Software Engineering COMP1035

Lecture 06

Requirements Validation

Today’s Learning Outcomes

1. Requirements Documents
   - and who uses them.
2. Final Requirements Validation

Requirements Engineering

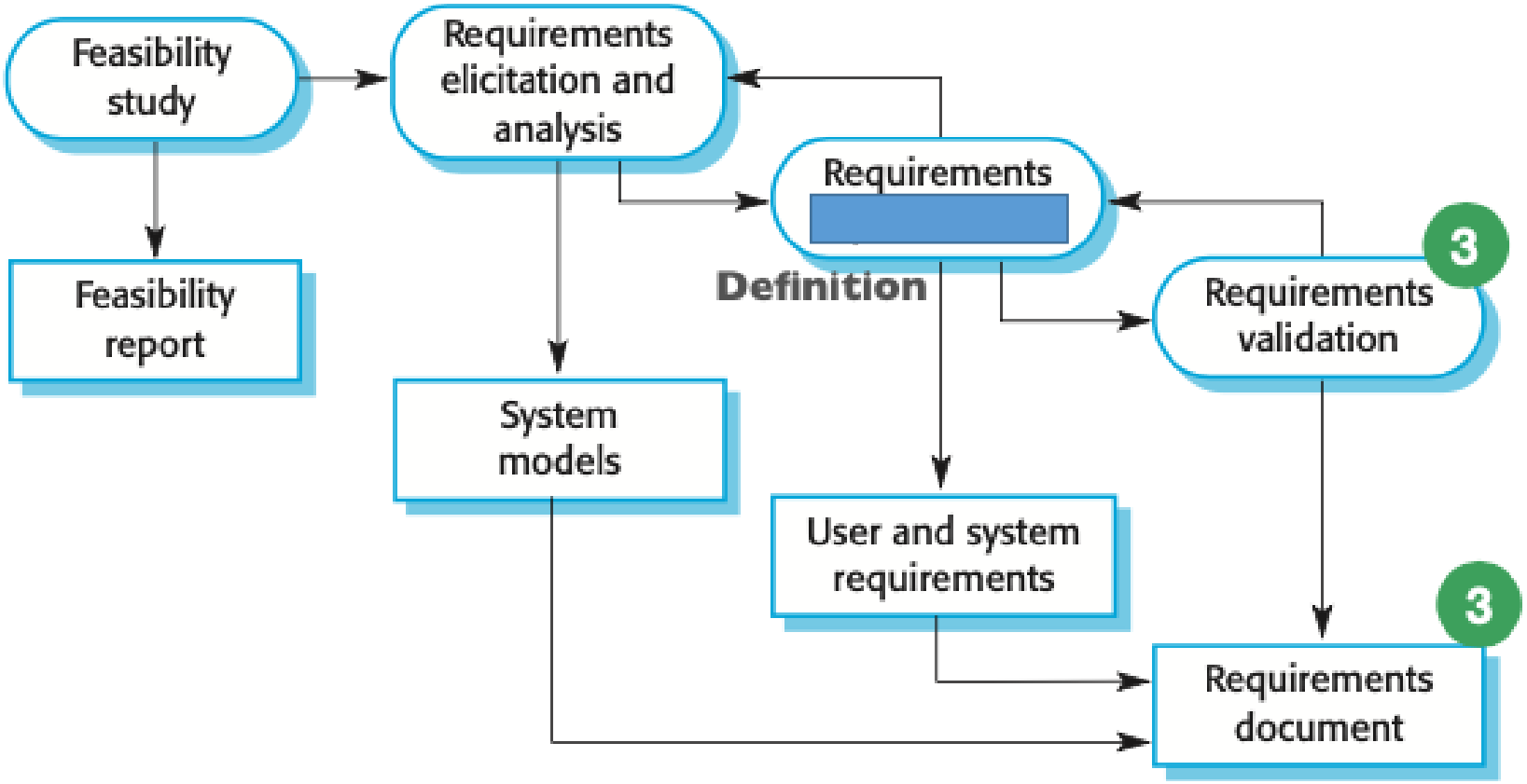


Figure 2.4 The requirements engineering process

Requirements Documents

So What Are We Documenting Into? And Why?

Requirements / Specifications Documents

- SRS = Software Requirements Specification

- IEEE Standard 830-1998.

Table of Contents

Requirements / Specifications Documents

- Translation from anecdotal user wishes to a formal document.

a) Specific requirements should be stated in conformance with all the characteristics described in 4.3.
b) Specific requirements should be cross-referenced to earlier documents that relate.
c) All requirements should be uniquely identifiable.
d) Careful attention should be given to organizing the requirements to maximize readability.

