COM201: System Analysis and Design

Lecture 4: Analysis (understanding the problem)

Dr. Fayza A. Nada

1. Title

2. Overview

3. SDLC: Analysis Review

Purpose: Review analysis slide from lecture 2 (original content sourced from Satzinger et al.)

Notes:

- **The idea of problem domain. i.e, business area that requires computerized solution**
- **The analyst attempts to discover *what* the system needs to do? → understand the problem**
- **Review activities with a particular focus on first two activities**
- **Make a special mention that the key output for this phase are the *requirements***

4. A Poor Analysis Phase

Purpose: To look at reasons for why the analysis phase might not yield good information

Notes:

- **Recall that stakeholders typically refers to clients, users, and developers**
- **Requirements evolution relates to modifying existing system and problems with original requirements**
- **Discovery is highlighted because methods for discovering requirements will be discussed in next slides**

5. The Role of Systems Analyst

Purpose: To discuss the main actions undertaken by a systems analyst (the typical role for this kind of work)

Notes:

Emphasising the communication aspects of the role is important

The word *requirements* is highlighted again to illustrate the importance of them

6. Systems Analyst Relationships

Purpose: examine an image showing examples of specific stakeholder relationships

Notes:

- **The image illustrates the pivotal or central role the systems analyst plays**
- **Due to the different kinds of people a system analyst meets, they typically have a better overall understanding of the project scope**

7. Elicitation Techniques

Purpose: Introduce and briefly discuss common requirements elicitation techniques (methods)

Notes:

Requirements elicitation:

- A process by which analysts gather information on what the system should do from as many sources as possible
- *Synonyms:* discover, capture, investigation
- We require as many sources as possible to ensure a complete set of requirements
- Becoming an expert in the domain is important and the following techniques assist this process

Interviews:

- Really valuable method to get requirements and understand domain in detail
- 3 Steps: prepare interview, conduct interview, and follow up interview for clarification
- Useful to talk to: customers, suppliers, stockholders, expert users …
- Can interview individuals and groups (check consistency of requirements)
- Can have closed (prepared questions) and open (no questions) interviews

Surveys:

- *Used* to discover preliminary issues and requirements
- Can distribute to entire organisation relatively quickly even over a large geographical area
- Allows anonymous responses (in case people feel nervous about discussing problems in public)

- Big sample required for meaningful information (response might be low)
- Qualitative (open questions) are more descriptive and closer to an actual interview
- Quantitative (closed-questions) is measurable as you assign some kind of value
- Note that quality of answers might not be accurate but can still be a useful guide

Observations (ethnography):

- To gain a deeper understanding of how a user performs their work
- Can be difficult to get permission (why? → staff feel watched and made to feel nervous)
- Can discover what information is critical to achieving a particular task

Source documents:

- To see concrete examples of information use in the organization
- Examples include financial reports; forms such as order forms, invoices, receipts, purchase orders; procedure manuals and policies; IS manuals; charts
- Identify data items for database

8. Interviews are Great, however …

Purpose: To look at reasons **why *interviews* might not be successful** despite their importance

Notes:

- Point 1: Users cannot explain what they want
- Point 2: May not consider what is possible
- Point 3: Does not accept the project (resent)
- Point 4: Communication gap Actual speech or more accurately → technology jargon
- Point 5: No single user has all the answers.
- Point 6: Analyst requires certain skills and possess certain characteristics in order to discover the necessary information
- Point 7: Or can be aggressive which can result in users saying what they think the analyst wants

9. Examining a Source Document

Purpose: Demonstrate how a source document can yield useful information, in this case for designing the database

Notes:

- **Colour blocks are animated and will fly in to group related information and name them**
- **Illustrate the kind of information we can obtain and how we can use this information:**
- **Identify things of interest which eventually form aspects of the database.**

10. Domain Model (ERD)

Purpose: how form analysis can lead to a design. In this case an ERD (Entity Relationship Diagram)

Notes: The entity colours match the groupings from the previous slide

11. Modelling Business Activities

Purpose: An example of how work processes can be captured earlier using a modelling technique called Use Case diagrams

Notes:

