

Software Engineering

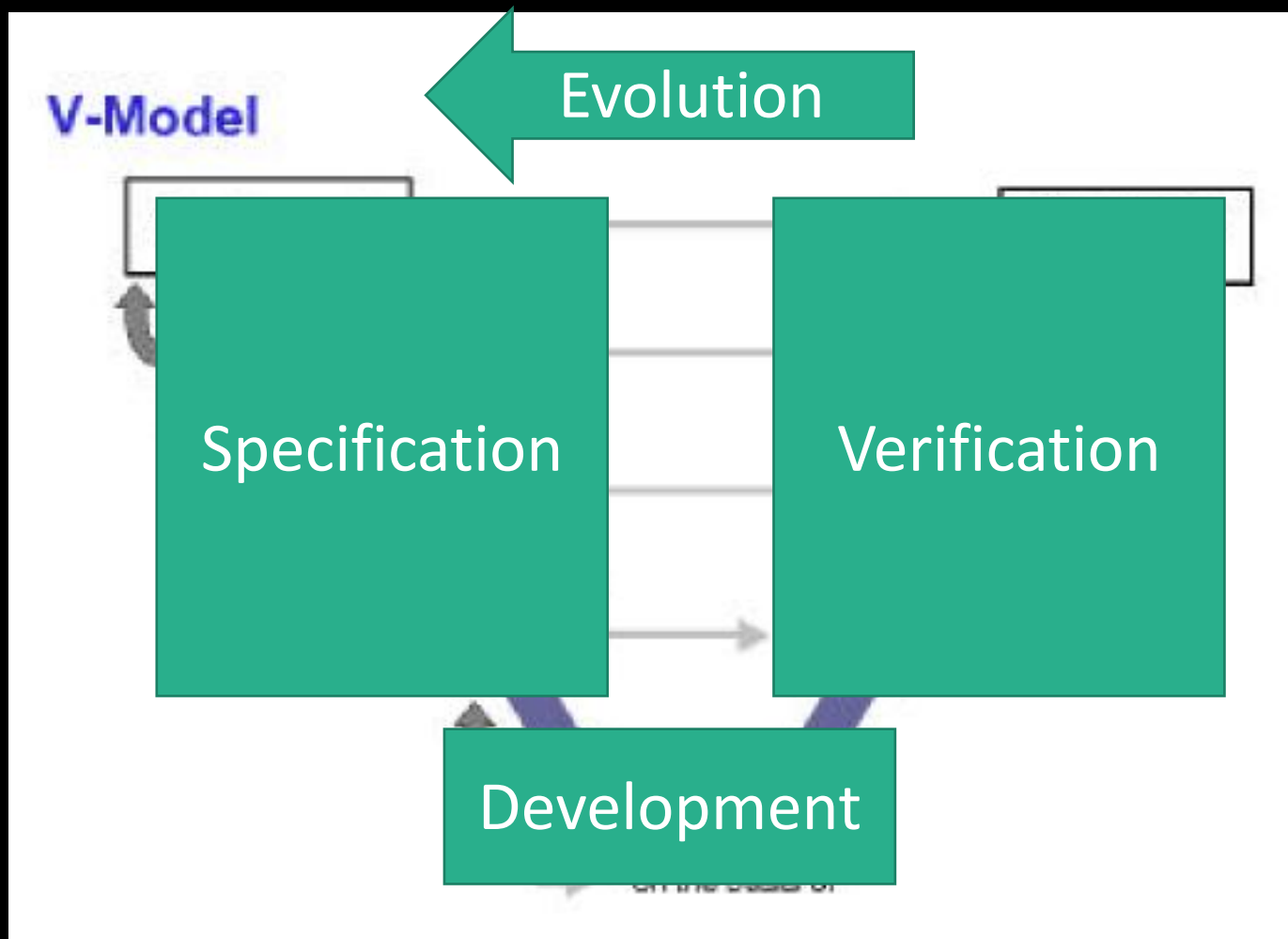
COMP1035

Lecture 02

Requirements

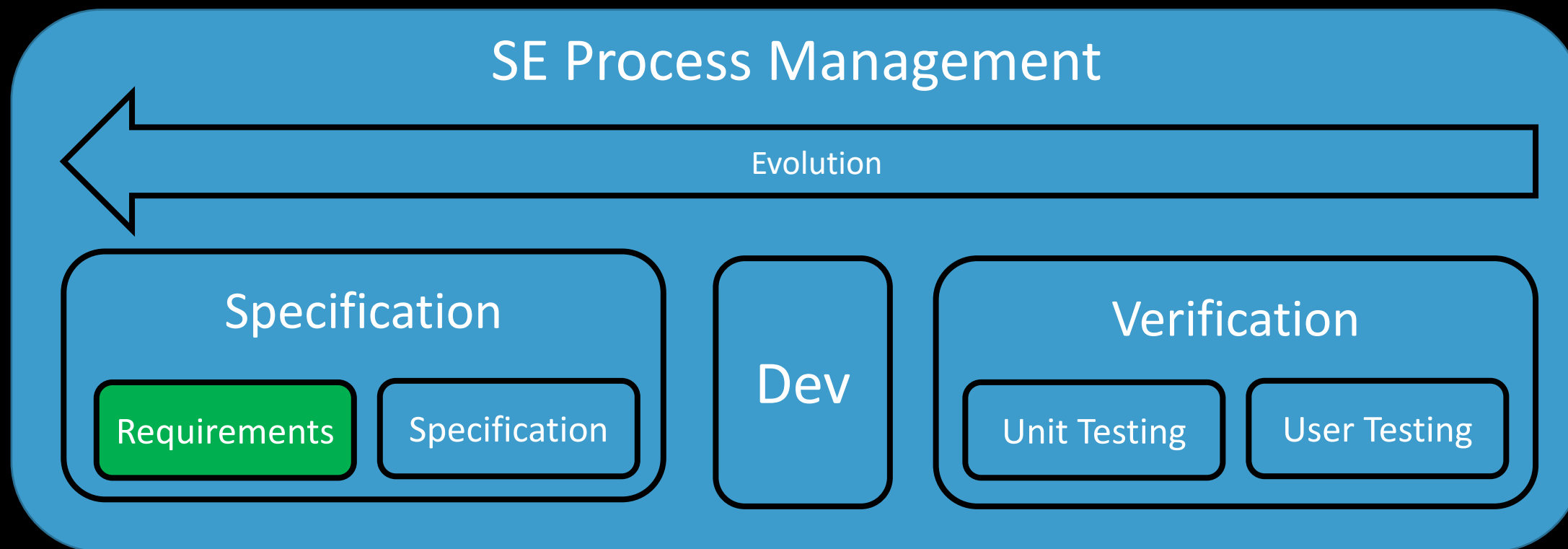


Keeping Track of SE Module





Keeping Track of SE Module



1. Why requirements are so important.
 - That Over-Budget costs are **mostly** associated with Bad Requirements.
2. What 'Requirements Engineering' is.
3. Introduction to 'Requirements Elicitation' stage.
4. Initial Methods for identifying requirements areas.