- Usually, a big set of nested lists, with unique IDs.
- Diagrams produced go with them (and cross reference those IDs).
- Lists should be categorised – e.g., importance, risk etc.
- Usually define the “acceptance testing” at the end of the project.

Chapter	Description
Preface	This defines the expected readership of the document and describe its version history, including a rationale for the creation of a new version and a summary of the changes made in each version.
Introduction	This describes the need for the system. It should briefly describe the system's functions and explain how it will work with other systems. It should also describe how the system fits into the overall business or strategic objectives of the organization commissioning the software.
Glossary	This defines the technical terms used in the document. You should not make assumptions about the experience or expertise of the reader.
User requirements definition	**Here, you describe the services provided for the user. The nonfunctional system requirements should also be described in this section. This description may use natural language, diagrams, or other notations that are understandable to customers. Product and process standards that must be followed should be specified.**
System architecture	This chapter presents a high-level overview of the anticipated system architecture, showing the distribution of functions across system modules. Architectural components that are reused should be highlighted.
System requirements specification	This describes the functional and nonfunctional requirements in more detail. If necessary, further detail may also be added to the nonfunctional requirements. Interfaces to other systems may be defined.
System models	This chapter includes graphical system models showing the relationships between the system components and the system and its environment. Examples of possible models are object models, data-flow models, or semantic data models.
System evolution	This describes the fundamental assumptions on which the system is based, and any anticipated changes due to hardware evolution, changing user needs, and so on. This section is useful for system designers as it may help them avoid design decisions that would constrain likely future changes to the system.
Appendices	These provide detailed, specific information that is related to the application being developed—for example, hardware and database descriptions. Hardware requirements define the minimal and optimal configurations for the system. Database requirements define the logical organization of the data used by the system and the relationships between data.
Index	Several indexes to the document may be included. As well as a normal alphabetic index, there may be an index of diagrams, an index of functions, and so on.

Figure 4.17 The structure of a requirements document

Requirements / Specifications Documents

ID	Description	Importance
F1	First requirement description	High
F2	Second requirement description	High
NF1	Third requirement description	Medium
F3	Fourth Requirement description	Low
NF2	Fifth requirement description	High
F4	Sixth Requirement description	High

