Software Engineering

CS305, Autumn 2020

Week 3

Class Progress…

- Last week:
  - Git overview (cloning, commit, tagging, remote repos)
  - Requirements Engineering (RE)
    - What is it?
    - Why important?
    - What is the result?
    - What are the desired characteristics of requirements?
    - Where do the requirements come from?
    - What are the different types of requirements?

Class Progress…

- This class
  - What are the problems in requirement gathering?
  - How are requirements typically captured?
  - What are the next steps are gathering requirements?
- Recall (how to get it right?)
  - Requirements Engineering involves different activities:
    - Elicit, Analyze, Specify, Validate, Manage - *Iterate*

Elicit

- Gathering requirements from various sources:
  - Stakeholders
  - Documents
    - Manuals, books, papers etc.
  - App domain
- Not a straightforward task
  - Domain knowledge is
    - distributed,
    - rarely in written form,
    - has conflicts (due to multiple sources),
    - is noisy (due to possible behavioral change in actors when you observe them),
    - prone to biases (people may try to influence you and omit information)
  - Disconnect between perception and practice
    - Customer's perspective of a simple 3 steps might involve N steps in practice

Elicit - techniques

- Background Reading
  - Used when one is not familiar with the org. and used before interviewing
  - Sources: e.g. company annual reports, job descriptions
  - Cons: time consuming, may contain irrelevant details, out-of-sync
- Interviewing
  - Pros: can uncover a rich set of info through follow-up probing
  - Cons: requires specialized skills to interview people
- Collecting facts and figures through hard data and samples
  - Which data to collect? What is a sample?
  - Sources: financial reports

Elicit – techniques contd..

- Surveys
  - Pros: Quickly collect info from large population, remote administration possible
  - Cons: may miss opportunities to collect relevant information
- Meetings
  - Summarization of findings
- Collaborative, Social, and Cognitive Techniques
  - E.g. Brainstorming, collecting information about participants by observing them in their environment, finding problem solving methods of participants

Elicit – techniques summary

- Traditional Techniques
  - Surveys,
  - Meetings,
  - Hard data and samples,
  - Interviewing,
  - Background reading

Modeling requirements

- Purpose: structured organization of requirements gathered for analysis and refinement
- Several ways depending upon focus and objectives / depends on what and how to model.
  - Organizational / Enterprise modeling e.g. goal modeling
  - Behavioral / Information modeling e.g. sequence, class, structural diagrams
  - Modeling quality aspects e.g. task models

  They are all complementary. Can have a mix of one or more.

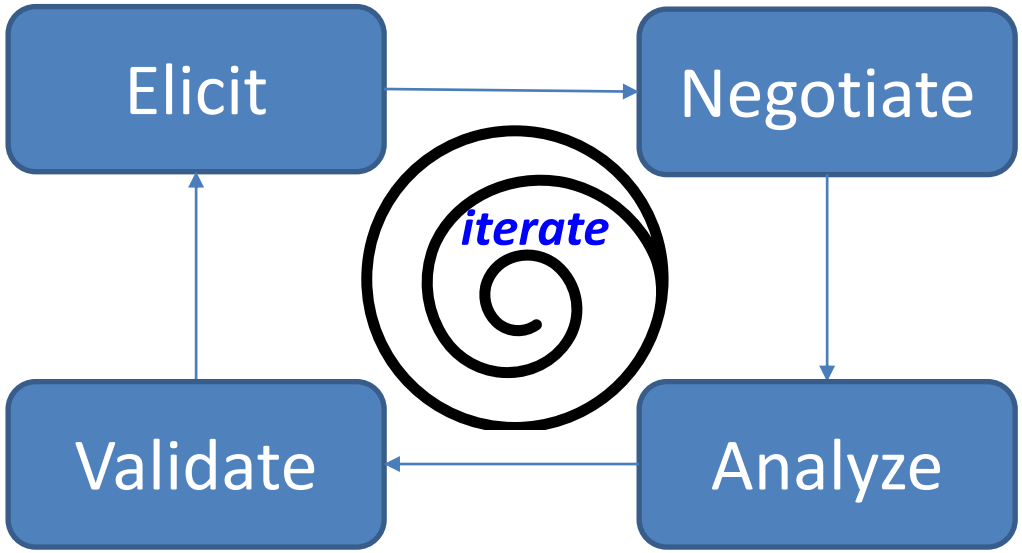
  - Goal Modeling is extremely popular,
    - a natural way – start with goals and continuously refine them
  - Natural language for modeling
  - Unified Modeling Language