Why Requirements are Important



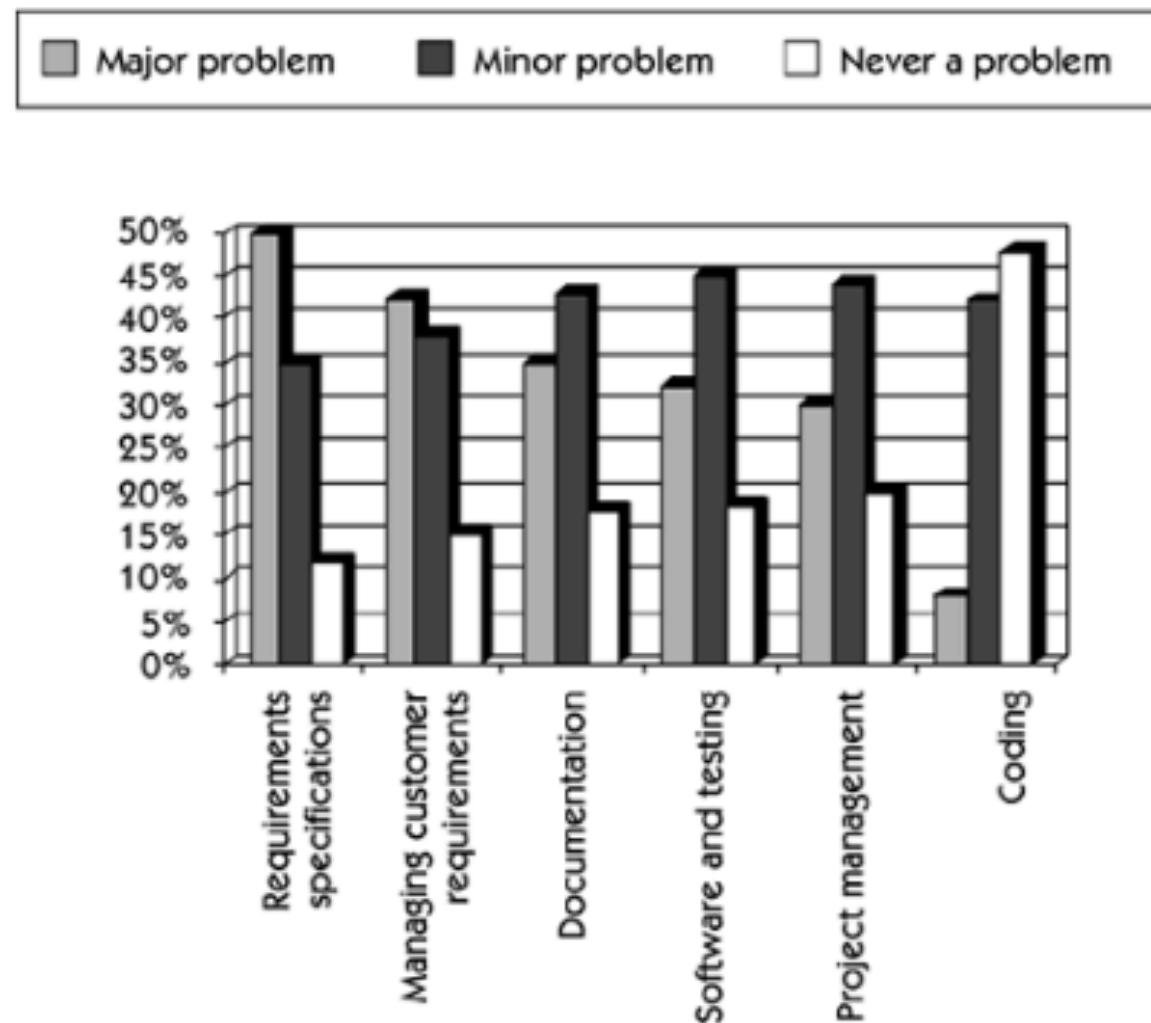
Ongoing SE Project Failurees

- The global cost of IT failure was estimated at +6 trillion dollars in 2009. (source: Roger Sessions, The IT Complexity Crisis: Danger and Opportunity)
- Only 32% of software projects were “successfully completed” (i.e., on time, on cost, and with expected functionality) in 2009. (Source: Standish CHAOS 2009 Update)
- Only 16% of software projects were successfully completed in the UK in 2009. (Source: British Computer Society)
- “A failing industry ...
 - If building engineers build buildings with the same care as software engineers build systems, the first woodpecker to come along would be the end of civilization as we know it.” (Source: Paul Dorsey, Top 10 Reasons Why System Projects Fail)

- **Incomplete requirements** and **lack of user involvement** rank top of the list of causes of project failure. (Source: Standish CHAOS 1995 report)
- Understanding product requirement is **the major problem** in software development.
- Coding or programming is not a major problem.

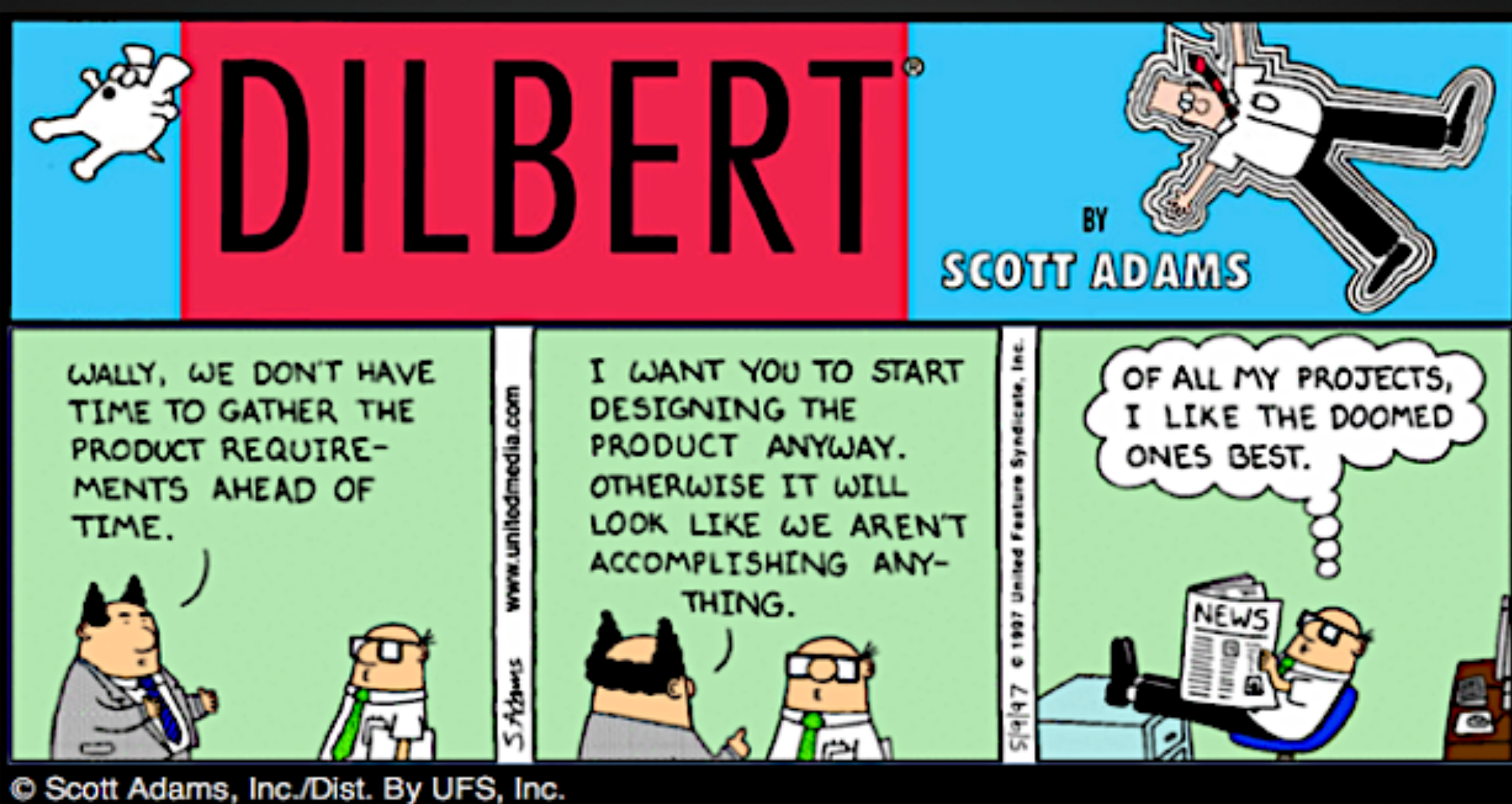
What Causes Failure?

Figure 1-1. Largest software development problems by category. (Data derived from [ESPITI \[1995\]](#).)





What Causes Failure?