Who Looks at SRS Documents

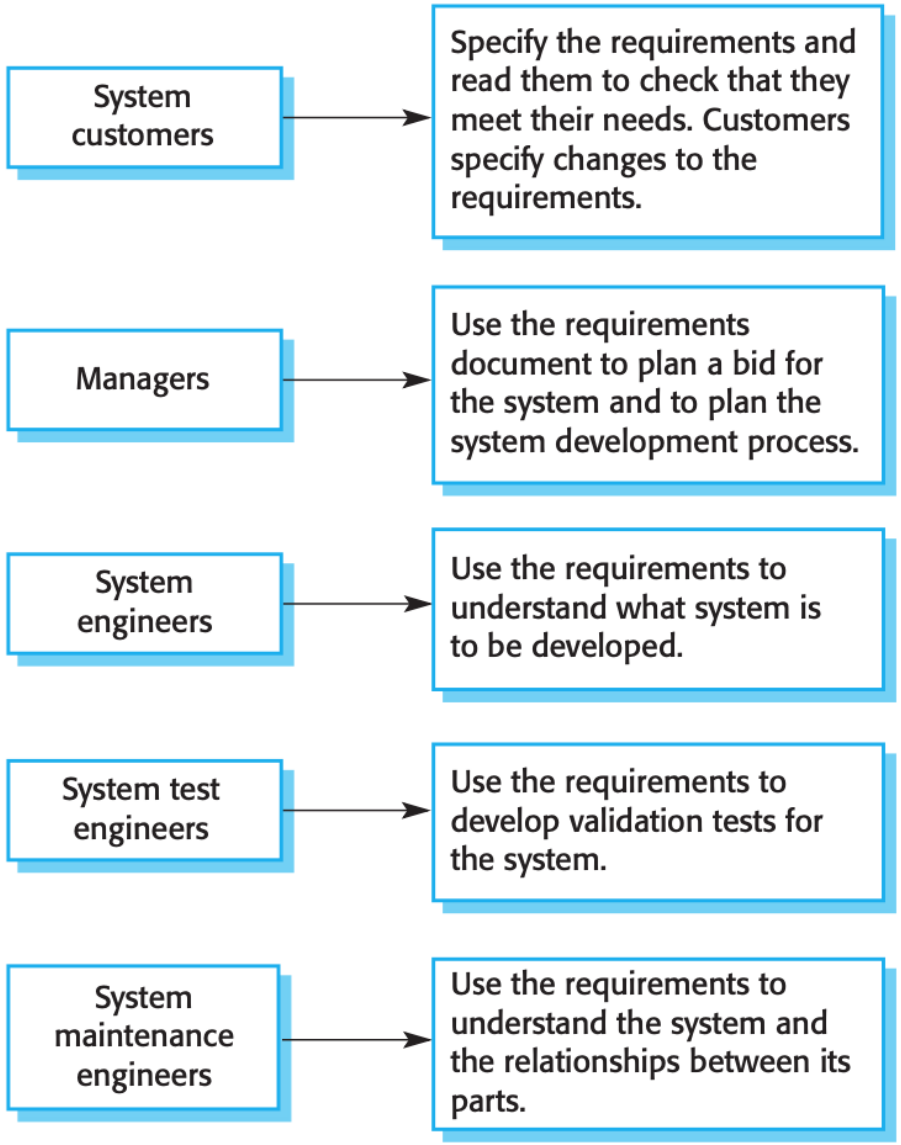
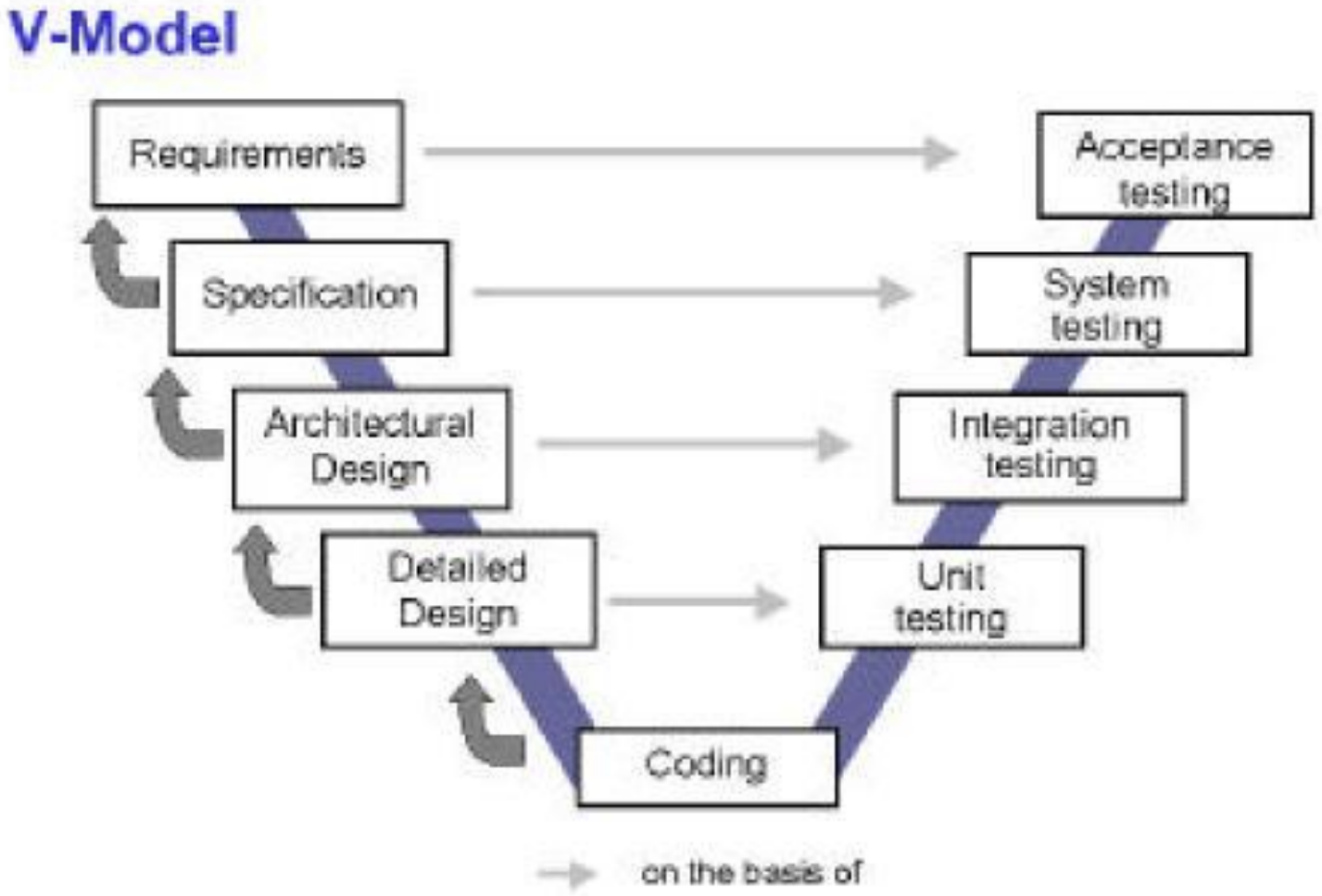


