

CS 4530: Fundamentals of Software Engineering

Module 2.1: Requirements Analysis

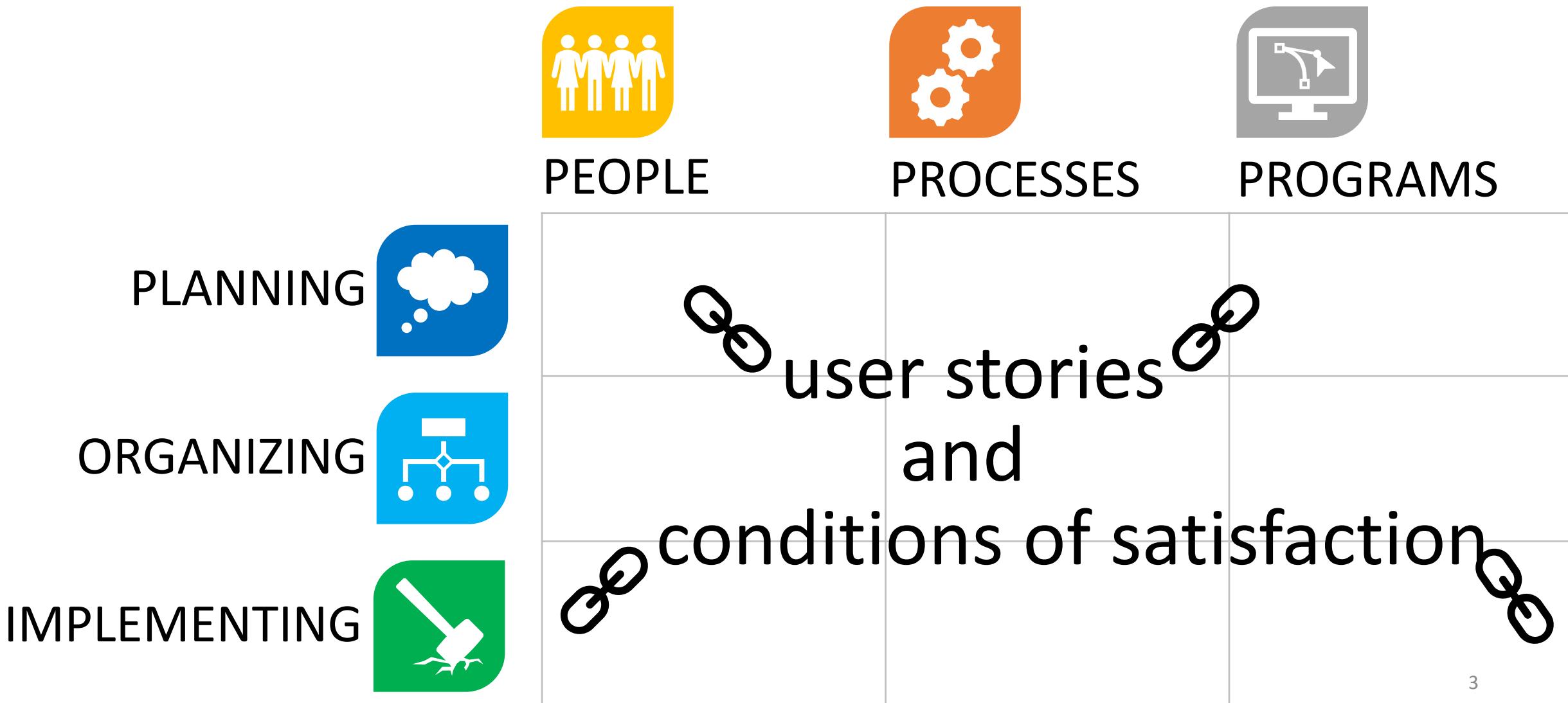
Adeel Bhutta, Rob Simmons, and Mitch Wand