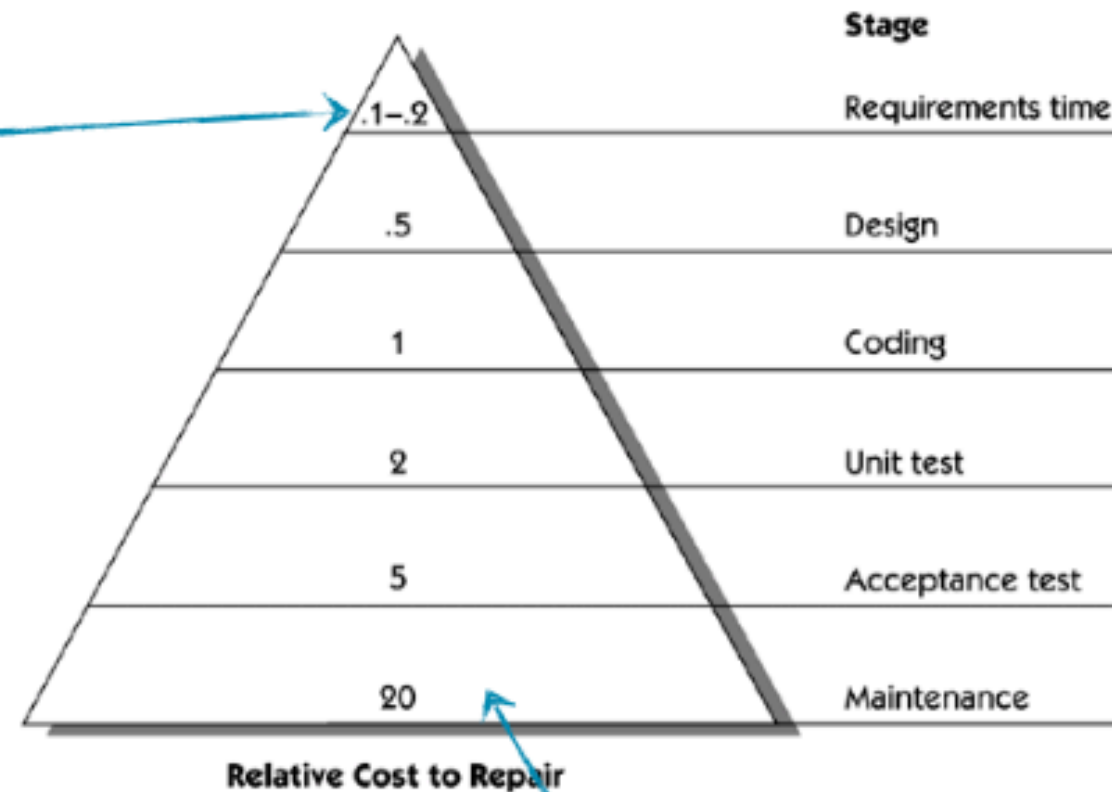


- “**The hardest single part of building a software system is deciding precisely what to build ...** No other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later.” (Source: Fredrick Brooks, No Silver Bullet: Essence and Accidents of Software Engineering)
- As much as a **200:1 cost savings** results from finding errors in the requirements stage versus finding errors in the maintenance stage of the software lifecycle.



What Causes Failure?

Figure 1-2. Relative cost to repair a defect at different lifecycle phases. (Data derived from [Davis \[1993\]](#).)



We want to fix problems here (obviously)

Basically, you want to avoid changes after release

The Problem



The Requirements Problem

- You all want to build software.
 - But there is a **great deal more** to doing that successfully than learning to cut code.
 - Understanding “stakeholder” requirements (i.e., client, customer, user and other interested parties needs) is a key to effective systems development.
 - That means you are going have to learn **how to identify their requirements.**
 - Which means you are going to have to learn to work with human beings as well as machines ...



The Requirements Problem





The Requirements Problem

- “I would go a step further and assert that is really impossible for clients, even those working with software engineers, to specify completely, precisely, and correctly, the exact requirements of a modern software product.” (Source: Fredrick Brooks, No Silver Bullet: Essence and Accidents of Software Engineering)
- We need to be able to **determine them**, by **working with** clients.
- Learning how to do requirements engineering:
 - In more technical terms its about “requirements engineering” or figuring out what **stakeholder needs** and what **services** a product will need to provide **to meet those needs**.



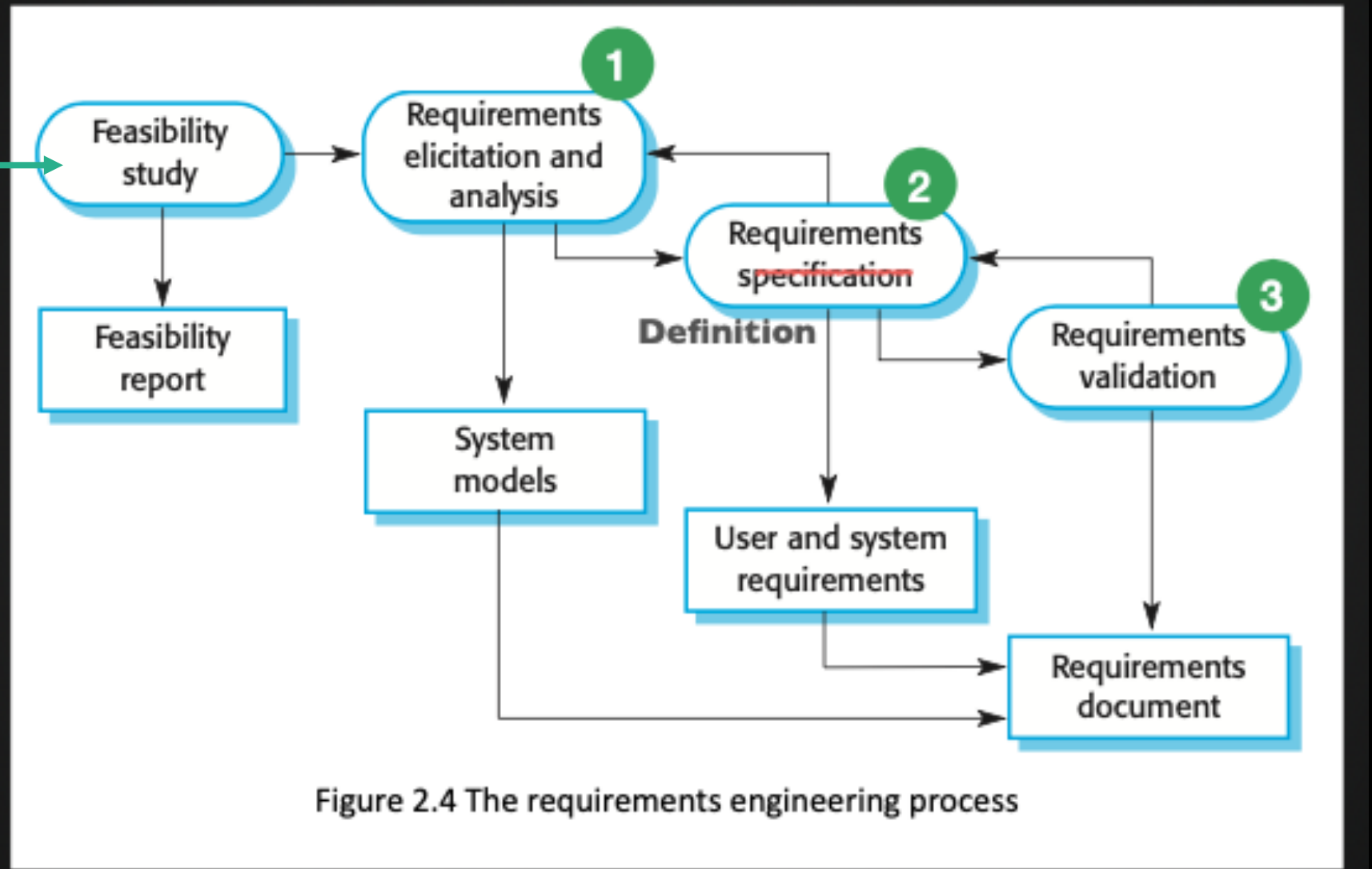
The Requirements Problem

- **“Requirements are your project’s foundation.** They define the level of quality you need, facilitate decision making, provide a basis for tracking progress, and serve as the basis for testing.”
- “If you let them remain unstated, you have no opportunity to examine and negotiate them with your customer and no way to tell when your project has met its objectives.” (Source: Brian Lawrence, Top Risks of Requirements Engineering)

What is Requirements Engineering?

Requirements Engineering

Short-term, relatively cheap studies that inform the decision of whether or not go ahead with a more detailed analysis.