Figure 4.16 Users of a requirements document

Who Looks at SRS Documents



http://sqa.fyicenter.com/FAQ/Software-Development-Models/Software_Development_Models_V_Model.html

Requirements Validation

Why We Do Requirements Validation

1. Checking that you are right.
2. Avoiding ReWorking.
3. Contractually Agreeing.

1. Checking That You Are Right

"… the process of checking that requirements actually **define the system that the customer really wants**"

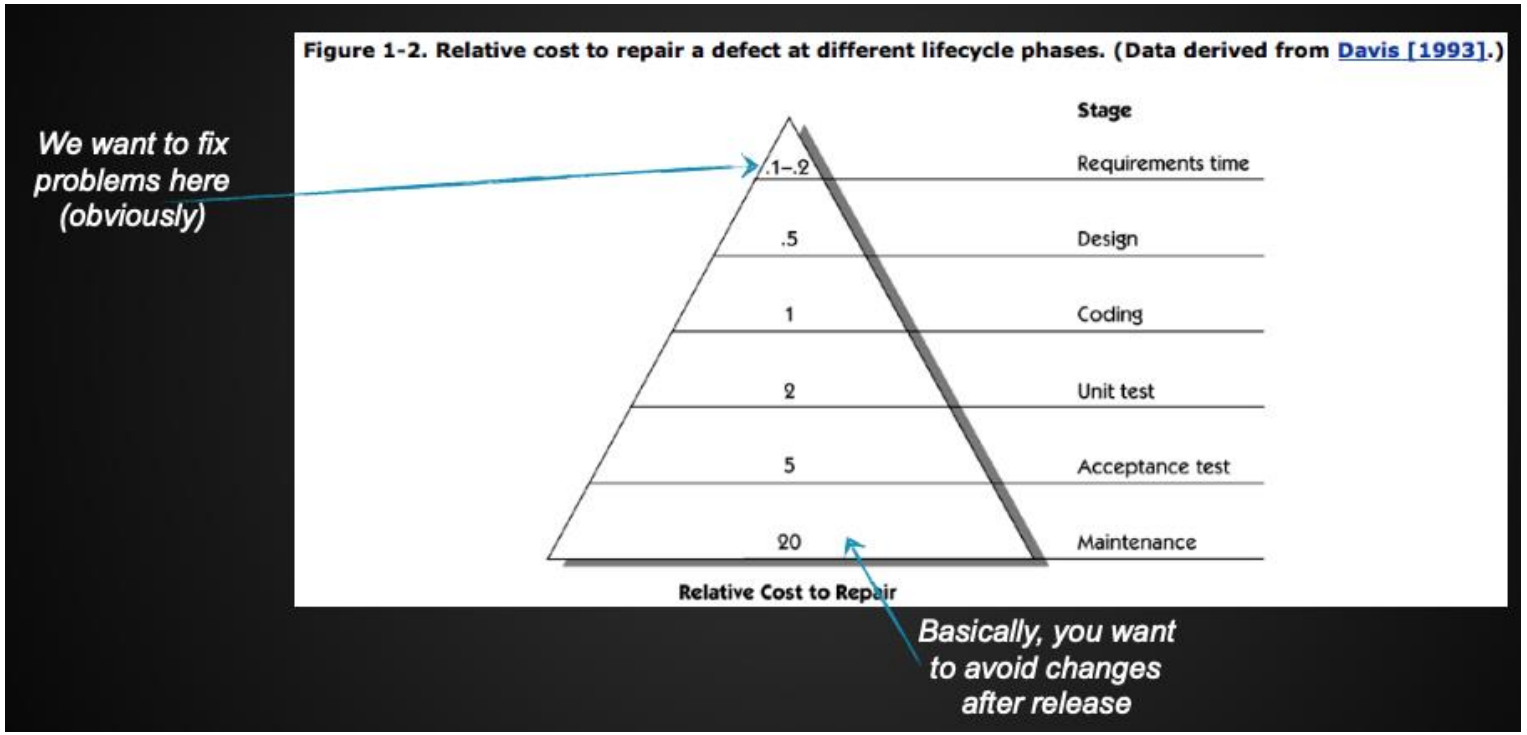
- The SE Book

2. Avoiding ReWorking

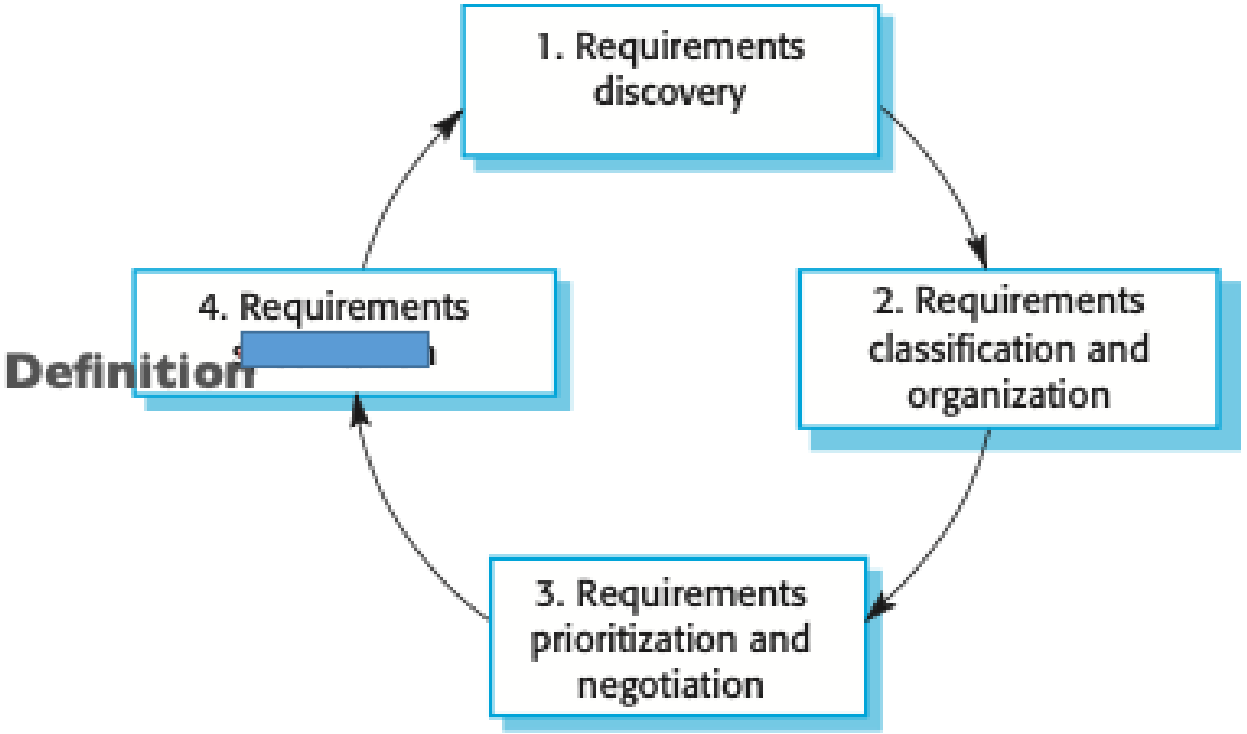
"… errors in a requirements document can lead to extensive rework costs … The **cost fixing a requirements problem by making a system change is usually greater than repairing design or coding errors**."

