CS 4530: Fundamentals of Software Engineering
Module 1.2: User Stories

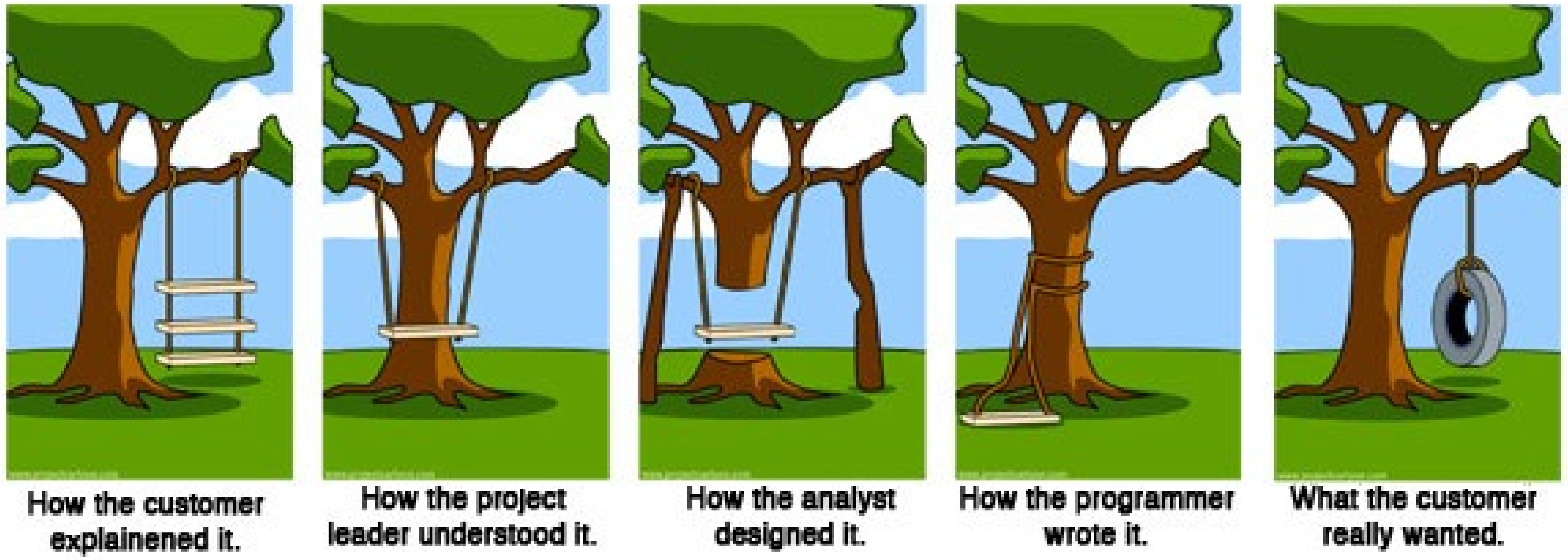
Adeel Bhutta, Rob Simmons, and Mitch Wand

Khoury College of Computer Sciences

Learning Goals for this Lesson

- At the end of this lesson, you should be able to
  - Explain the structure of a user story
  - Identify and fix user stories that don’t have the correct structure
  - Define the relationship between conditions of satisfaction and user stories, and the difference between essential, desired, and extension conditions of satisfaction
  - Explain the difference between functional and non-functional requirements, and give examples of each

User stories come from analyzing the user's requirements


How the customer explainened it.
How the project leader understood it.
How the analyst designed it.
How the programmer wrote it.
What the customer really wanted.