Khoury College of Computer Sciences

Learning Goals for this Lesson

- At the end of this lesson, you should be prepared to
 - Explain the overall purposes of requirements analysis
 - Recall the three major dimensions of risk in requirements analysis
 - Explain the connection between requirements analysis and user stories

The big picture



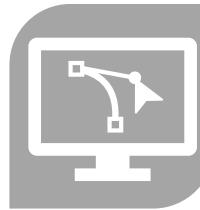
Requirements Analysis connects High-Level Planning To User Stories



PEOPLE



PROCESSES

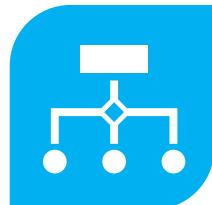


PROGRAMS

PLANNING



ORGANIZING



IMPLEMENTING



Requirements Analysis

User Stories

Testing Conditions of Satisfaction

Overall question: How to make sure we are building the right thing



How the customer explained it.



How the project leader understood it.



How the analyst designed it.



How the programmer wrote it.



What the customer really wanted.

Why is requirements analysis hard?



Problems of understanding

Do users know what they want?
Do users know what we don't know?
Do we know who are users even are?



Problems of scope

What are we building?
What non-functional quality attributes are included?



Problems of volatility

Changing requirements over time



How do we capture the requirements?

- There are many frameworks for capturing requirements.
- Often described as x -Driven Design (for some x)
- They differ in scope & details, but they have many features in common.

See also [edit]

- Behavior-driven development (BDD)
- Business process automation
- Business process management (BPM)
- Domain-driven design (DDD)
- Domain-specific modeling (DSM)
- Model-driven engineering (MDE)
- Service-oriented architecture (SOA)
- Service-oriented modeling Framework (SOMF)
- Workflow

Common Elements

- Meet with stakeholders
 - Develop a common language
 - Collect desired system behaviors that offer value
 - Document the desired behaviors
 - Iterate and refine!!

User stories are the least common denominator of most approaches

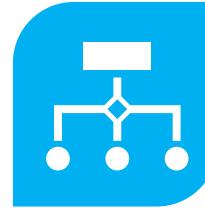


“Building the Right Thing” Necessarily involves Value Judgments

- Right for whom?
- Who benefits?
- Limitations of budget/time/personnel
- Competing/incompatible ideas about what's important
- Competing/incompatible sets of COSS

Requirements Gathering includes Prioritizing User Stories

- The last lecture covered **prioritizing conditions of satisfaction** within a user story.
- Now we're talking about **prioritizing user stories**. This is a complex *planning activity* that is constrained by resources (budget, time, personnel) and multiple (competing or incompatible) ideas about what's important.



Requirements Gathering Example: Value Sensitive Design

- Value Sensitive Design (VSD) is one framework (of many!) for doing requirements gathering
- VSD guides designers and engineers to pay special attention to **stakeholders** and **human values** when writing and prioritizing user stories
- Combines **empirical**, **value**, and **technical**

VSD Example – Informed Consent

Empirical Investigation:

- ❖ Understand what we mean by informed consent, encompasses:
 - Disclosure. Do we know the pros and cons of taking an action?
 - Comprehension. Do we understand the disclosures?
 - Voluntariness. Is there coercion or manipulation?
 - Agreement. Is there a clear opportunity to consent or not?
 - Competence. Are we capable to give consent?

Values Investigation:

- ❖ Who are the direct and indirect stakeholders?
- ❖ Do the stakeholders have conflicting values?
- ❖ How can we resolve them?

Technical Investigation:

- ❖ What are the technical mechanisms for implementing informed consent.
 - One way => cookie consent management system.
 - Websites use them to obtain and manage user permission for using cookies.

Review: Requirements analysis

- How do we make sure we are building the right thing?
- How do we learn from potential users before we start?
- Values: what even makes something the “right thing”
- Most forms of x -Driven Design could be a whole course on their own