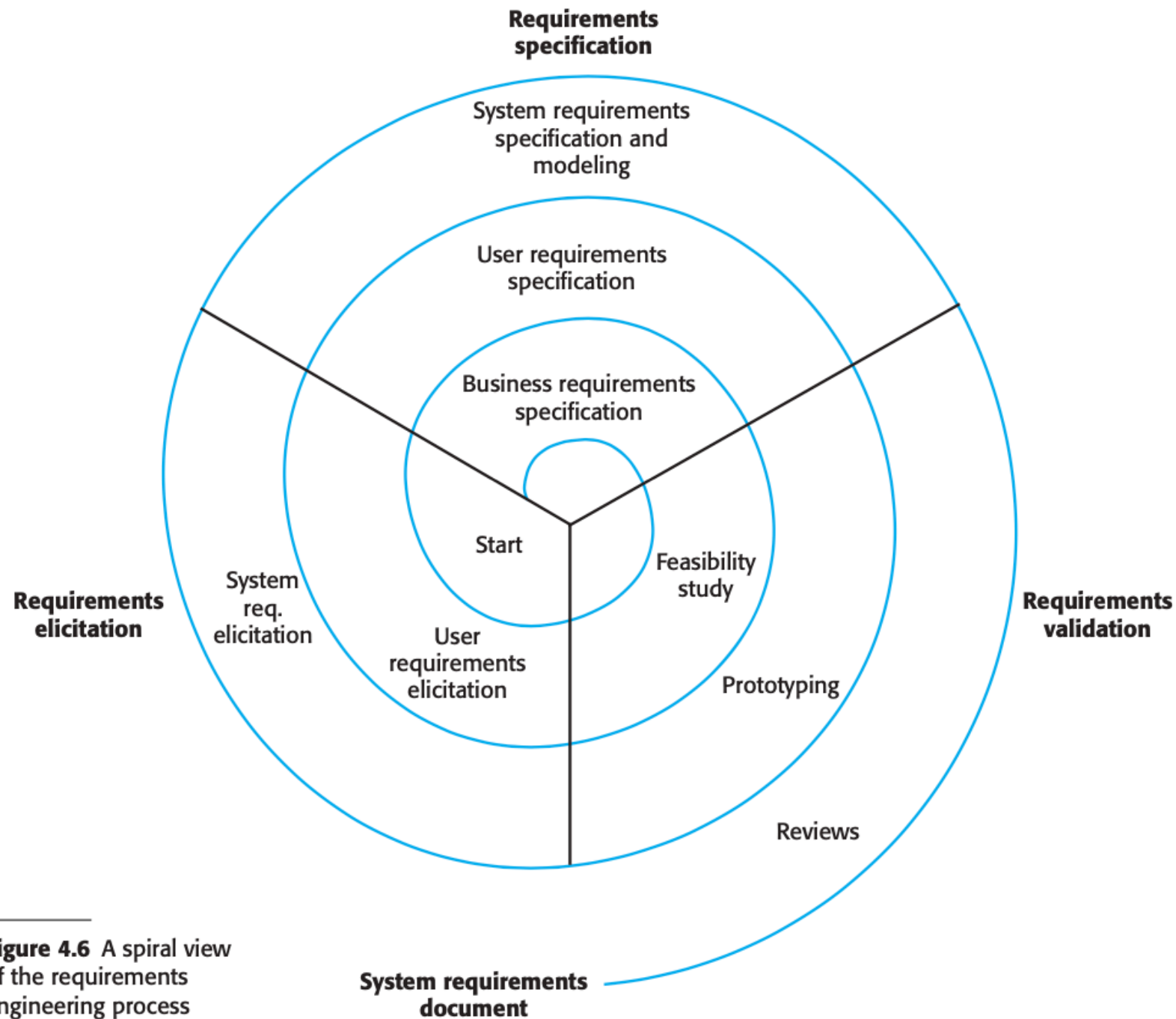
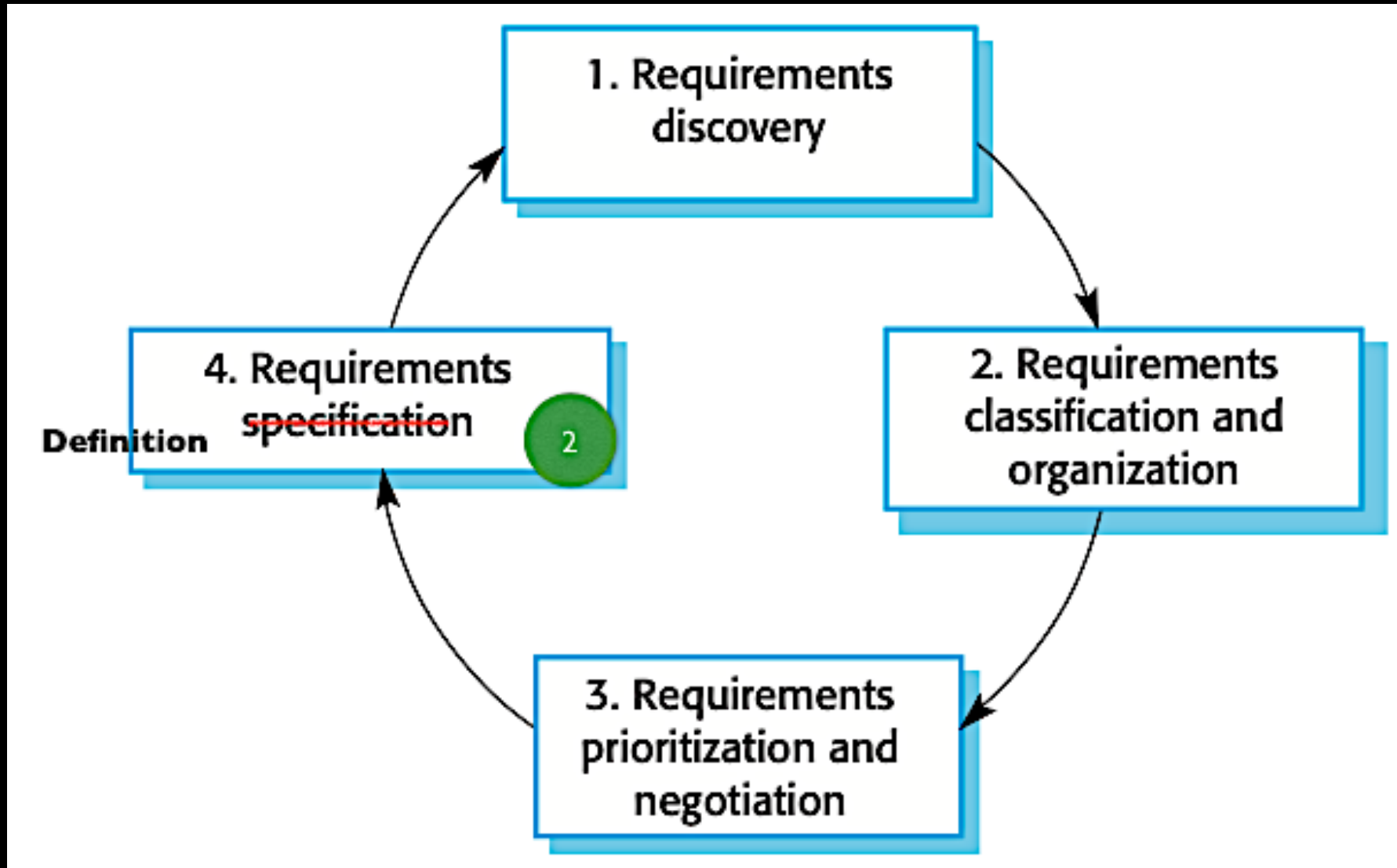