User stories come from analyzing the user's requirements

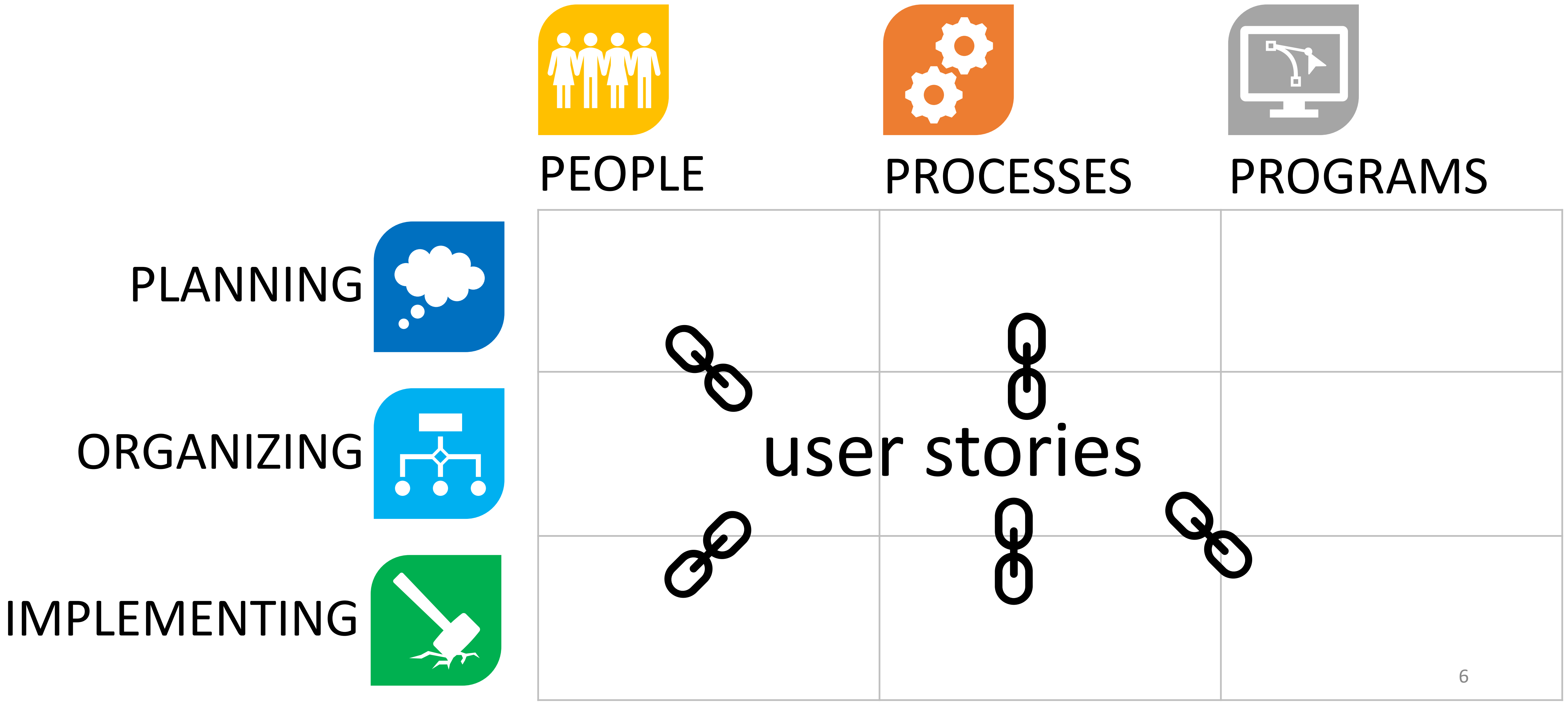
As a(n) active outdoor park-goer
I want a safe way to fly through the air under a tree
so that I can feel the wind in my hair

What are user stories?

- …a least-common-denominator approach documenting the requirements of users when **planning** projects
- …a tool for keeping large collaborative teams on the same page when **organizing** projects
- …formulaic statements of this form:

As a <role>
I want <capability>
so that I can <get some benefit>

User stories in software engineering



Components of a user story

As a(n) <u>active outdoor park-goer</u>
I want a <u>safe way to fly through the air under a tree</u>
so that I can <u>feel the wind in my hair</u>

As a <role>
I want <capability>
so that I can <get some benefit>

Roles: “who”

- Roles are positions or functions that people inhabit!
  - “As a web server…” is not a user story!
  - “As a human being…” is better, but that’s still not a role
  - “As a user…” is almost always a cop-out
- The person is not you!
  - You may want to build software to develop skills, or to make a codebase more maintainable. That’s good, but it doesn’t fit as a user story.

As a <role>
I want <capability>
so that I can <get some benefit>

Capabilities: “what”

- A capability is a specific thing I want to be able to *do*
  - Because we’re building software, this is usually an action we can provide by using software, otherwise let’s not build software
- A capability relates to a role
  - “As a teacher at Northeastern, I want to be able to access a laser cutter so I can finish an art project” is not a good user story.

As a <role>
I want <capability>
so that I can <get some benefit>

Capabilities: “what”

- A capability is not a product
  - “As a College Administrator, I want a web application that does <this> and <that> so that I can…” is not a user story!
  - Better: “I want to be able to see student’s grades” or “I want a way to track course enrollment over time”
  - This is easy to get wrong in practice!
    Sometimes you really want to build a tic-tac-toe game.
  - You’re supposed to ask, “do I even need to build this?”

As a <role>
I want <capability>
so that I can <get some benefit>

Benefit: “why?”

- Benefits are key for user stories actually focusing on what matters to the user.
- If a specific feature doesn’t relate to the benefit…
  - …maybe that feature isn’t worth building.
  - …maybe that feature is part of a different user story. (Maybe we should prioritize that different user story instead?)