Analysis

- Two tasks primarily:
  - Verification
    - Developers check for SRS conformation (correctness, performance, completeness, pertinence etc.)
  - Validation
    - Check if customers' needs are satisfied
- Outcomes:
  - Feasibility study – checks for time, budget, meeting org. objectives, system integration requirements etc.
  - Risks identified and addressed
  - Prioritized list of requirements (mandatory, nice-to-have, superfluous)

RE Process

- Iterate over the 4 activities



*In practice, you always end up doing a bit of **design** in RE and **vice-versa***

RE and Design

- When you refine:
  - System design may start emerging
  - Discover how system inter-operates with other systems
    - This generates design requirements
    - System component interaction exposes design alternatives, procedures, data formats, etc.

Requirements Modeling Techniques Examples

Form-based spec: example

Insulin Pump/Control Software/SRS/3.3.2

Function Compute insulin dose: Safe sugar level

Description Computes the dose of insulin to be delivered when the current measured sugar level is in the safe zone between 3 and 7 units.

Inputs Current sugar reading (r2), the previous two readings (r0 and r1)

Source Current sugar reading from sensor. Other readings from memory.

Outputs CompDose Š the dose in insulin to be delivered

Destination Main control loop

Action: CompDose is zero if the sugar level is stable or falling or if the level is increasing but the rate of increase is decreasing. If the level is increasing and the rate of increase is increasing, then CompDose is computed by dividing the difference between the current sugar level and the previous level by 4 and rounding the result. If the result, is rounded to zero then CompDose is set to the minimum dose that can be delivered.

Requires Two previous readings so that the rate of change of sugar level can be computed.

Pre-condition The insulin reservoir contains at least the maximum allowed single dose of insulin..