Figure 4.6 A spiral view of the requirements engineering process

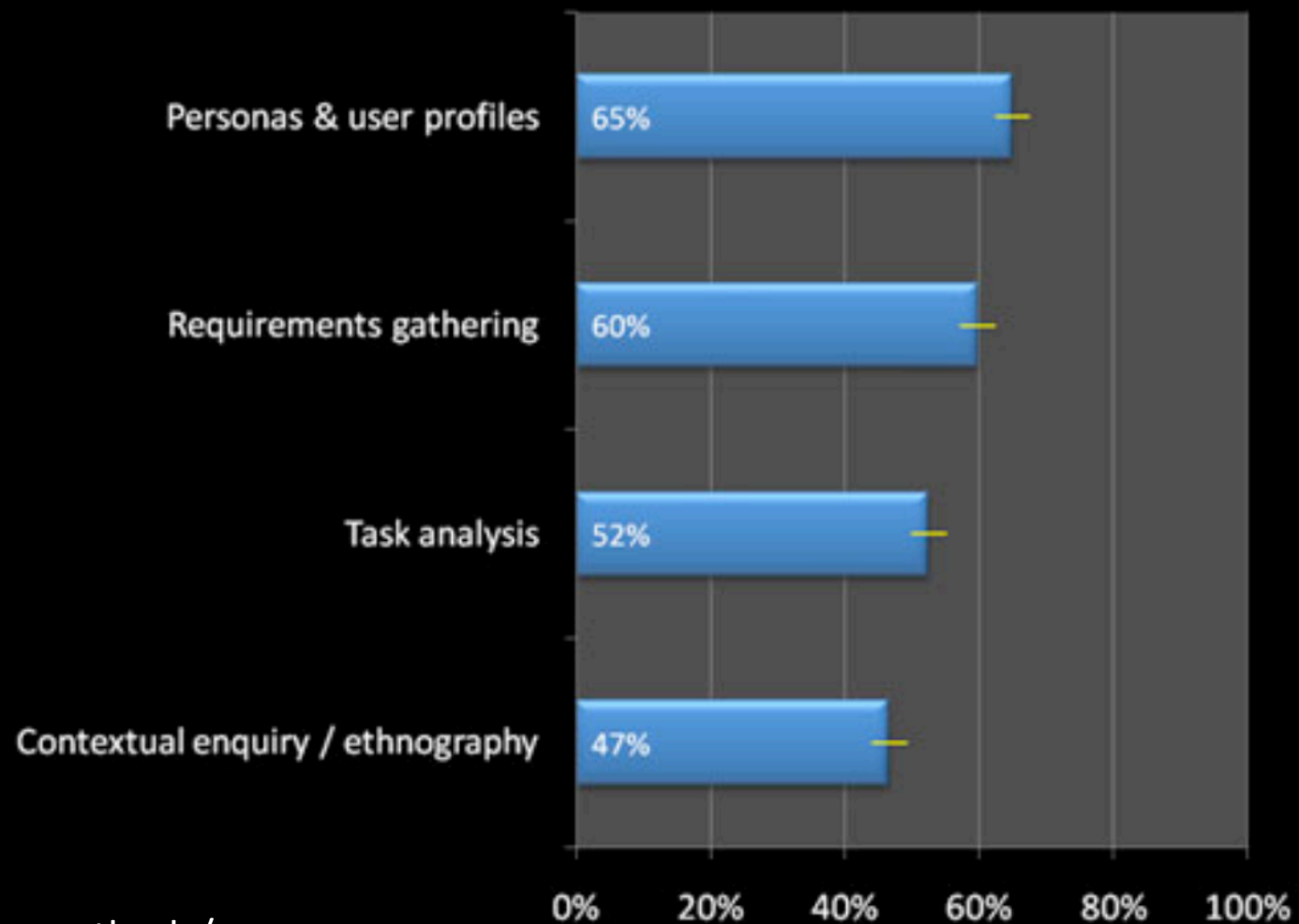
Requirements Elicitation



Requirements Elicitation



Initial Methods for Requirements Elicitation



<https://measuringu.com/ux-methods/>

- Final Stage: Determine **all the people** that will use the system.
- From a mixture of
 - Working with initial briefs
 - Problem analysis
 - Interviews/discussions
- Have to identify
 - Primary stakeholders
 - Secondary stakeholders
 - Tertiary stakeholders

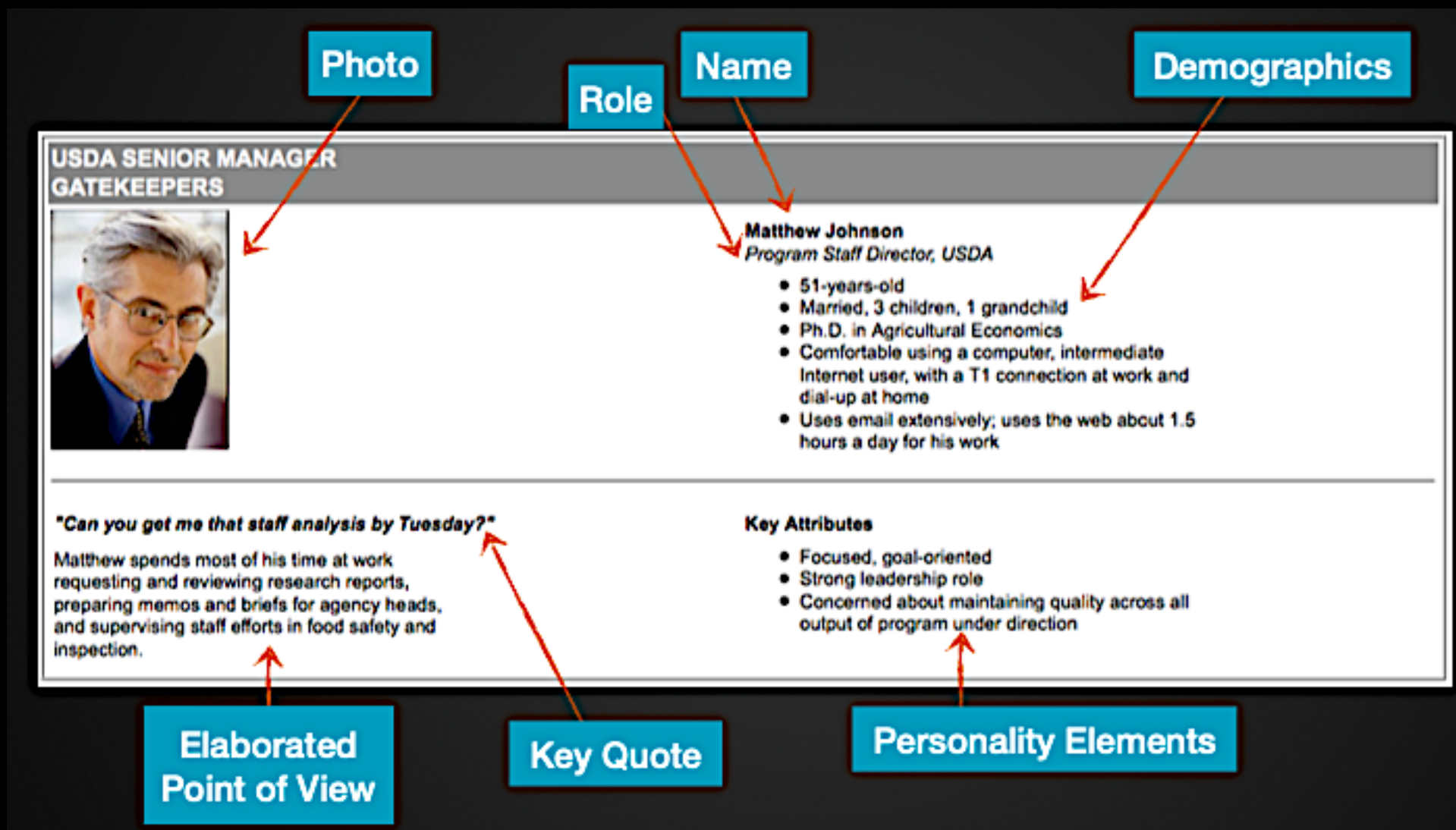
- There are all sorts of ways of classifying/analysing them (beyond primary, secondary, tertiary).
- Importance/Priority.
- Impact of their needs.
- You can list example people.
- You can list what you want from them.



- Represents a real **type of user** from your stakeholders.
- **Main Aim: Personas should differentiate stakeholders clearly.**
- Identify: motivations, expectations, goals, knowledge.
- Usage: Help you to put yourself in the shoes of ...
- Ideally: try to have a small number – all one page at most.



Personas





- Not a report on real people from the client company
- Choose a generic representative name/photo/etc.
- Make 2 or 3 that demonstrate **key user types**.
- It might be that several stakeholders can be represented by one persona, e.g., lecturers / module convenors / supervisors.
- It might be that one stakeholder type – needs several personas, e.g., regular gamers, casual gamers, high-spenders.



- “Externalisation”
 - SE mistakes are often from “what was not said and not documented”.
- At design time: Would “Jim” use this?
 - Stops designers designing for themselves.
- Prioritisation of requirements.
 - And asking: Does this design satisfy all 3 personas.
- Developing scenarios (next lecture)!



