those customers and the system displays that customer's details. The system also displays the customer's payment history for the last six months.
Credit Check A Customer
Sales Order Clerk
extends
Description
In use case "Credit Check A Customer" if the system finds no customer's matching the surname given then the system displays an error message and allows the user to enter a different surname to search against.
Customer Not Found


Enter A Sales Order
extends
Credit Limit Exceeded
uses
Sales Person
Lookup Customer
uses
Find Customer
uses
extends
Edit Customer
Customer Not Found


Enter A Sales Order
extends
Credit Limit Exceeded
Sales Person
Lookup Customer
Edit Customer

Analysis Principles

- Information domain of problem must be presented & understood.
- Models depicting system information, functions, and behavior should be developed.
- Models and problems must be partitioned in a manner that uncovers detail in layers.
- Analysis proceeds from essential information toward implementation detail
- Must be traceable.

Information Domain

- Encompasses all data objects that contain numbers, text, images, audio, or video.
- Information content or data model
  - shows the relationships among the data and control objects that make up the system
- Information flow
  - represents manner in which data and control objects change as each moves through system
- Information structure
  - representations of the internal organizations of various data and control items

Modeling

- Data model
  - shows relationships among system objects
- Functional model
  - description of the functions that enable the transformations of system objects
- Behavioral model
  - manner in which software responds to events from the outside world

Partitioning

- Process that results in the elaboration of data, function, or behavior.
- Horizontal partitioning
  - breadth-first *decomposition* of the system function, behavior, or information, one level at a time.
- Vertical partitioning
  - depth-first *elaboration* of the system function, behavior, or information, one subsystem at a time.

Requirements Views

- Essential view
  - presents the functions to be accomplished and the information to be processed while ignoring implementation
- Implementation view
  - presents the real world realization of processing functions and information structures
- Avoid the temptation to move directly to the implementation view and assuming that the essence of the problem is obvious.

Specification Principles - 1

- Separate functionality from implementation.
- A process-oriented specification language is needed.
- Specification must encompass the system containing the software component.
- Specification must encompass the environment.
- System specification = cognitive model.

Specification Principles - 2

- Specification must be operational (talk about how it works).
- Must be tolerant of incompleteness and easy to add to.
- Specification must be localized and loosely coupled (pieces of things are independent of each other).

Specification Representation

- Representation format and content should be relevant to the problem.
- Information contained within the specification should be nested.
- Diagrams and other notational forms should be restricted in number and consistent in use.
- Representations should be revisable.

Specification Review

- Conducted by customer and software developer.
- Once approved, the specification becomes a contract for software development.
- The specification is difficult to test in a meaningful way.
- Assessing the impact of specification changes is hard to do.

Requirements Review

- Goals & objectives review.
- Compare requirements to goals & objectives.
- Consider system operating environment.
- Assess and document all risks in system developmental operation.
- Discuss testing procedure.

Evaluating Specification Techniques - 1

- Requirements are understandable to naive user.
- Requirements form basis for design and testing.
- Automated requirements checking?
- Requirements form external view of system.