Post-condition r0 is replaced by r1 then r1 is replaced by r2

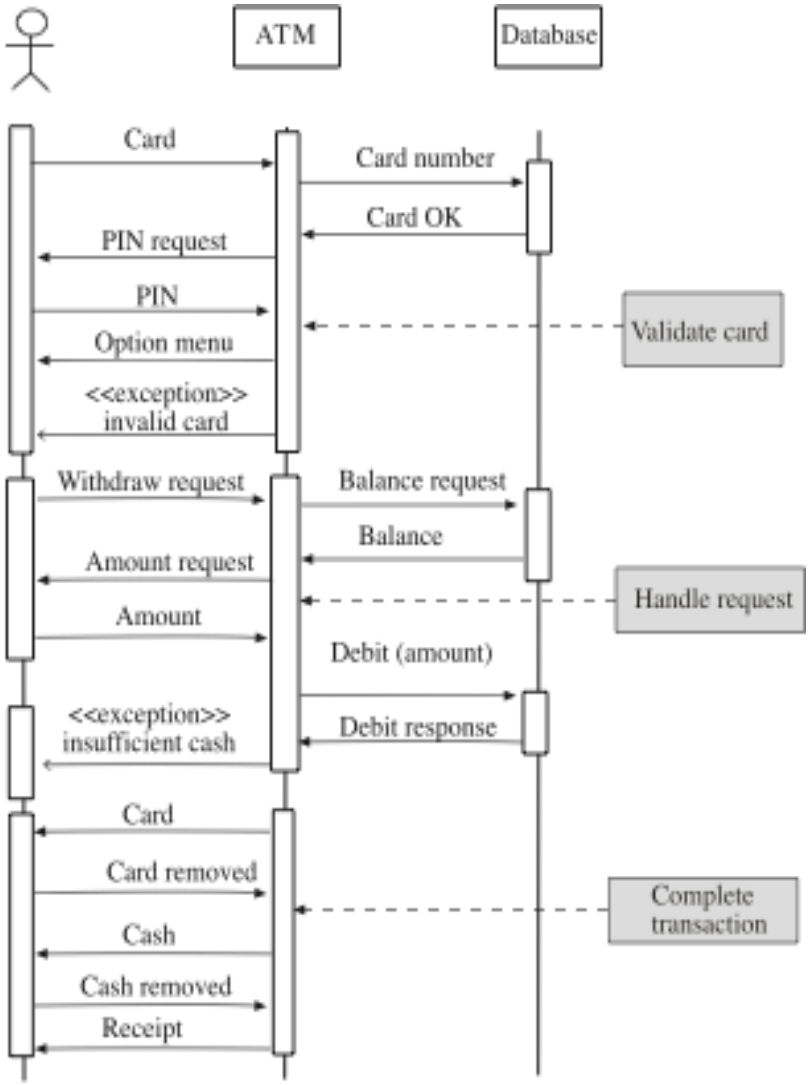
Side-effects None

source: Software Engineering, Ian Sommerville

Graphical Models

- Easier to consume
- Intuitive
- E.g.
  - Sequence diagrams
  - Finite state machines
  - Data-flow diagrams

Sequence diagram – ATM withdrawal



slide courtesy: Alex Orso

Scenarios

- Effective requirement elicitation technique
- Captures real-world use cases of the system
- Desirable features:
  - Description of initial condition
  - Description of normal flow of events
  - Description of failure scenarios (what can go wrong)
  - Information about other activities happening simultaneously
  - Description of end state

Scenarios - Example

- LIBSYS – controlled electronic access to copyright material from a group of university libraries

Initial Assumption: The user has logged on to the LIBSYS system and has located the article to be accessed

Normal: The user selects the article to be copied. The journal (in which the article is a part) prompts the user to provide subscriber information OR pay for the article using credit card or organizational account number

The user is then asked to fill in a copyright form that captures the transaction details. The form is then submitted to the LIBSYS system

If copyright form is OK (upon checking), the PDF version of the article is downloaded to the working area of the user's computer and the user is informed that the article is available for printing. The user is asked to select a printer (if the article is available for print-only) and after obtaining a confirmation from the user that the printing is complete, the article is deleted.

Scenarios – Example (contd..)

What can go wrong: Incorrect filling of copyright form, should prompt the user to refill the form again with suitable corrections. If the user fills the form incorrectly in the second attempt, request for access to the article is rejected.

When the system rejects the payment details, the article request is rejected.

When the download fails, retry until succeeds or until the user terminates the session.

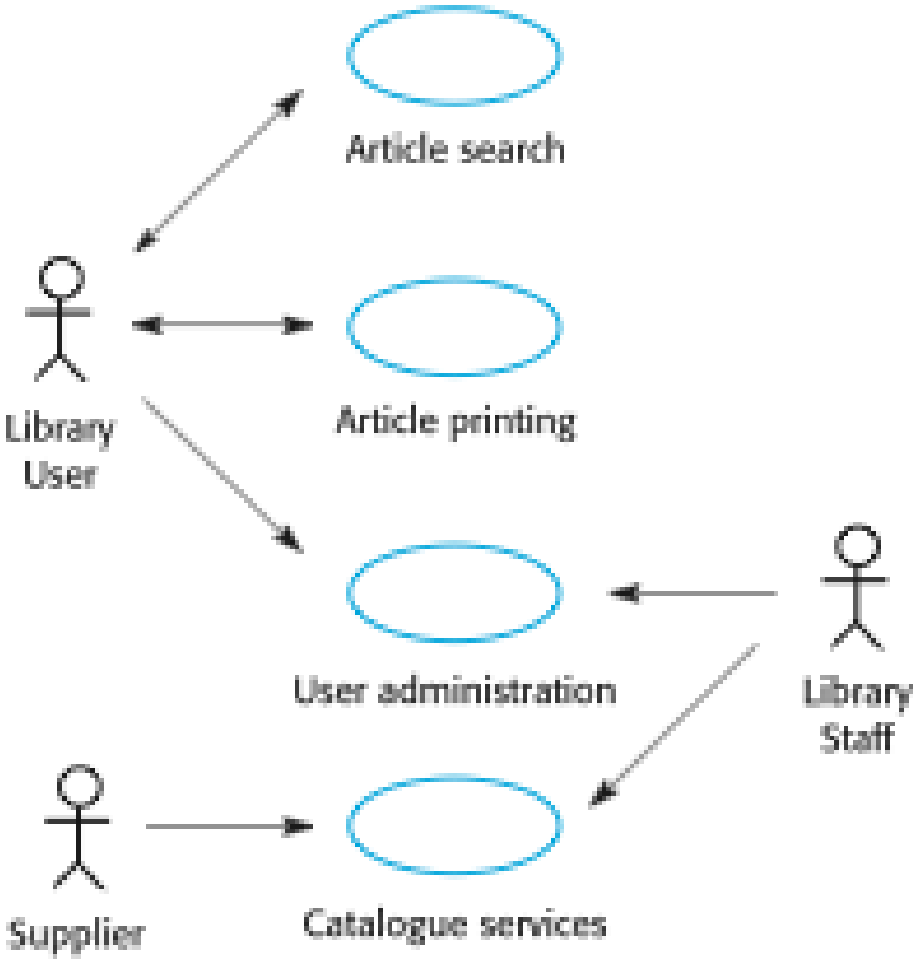
If the article could not be printed (When tagged as print-only), the article is deleted and the cost refunded to user. If the article is not tagged as print-only, it is retained in LIBSYS workspace.