Persona Board

<https://wiki.fluidproject.org/display/fluid/Persona+Format>

Kivio Users

	The researcher	The Sysadmin	The OSS developer	The CS student
				
Name	Alexander Weiß	Donald M. Berry	Kristian Larsson	Eric Neville
Age	30	30	26	24
Location	Germany	US	Sweden	France
Social Life	Alexander lives with his girl-friend in a flat in Hamburg.	Donald lives with his wife and 1-year old daughter in a house in Portland.	Kristian shares an apartment with two friends in Stockholm. His girl-friend lives in Uppsala. They see each other every weekend.	Eric lives with his parents in a small city close to Lyon. He visits the university there. Often, he stays at his friend's apartment for playing PC games and programming.
Work Life	He works at centre for environmental systems research and designs plans for replacable energies in a EU-funded project.	He is a lead system administrator in a huge network solutions company in Portland.	A software developer with a dayjob in a medium-sized software company. Works on KDE in his spare time.	He is a student of computer science. Besides university, he performs small programming jobs for people in his neighbourhood.
Computer Experience	All are highly experienced with computers.			
Time at a computer per week	26-50 hours per week	35-50++ hours per week	30-50++ hours per week	25-45 hours per week
Computer tasks	Office tasks and Field-dependent. Also educational and recreational. No development.	Development and network administration. Does not use PC for office tasks, educational, and even recreational.	Mostly development and recreational. Also network administration and office.	Mostly development. Also educational, recreational, and network administration. Does not use for office work.
Relation to OSS.	He is not passionate about OSS.	He is a convinced user of OSS.	He is involved with OSS development.	He is a convinced user of OSS.
Requirements wrt diagramming	office requirements	highest claims	easy-going	eager beaver
Frequency of drawing diagrams	Each 2 nd month	Twice per month	Once a month	Each 2 nd month
Diagram main type	Flowcharts. Also visualising thoughts. No technical ones.	All, except sitemap	Visualising thoughts	Diagrams mostly UML
Size and complexity	15-20 elements, 2-3 levels, 3-7 shapes	15-30 elements, 2-5 levels, 4-9 shapes	15-20 elements, 2-3 levels, 3-7 shapes	15-20 elements, 2-3 levels, 3-7 shapes
Diagram purpose and context.	For non-IT job.	Diagrams are for the IT-Job, never for himself.	Diagrams are mostly for himself but, also for formally presenting.	For formally presenting in university. Not for himself, as work input or for any jobs
Current diagramming	Power Point or OOo.	Visio.	Pen and Paper.	Dia, Umbrello.

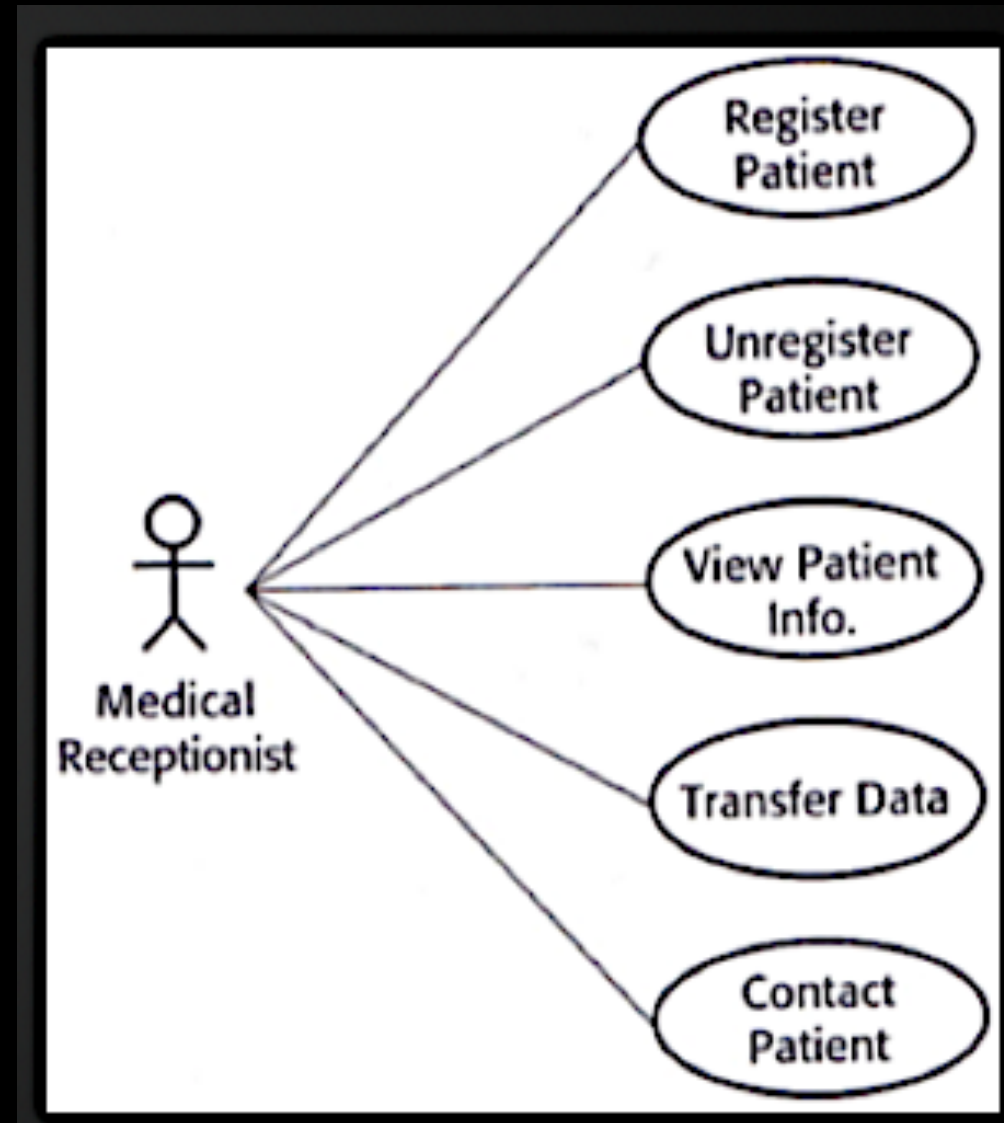


Persona Board

TEST CASE APPLICATION PERSONAS				
PERSONA	THINKING	HEARING	SAYING	DOING
DEVELOPER (R)	HAS MY STUFF BROKEN? DID I BREAK THE BUILD IS MY STORY "DONE"		WHAT IS REQUIRED FOR TESTING?	CREATING TEST SUITES, TEST CASES & TESTS UPDATING THEIR OWN TESTS
QA ENGINEER (R)	AM I ON TRACK TO MEET TESTING REQS			EDITING OTHER PEOPLES TESTS MAINTAINING RISKS
TC4 ADMINISTRATOR				CREATE TESTED ITEMS, TEST TGTS MANAGE CATEGORIES.
QA MGR	"STUFF NEEDS TESTING" "DOES TEST PLAN HAVE ALL TEST CASES" "RISE PROBLE FOR PARTS OF PRODUCT" "IMPLEMENTING RIGHT TYPE OF TESTING"		"HOW MANY TESTS PASSED/FAILED ON A DAILY BASIS" "WHY DID THIS FAIL"	PREPARING TEST PLANS
QA DIRECTOR			WHAT IS THE DELTA CHANGE FROM YESTERDAY	REVIEWING DASHBOARD APPROVING A REQUEST TO GO

<https://scrumprout.tumblr.com/post/30516447738/building-the-persona-board>

- We need to elaborate the tasks that each Stakeholder will do.
- The most common method for this is a 'use case' diagram.
- They represent the people who use the system and the tasks they have to perform.
- We call the people 'actors'.