- **A really simple diagram used in lots of different contexts**
- **Model behaviour of the system that eventually becomes services offered by the system**
- **Provides a high level view of what the system does and what types of people or systems interact with it.**
- **Useful analysis tool for use in any stage of requirements analysis.**
- **Aim is to be informal, simple and easy to understand.**
- **Represent a real world scenario**
- **Notation guide:**
    i. **Use cases: Business activity that comprises a series of smaller actions to support a goal. Effectively a function the system should provide**
    ii. **Actors: role a user plays, or external systems, sometimes databases**
    iii. **Associations:**
        1. **Communication between an actor and a use case to represent that actor initiates or receives some kind of value from a use case**

2. **<<Include>>: Association between a primary use case (main activity) and a secondary use case that indicates that the secondary activity be *must* performed as part of the primary (base) activity.**

3. **<<Extend>> (not shown): Association between a primary use case (main activity) and a secondary use case that indicates that the secondary activity *may* be performed as part of the primary (base) activity (usually as a result of some condition that you can represent in a more detailed model).**

iv. **Systems boundary box: grouped functionality or the set of activities for a particular scenario**

12. What are Requirements?

Purpose: Discussion of systems development project requirements

Notes:

- **Foundation of whole project, which is why they should be correct, complete etc.**
- **Describe just what is it that the system needs to do and what business activities need to be supported**
- **Also capture what kinds of information need to be stored, transformed, presented and so on.**
- **Recall stakeholders: anyone with an interest in the outcome of the development project.**
- **Through the rest of the lecture we will see why they are important, how to capture or discover them.**

13. Effects of Poor Requirements

Purpose: Consequences on a project of poor requirements elicitation, documentation or management

Notes:

- **Requirements related costs much greater.**
- **Why? Investment far greater during design and coding.**
- **Poor or incorrect requirements affects later stages of SDLC through exponentially increasing costs**
- **Therefore, investing time in effective requirements analysis early saves time, effort, and money**

- **Recall architecture and building a house (get plans right because it is much easier to change a plan than physical structure)**
- **So valuable that there is a new specialist role of Requirements Engineer who are responsible for discovering, document, validating, managing requirements.**

14. Characteristics of Good Requirements

Purpose: Briefly introduce specific characteristics that good requirements should possess

Notes:

- **Complete: They should include descriptions of all facilities required**
- **Precise: Exact and accurate**
- **Concise: No redundant words, diagrams etc. (no verbosity)**
- **Feasible: Should be able to implement requirement**
- **Testable: Should be able to show that this requirement has been met in test version of system**
- **Traceable: You know who suggested it, why it exists, its relationship to other requirements, etc. (check for potential effects on other requirements from changes)**
- **Consistent: There should be no conflicts or contradictions in the descriptions of the system facilities**
- **Correct: Match needs to users**
- **Atomic: Requirement cannot be decomposed further**
- **Unambiguous: Requirement has a single interpretation only (important for outsourcing coding to another country)**
- **Question now is what are the types of requirements?**

15. Types of Requirements

Purpose: Discusses **distinction between *functional and non-functional* requirements (VERY IMPORTANT SLIDE)**

Notes:

- Many classifications of requirements (Examples are based on a library system)
  - ***A function is something a system or subsystem does. Some activity.***

- **Non-functional describes** all other aspects of system such as performance, reliability, usability, portability, data, implementation, etc.

16. Types of Requirements

Purpose: The first slide looking at different types of non-functional requirements **(VERY IMPORTANT SLIDE)**

Notes:

- **General requirements:** sets out in broad terms what the system should do
- **Data requirements:** define the type of data the system shall operate upon or produce
    - Characteristics of data items.
- **Usability requirements:** state user interface and system availability constraints

17. Types of Requirements

Purpose: Second slide looking at examples of non-functional requirements

Notes:

- **Implementation requirements:** states how the system must be implemented
- **performance requirements:** specify the minimum acceptable performance of the system
- **Operational requirements:** specify constraints that should be satisfied during system usage

18. Expressing Requirements

Purpose: A quick overview of how requirements are represented after discovery

Notes:

- Now we have discovered the requirements, **how do we document them?**
- In this course, most requirements will be documented **using models** and **natural language descriptions**
- **Z Notation**
    - A formal specification language used for describing and modeling computing systems. useful for describing computer-based systems