Other activities: simultaneous download of other articles

Use cases

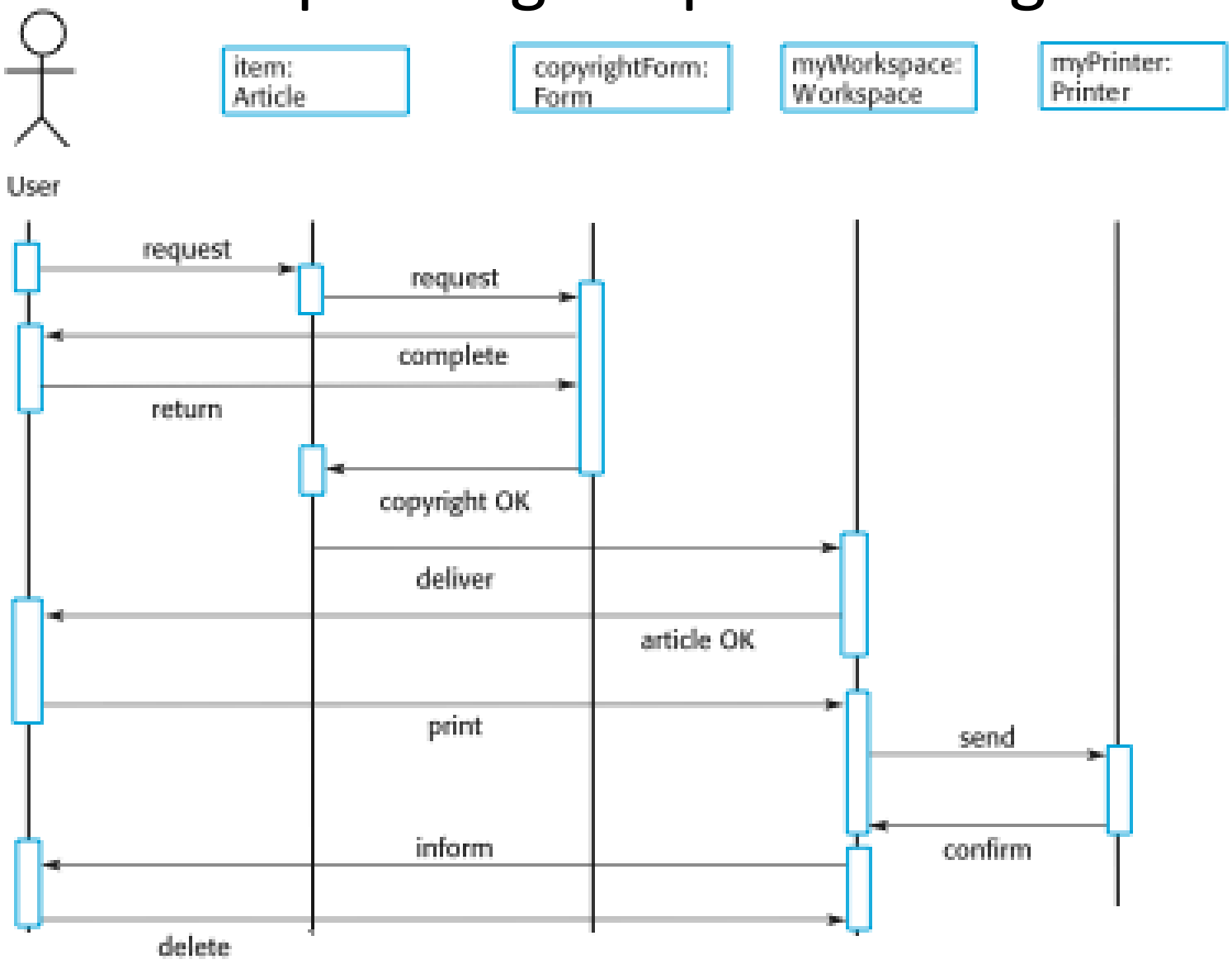
- UML's scenario-based technique
  - actors and interactions
- Should describe all possible interactions with the system
- Sequence diagrams may be used to add details to use-cases