Use Case Diagrams

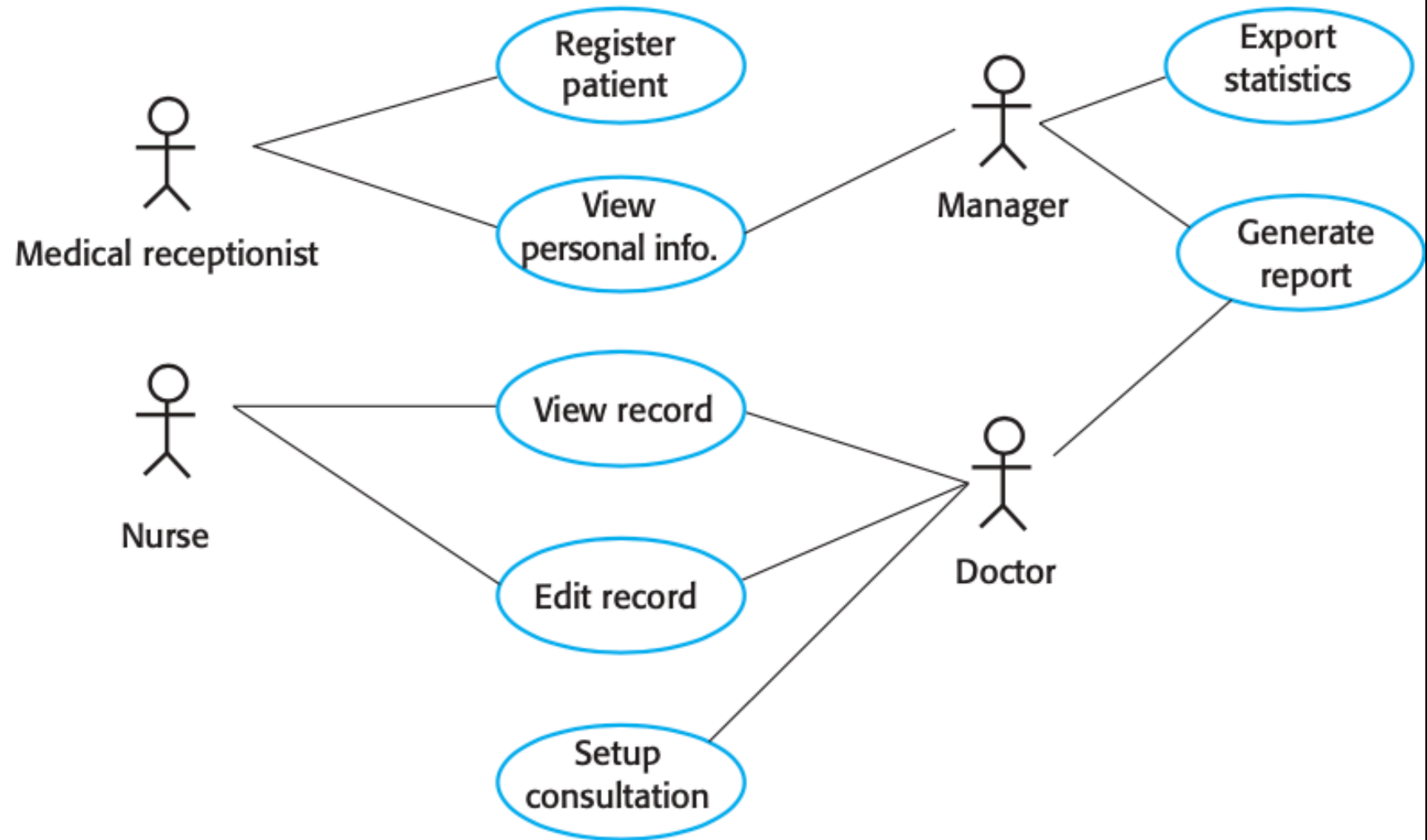
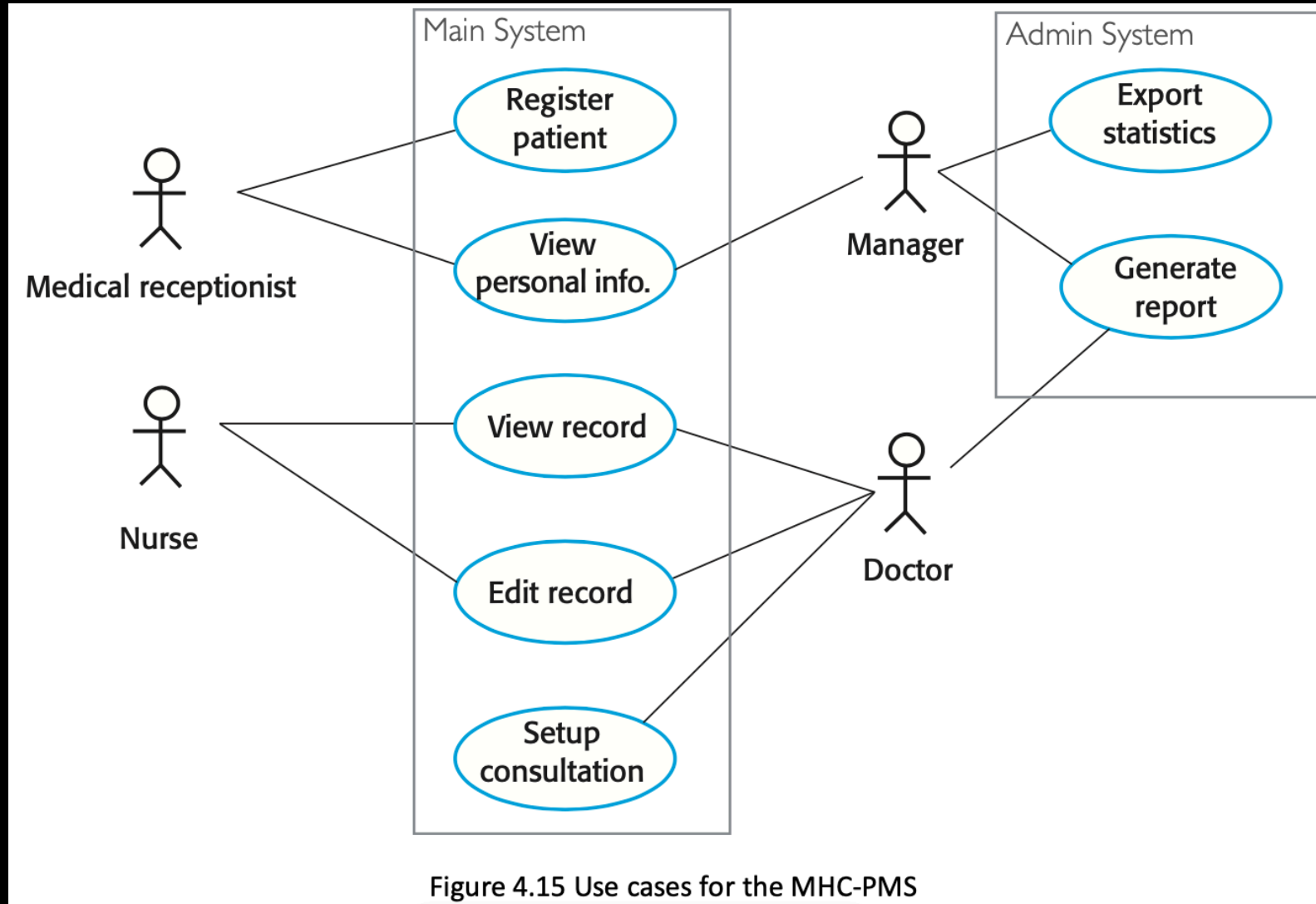