- The SE Book

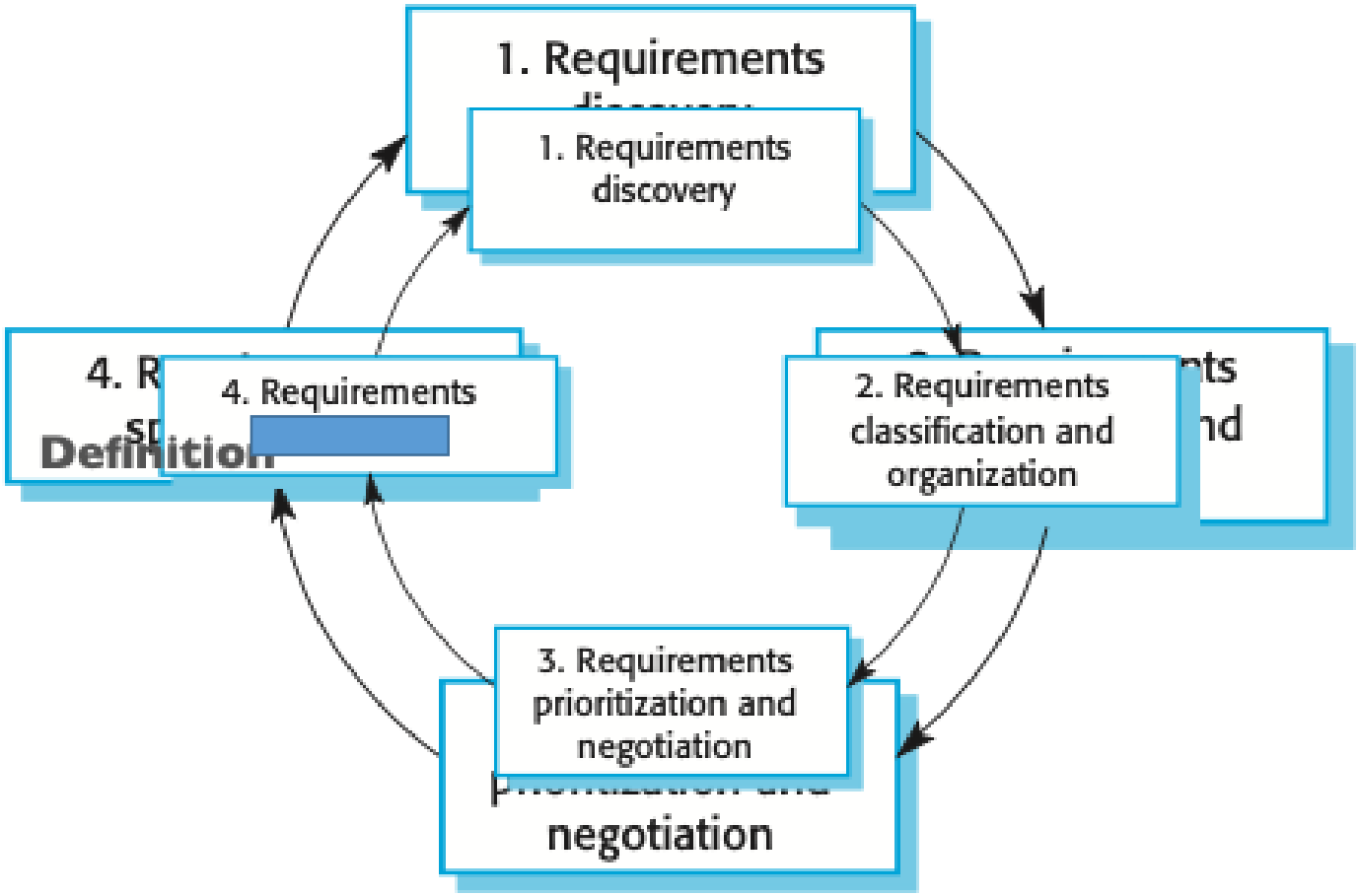
2. Avoiding ReWorking



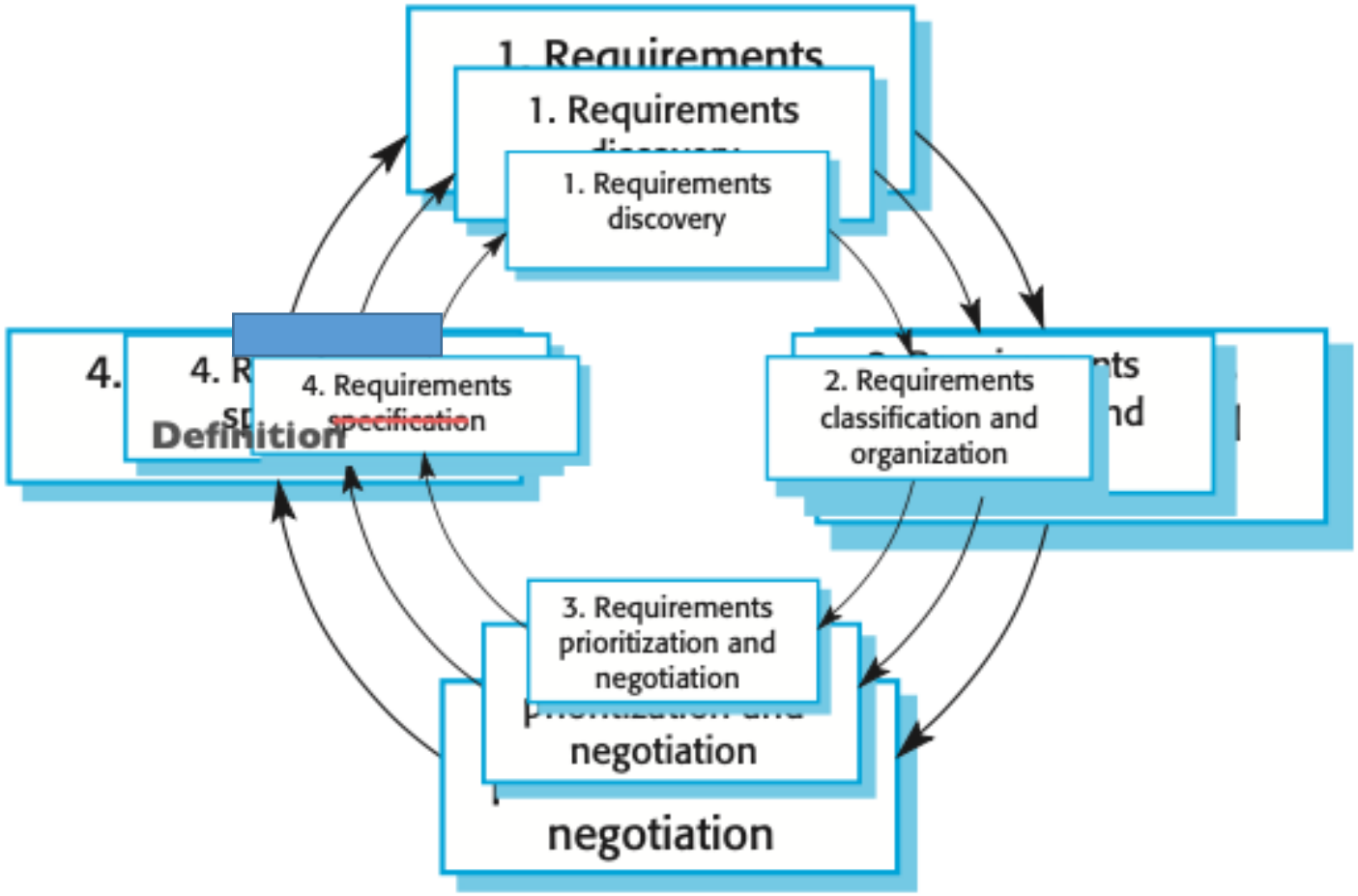
Requirements Validation Cycle



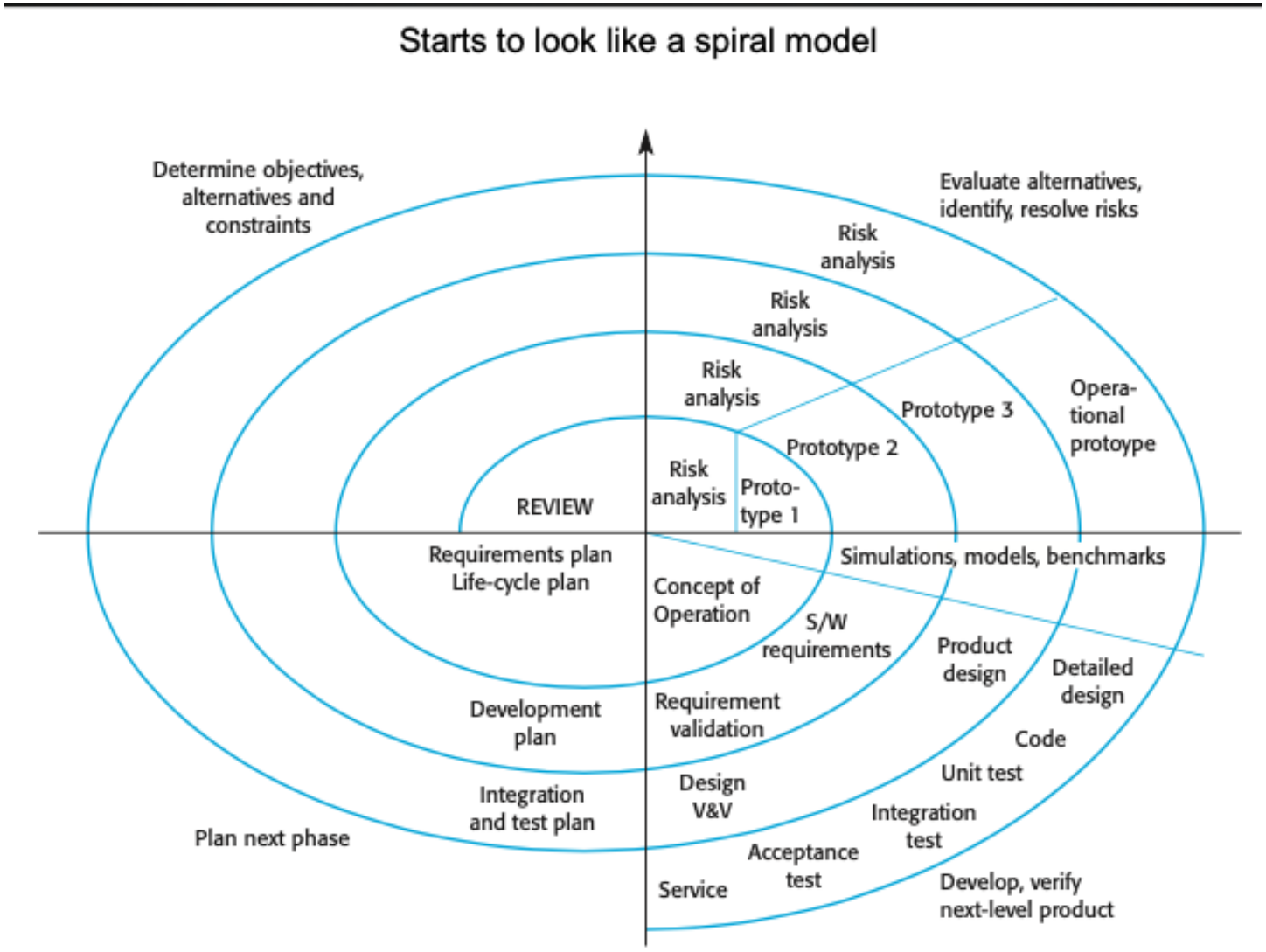
Requirements Validation Cycle



Requirements Validation Cycle



Requirements Validation Cycle



3. Contractually Agreeing

- At some point in a project, you must decide what **exactly what to build**.
- If this is for a customer, you want everyone to agree exactly what will be built.
  - Otherwise, you and the customer may have vastly different ideas.
  - You cost for your idea.
  - **But they don't pay until their idea is achieved**.

Requirements Validation

- You present it to your “boss and colleagues”.
  - You had to explain things to the audience.
  - This is a first sanity check.
  - Does it make sense when you tell your “boss”.

Internal

- You present it back to participants/clients/users.
  - Do they agree with your understanding?
  - Do they agree with what you think is “most important”.

External

Requirements Validation – **Internal**

- Using a focused method – a Requirements Review.
- Reviews appear in several stages.
- You want to do this with your team first.
  - It’s like a practice run for when you present things to your client.
    - The full requirements, the time plan, the requested budget to achieve it, etc.
- Get a manager, the Requirements leader, a developer, a quality manager, and the client manager (if different) together in a room.
- Two benefits
  1. The client manager gets a clear picture before taking it to the client.
  2. If you can explain it to them, without having trouble, then you are ready to take it to the client.

https://camilofitzgerald.wordpress.com/