As a <role>
I want <capability>
so that I can <get some benefit>

Properties of a good user story

- short: fits on a 3x5 card
- may have prerequisites
- has *conditions of satisfaction* that expand on the details
- satisfies the INVEST+E criteria (more on this later)

Good Examples of User Stories

- As a College Administrator, I want to keep track of students, the courses they have taken, and the grades they received in those courses, so that I can advise them on their studies.
- As a driver, I want to be able to report potholes to the city so that the town can more quickly act to keep me safe.
- As a pedestrian, I want to be able to report potholes to the city so that drivers stop dangerously swerving onto the sidewalk when I'm walking.
- As a card game enthusiast, I want to be able to play blackjack online so that I can…

Conditions of Satisfaction fill in details of the desired behavior

- Each condition of satisfaction
  - Describes a testable behavior, from the user's point of view
  - Must have a priority
  - Should be numbered within its user story
- There may be *multiple* options for filling in the details with conditions of satisfaction: this is a design problem!

Conditions of Satisfaction Have Priorities

- **Essential (E)** means the project is useless without it.
- **Desirable (D)** means the project is less usable without it, but it is still usable.
- **Extension (X)** describes a CoS that may not be achievable within the scope of the project. These might be things you'd want "in the next version".

Worked Example: Pothole reporting system

A town is designing a system where citizens can a report potholes and the town can monitor progress on repairing them.

User Story #1

- As a car commuter, I want to be able to report potholes to the city so that the town can more quickly act to keep me safe.

Conditions of Satisfaction

- As a car commuter, I want to be able to report potholes to the city so that the town can more quickly act to keep me safe.
  - 1.1 I should be able to report the location of a pothole to the system (E)
  - 1.2 I should be able to see whether the pothole I report has been repaired (E)
  - 1.3 I should be able to see whether others have reported potholes near me (D)
  - 1.4 I should be able to see an estimated time when the pothole should be repaired (X)

User Story #2

- As a pothole-repair-truck driver, I want the system to display the potholes I should be working on today.

Conditions of Satisfaction

- As a pothole-repair-truck driver, I want the system to display the potholes I should be working on today.
  - 2.1 I should be able to see my list of potholes for today (E)
  - 2.2 I should be able to report that I repaired a given pothole (E)
  - 2.3 I should be able to report that I was unable to repair a given pothole, and to supply a reason (E)
  - 2.4 My daily list of potholes should be listed in an order that cuts down the time I spend driving from job to job (D)

User Story #3

- As a maintenance supervisor, I want to be able to control the order in which potholes are repaired

Conditions of Satisfaction

- As a maintenance supervisor, I want to be able to control the order in which potholes are repaired
    - 3.1 I should be able to give a higher priority to potholes on a particular street (E)
    - 3.2 I should be able to give a higher priority potholes in a particular neighborhood (E)
    - 3.3 Anyone not a maintenance supervisor **should not** be able to change pothole priority (E)
    - 3.4 I should be able to see on a map where there are a lot of potholes (D)
    - 3.5 I should be able to see on a map which potholes that have been reported multiple times (D)

Writing User Stories: INVEST + E

- Independent
- Negotiable
- Valuable (has value to client)
- Estimable (able to estimate development effort)
- Small
- Testable

As a <role> I want <capability> so that I can <get some benefit>

- Ethical (connects with our & users' values)

User stories aren’t everything!

- User stories roughly describe the **functional requirements**
- The **non-functional requirements** are other properties that are also important to users and to other stakeholders
  - How quickly can a transcript be retrieval? (Performance)
  - How many student transcripts can our system store? (Scalability)
  - How long did I spend on the phone with support to set up the software? (Usability)
  - After my system is setup, is the access controlled at all? (Security)
  - Are these any times when I can’t use this system? (Availability)
  - How expensive will it be to adapt the system to new requirements? (Maintainability)

Other non-functional requirements

- Availability
- Capacity
- Efficiency
- Extensibility
- Maintainability
- Performance
- Privacy
- Response Time
- Response Time
- Scalability
- Security
- Supportability
- Testability
- Usability

Minimum Viable Product

- Once you decide on a set of user stories, the Essential (E) conditions of satisfaction for those user stories constitute the minimum viable product (MVP)
- A user story is "implemented " when all its essential COSs are implemented.
- Caution: when proposing a project, don't make your MVP too hard to complete (but don't make it too easy, either)

Review

- At the end of this lesson, you should be able to
  - Explain the structure of a user story
  - Identify and fix user stories that don’t have the correct structure
  - Define the relationship between conditions of satisfaction and user stories, and the difference between essential, desired, and extension conditions of satisfaction
  - Identify functional and non-functional requirements, and give examples of each