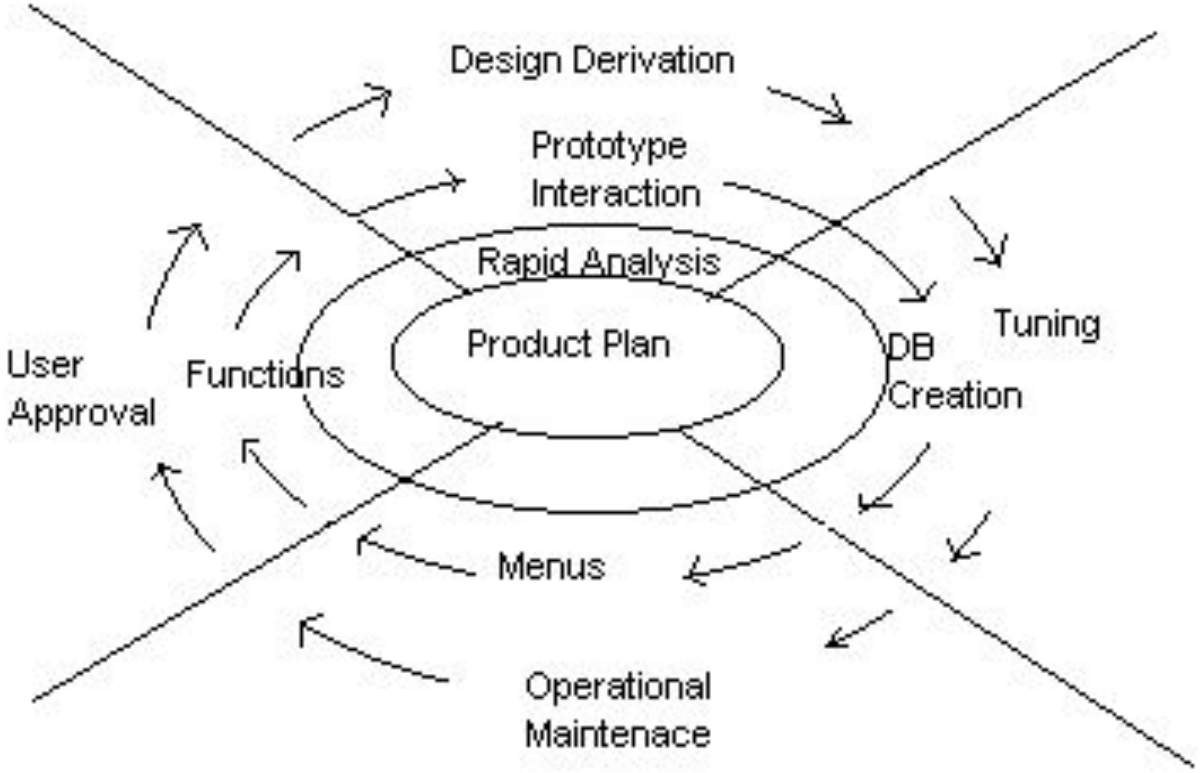
Evaluating Specification Techniques - 2

- Technique aid in organizing and structuring requirements.
- Technique for prototyping.
- Automatic test case generation from requirements.
- Appropriate for application.

Prototyping and Specification

- Throwaway prototyping
  - prototype only used as a demonstration of product requirements
  - finished software is engineered using another paradigm
- Evolutionary prototyping
  - prototype is refined to build the finished system
- Customer resources must be committed to evaluation and refinement of the prototype.
- Customer must be capable of making requirements decisions in a timely manner.

Evolutionary Rapid Prototyping



E.R.P. Process

- Goal is to write less throw away code than spiral model
- Starts with project plan and plans for software evolution
- Risks/Unknowns favoring ERP use:
  - Customer requirements
  - Technology
  - Interfaces

E.R.P. Risks

- Premature delivery.
- Premature design selection.
- Features in prototype can get lost in final product.
- There is a tendency to choose efficiency of production over product modifiability.

E.R.P. Advice

- Focus on what will take least amount of programming time over maintainability.
- With rapid prototyping don’t write code until ready.
- Spiral model any code written may be thrown out after risk assessment.

E.R.P. Analysis

- What.
- When.
- Cost.
- Completion.
- Re-use.
- Contingencies.
- Rewards.

Implementation and Testing

1. Code.
2. Test.
3. Debug.
4. Go to 1 (repeat).

Each Spin Cycle

- Each module is evaluated and rated:
  - Leave as is.
  - Rewrite.
  - Replace with existing code.
  - Discard, is no longer needed.

Re-implementation Symptoms

- Prototype contains spaghetti code.
- Prototype cannot be extended to meet full user requirements.
- Work to fix time implies less work to start again.