LIBSYS use cases



slide courtesy: Alex Orso

Article printing: sequence diagram



slide courtesy: Alex Orso

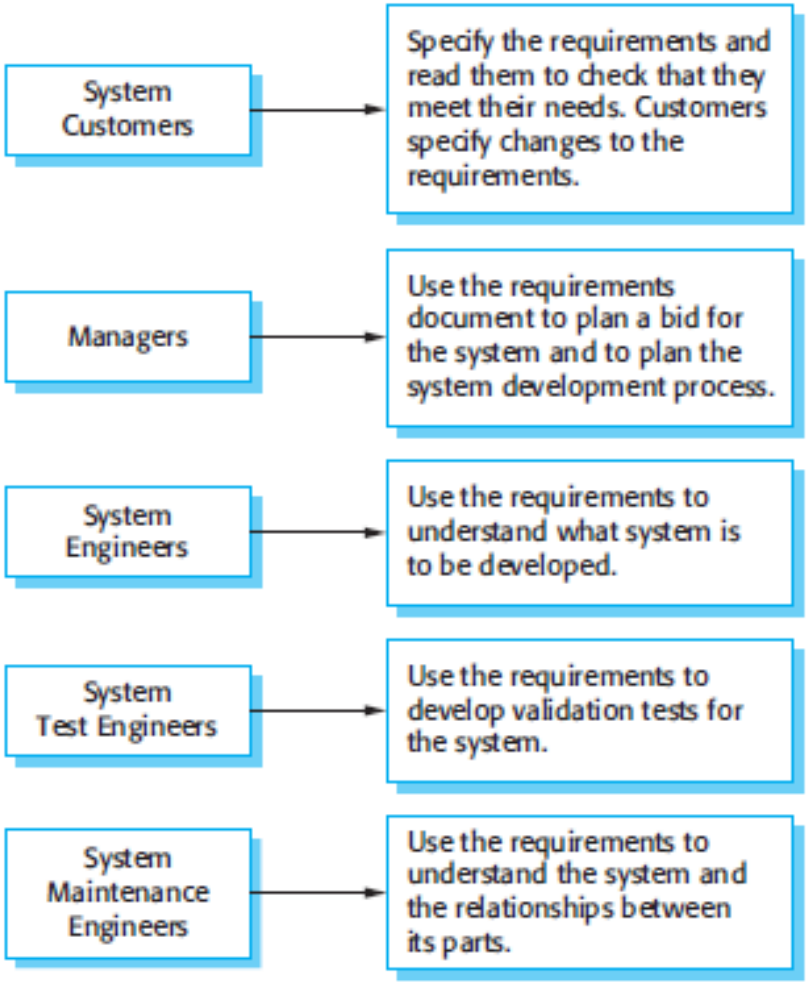
Traceability

- Is the link between requirements, their sources, and the system design
- Source traceability
  - Requirement -> stakeholders who proposed these requirements
- Requirements traceability
  - Requirement -> dependent requirement(s)
- Design traceability
  - Requirement -> design

Change Management

- Accommodating changing requirements
- Main stages:
  - Problem analysis – discuss what is the problem with a requirement and propose change
  - Change analysis and costing – assess effects of change on other requirements
  - Change implementation - modify requirements document (and other docs) to reflect change

Users of a Requirements Document



pic source: Software Engineering, Ian Sommerville

SRS Summary

- Way to communicate requirements to others
- Different projects require different SRSs depending upon the context e.g. small vs. large teams