Req. Conflicts?

- What happens when you find gaps in your understanding?
  - You are not ready to move to external validation!
- If it's a missing aspect, then you need to do more elicitation.
- It it's a conflict, you need to document the conflicting ideas.
  - And resolve with client.
- Don't go to full external validation until you are ready.

Example (Adapted): Electronic Library

Context:
"The purpose of this project is to create an attractive user-friendly prototype for a virtual archive (i.e. a virtual framework for virtual items or collection groups within a larger collection) of research materials".

Requirement A: Item Retrieval
"This option allows the user to retrieve items in any format".

Requirement B: No File Conversion
"File conversion should not be supported".

Requirements Partitioning:
Requirement A tagged as a *usability* requirement
Requirement B tagged as a *cost / schedule* requirement

Conflict Identification:
QARCC's expert knowledge system flags the possibility of a conflict due to the fact that *usability* and *cost / schedule* requirements typically stand a good chance of conflicting with each other. The conflict is then verified by the development team with the following issue: "*What is meant by any formats? It may not be possible to retrieve in any format since file conversion will not be supported".*

Resolution Generation:
Done manually, options as follows:

a) Support file conversions for all major types and increase the budget for the project.
b) Support file conversions for limited set of formats (e.g. .pdf, .rtf and .doc) and increase the budget for the project.
c) Make budget and schedule allowances by removing Requirement F: *"Provide a wizard feature for setting up archives".*