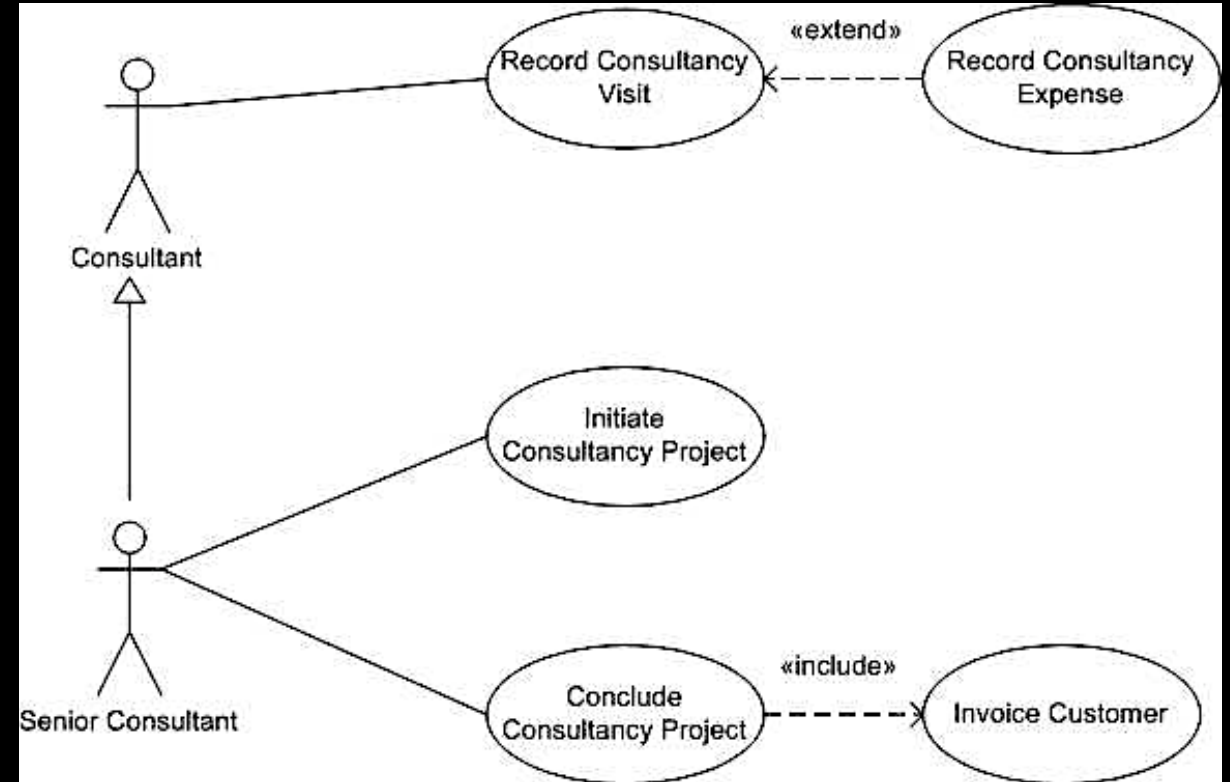
Figure 4.15 Use cases
for the Mentcare system

Use Case Diagrams



Use Case Diagrams

- A use case can be an extension of a task.
 - That should have done first.
- Some tasks may necessarily include other tasks.
 - That can be done separately.

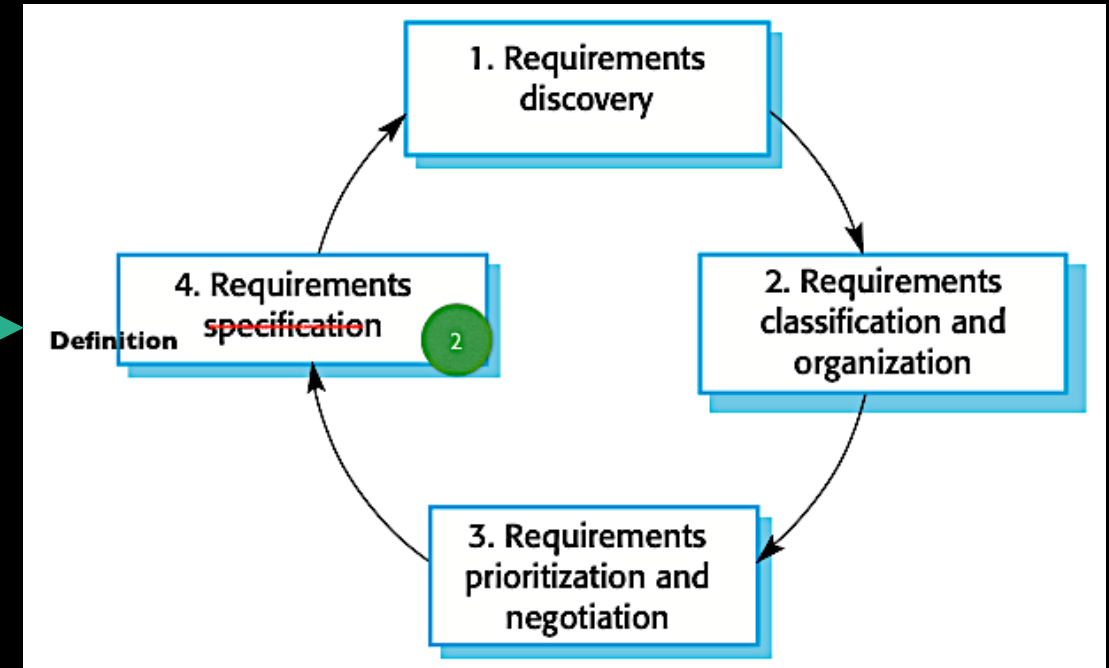
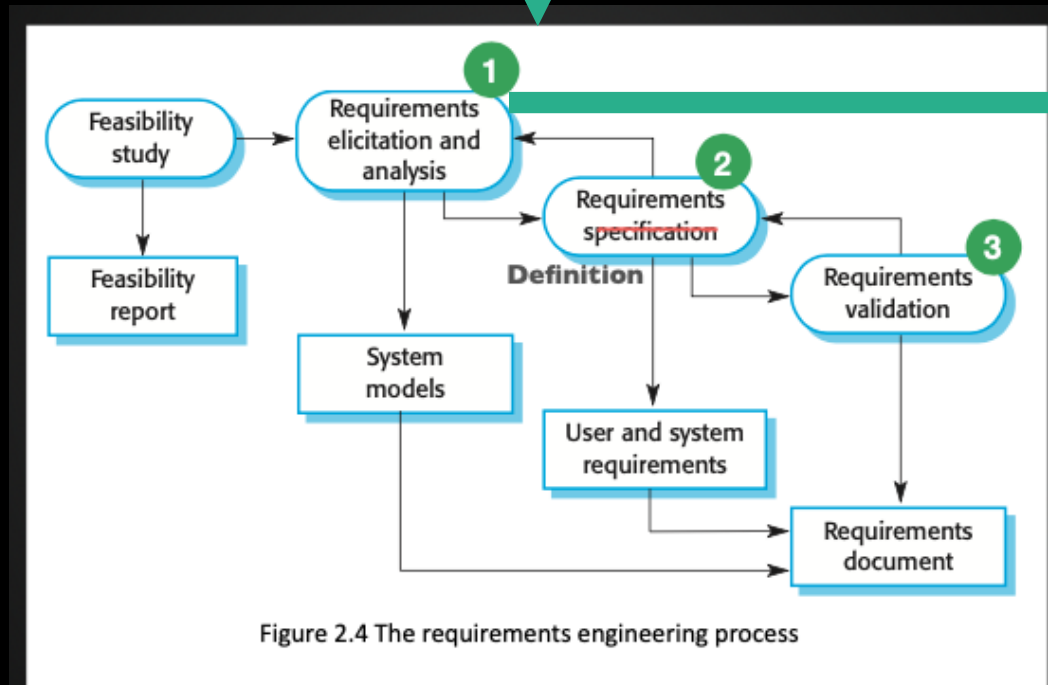
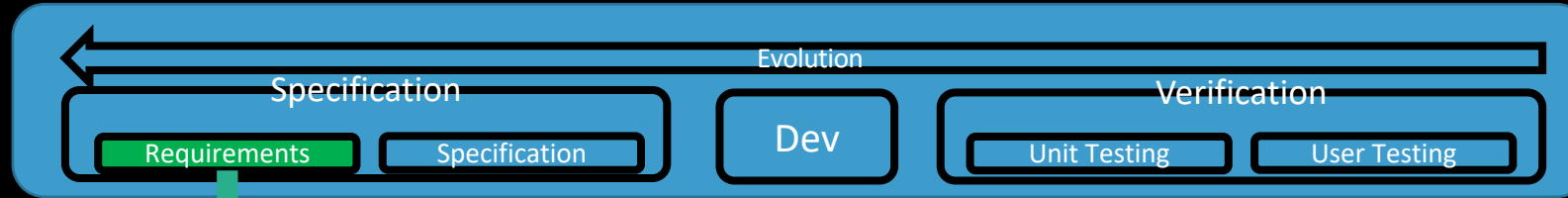


https://geeksw.com/tutorials/miscellaneous/uml/resources/making_use_case_diagram_using_generalization_in_uml.php



Keeping Track of SE Module

SE Process Management



- Primary, Secondary & Tertiary Stakeholders
 - <https://bizfluent.com/info-8353421-primary-secondary-tertiary-stakeholders.html>
- Personas
 - <https://www.usability.gov/how-to-and-tools/methods/personas.html>
- Use Case Diagrams
 - <http://www.agilemodeling.com/artifacts/useCaseDiagram.htm>

THANK
YOU