CMSC 128
Introduction to Software Engineering
2nd Semester AY 2016 – 2017
5 – Capturing Requirements from Stakeholders

Reginald Neil C. Recario
Institute of Computer Science
University of the Philippines Los Baños

0

CMSC 128
Introduction to Software Engineering

CMSC 128 Introduction to SE

- What are stakeholders?
- What are requirements?
- Techniques for capturing requirements
- References
- Credits

1

What are stakeholders?

What are stakeholders?

Stakeholders are individuals who are directly or indirectly involved and can directly or indirectly influence the development of a software.

Example:

Who are the stakeholders of SystemOne?

What are stakeholders?

Possible stakeholders:

- Students
- Faculty members
- Administrators
- OUR

What are stakeholders?

Example:

Who are the stakeholders of Metrobank Online?

What are stakeholders?

Possible stakeholders:

- Depositors / Cardholders
- Account Analysts
- Bank Managers
- IT Personnel

2

What are requirements ?

What are requirements?

Why do you need to capture/elicit requirements?

What are requirements?

- Bad requirements essentially translates to a bad product
- There is a need for a formal and structured process for developing and managing requirements

What are requirements?

Requirements development (engineering): (1/2)

- requirements elicitation: techniques for seeking, extracting, acquiring and documenting requirements
- requirements analysis

What are requirements?

Requirements development (engineering): (2/2)

- requirements specification
- requirements maintenance

Reasons for bad requirements

(1) Inexistence of a formal process by which the requirements are collected or elicited

What are requirements?

Reasons for bad requirements

(2) Serious breakdown in the personal communications between the different requirements stakeholders, and mainly between the client and user representatives and the analyst.

Reasons for bad requirements (3)Inexistence or lack of a formal requirements validation process

Reasons for bad requirements

(4)Failure in the management of the requirements engineering processes

What are requirements?

Reasons for bad requirements

(5)Lack of use of tools to deal with requirements for large software projects

What are requirements?

Reasons for bad requirements

(6)Lack of application domain experience of the requirements development team members

What are requirements?

Types of requirements

- Functional requirements
- Non-functional requirements

What are requirements?

Functional requirements

- Functionalities and services provided to users to meet their needs and to achieve business objectives
- Tangible, visible

What are requirements?

Non-functional requirements

- Constraints on the provision of functional requirements
- Quality requirements: system wide or function specific

Elicitation of functional requirements (1/4)

- Understand client’s needs and communicate them to developers

Elicitation of functional requirements (2/4)

- Extraction, discovery / invention, acquisition/ elaboration of stakeholders needs

What are requirements?

Elicitation of functional requirements (3/4)

- System analyst, business analyst, requirement engineer or requirement facilitator
    - Under and over specification of requirements

Elicitation of functional requirements (4/4)

- System analyst…(cont.)
  - Understand application domain, identify sources of requirements and stakeholders, selection of elicitation techniques

3

Techniques for Capturing Requirements

Techniques for capturing requirements

Interviews

Questionnaires

Task analysis and scenarios

Preliminary requirements domain analysis Brainstorming

Workshops

Meetings

Prototyping

Ethnographic assessment: describe what is being done by observing or practicing

Techniques for capturing requirements

Let us use SAIS as a working example for this section.

Techniques for capturing requirements

> Interview

- Informal technique used at the beginning of the elicitation process to engage the various participants.

Techniques for capturing requirements

> Interview

- Individual interviews are used to establish the initial mission statement and goals of the system being developed.

Techniques for capturing requirements

> Interview

- Interviews can be structured, unstructured, or hybrid.

Techniques for capturing requirements

> Interview

- Careful formulation of interview questions is needed to avoid negative impact on the participants.

Techniques for capturing requirements

> Questionnaires

- Used at the beginning of the elicitation process to poll the participants and stakeholders and to obtain their thoughts on issues related to the system.

Techniques for capturing requirements

> Questionnaires

- Questions must be clear, concise, and appropriate to the issues of the project
- – multiple choice, true and false, or open-ended questions.

Techniques for capturing requirements

> Questionnaires

- Answers can be used by the requirements engineer to identify initial conflicting requirements requests that need to be resolved later.

Techniques for capturing requirements

> Task analysis technique

- Identification of the system uses and the description of the interactions between the users and the system.

Techniques for capturing requirements

> Task analysis technique

- Interactions can be described using scenarios or flow of events.

Techniques for capturing requirements

> Task analysis technique

- Top-down analysis:
  - first high level system tasks are identified.
  - then refined into lower level tasks and sub-tasks for which scenarios have to be elicited.

Techniques for capturing requirements

> Task analysis technique

Task analysis requires domain knowledge expertise and an understanding of the tasks performed by similar systems.

Techniques for capturing requirements

> Preliminary requirements domain analysis

- Preliminary requirements and domain analysis techniques start from an initial set of requirements.

Techniques for capturing requirements

> Preliminary requirements domain analysis

- Preliminary requirements are proposed by the expert requirements engineer and are passed to the various stakeholders for discussion.

Techniques for capturing requirements

> Preliminary requirements domain analysis

- The engineer must be an authority in the domain of application.

Techniques for capturing requirements

> Brainstorming

- Based on brainstorming sessions: informal meetings to discuss the desirable functionalities and services.

Techniques for capturing requirements

> Brainstorming

- Inventive ideas that might or might not be feasible are generated.

Techniques for capturing requirements

> Brainstorming

- All ideas are encouraged and the merits are discussed by the requirements analyst and other participants later in more formal settings.

Techniques for capturing requirements

> Brainstorming

- The end product is an initial set of software requirements.

Techniques for capturing requirements

> Group meeting

- A group meeting includes all participants and stakeholders to start the elicitation process.

Techniques for capturing requirements

> Group meeting

- The formation of the group and the dynamics of the interactions among group participants are crucial for the success of this technique.

Techniques for capturing requirements

> Group meeting

- Cohesiveness, openness, and transparency are key communications features during group meetings.

Techniques for capturing requirements

> Workshop

- Formal settings used to elicit software requirements.

Techniques for capturing requirements

> Workshop

- Participants –a facilitator, a technical writer, the requirements engineer, and representatives from the client and users' groups.

Techniques for capturing requirements

> Workshop

- The aim is to reach an agreement on a working draft of the software requirements.

Techniques for capturing requirements

> Workshop

- Must be prepared thoroughly ahead of time and expectations from each participant must be explicitly known.

Techniques for capturing requirements

> Workshop

- Elicit best case and worst case scenarios and fit them in the various use cases of the use case model.

4 References

References

Marsic, Ivan. Software Engineering. 2012.

Pressman, Roger. Software Engineering: A Practioner's Approach (7th ed). 2010.

Saleh, Kassem. Software Engineering. 2009.
Sommerville, Ian. Software Engin eering (10th ed). 2015.

Sommerville, Ian. Software Engineering (9th ed). 2011.

And other Software Engineering books and auxiliary online references

Thanks!

Any *questions* ?

You can find me at

- @RegRecario
- rcrecario@up.edu.ph

5 Credits

Credits

Special thanks to all the people who made and released these awesome resources for free:

- Presentation template by SlidesCarnival
- Photographs by Unsplash