d) Add the requirement: "*A separate version of each item, one for every format required, must be submitted to the system when a new item is added".".*
e) Only support the retrieval of items in formats that are available.

Resolution Selection:
The developers proposed e) to be the best option and this was agreed upon by all stakeholders.

Requirements Validation – External

- But at some point, you've got to agree a plan with a client.
  - Essentially a Requirements Review with the client.
  - And you don't want to look like a fool when you do (hence the internal validation first).
- This time the people in the room are the key people from both companies.
  - Probably not a developer, and tester and Quality Manager, etc.
  - But e.g., the manager, and finance manager from the client's side.
- By the end of this final validation, you should have 'the plan'
  - Budget, time, requirements etc.
  - To the best of your ability.
  - Because if you are wrong, it's only going to create delay, or take you over budget.

Requirements Validation – and Beyond

- There is one way to validate requirements.
- By moving on – to the next Stage – Specifications
  - Nothing highlights gaps in your understanding more.
  - Then trying to decide how to build the system that meets the requirements.
  - But by then it's already more expensive.
    - **Agile** techniques are designed to bring design earlier to avoid this – more later in the module.
- Next lecture, we move on to Specifications.

Summary

- Requirements Documentation –
  - Importance? Priority?
  - Software Requirements Specifications (SRS)
- Requirements Validation – Methods?
  - How to resolve conflict?


THANK
YOU