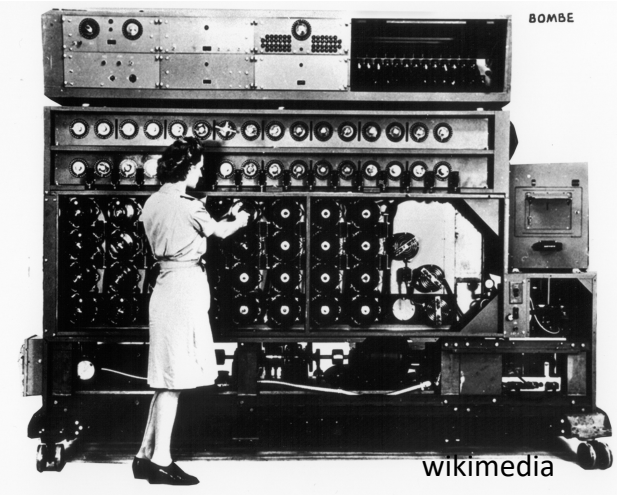
Lecture 4
Requirements Discovery: Scenarios

Com S/SE 409/509

Robyn Lutz

rlutz@iastate.edu

Robyn's Office Hours: **Tues & Thurs, 9:30-10:45**
Wandi's Office Hours: **Mon 10 am**. & **Wed 7 pm**
Olukorede's Office Hours--**Wed 10**

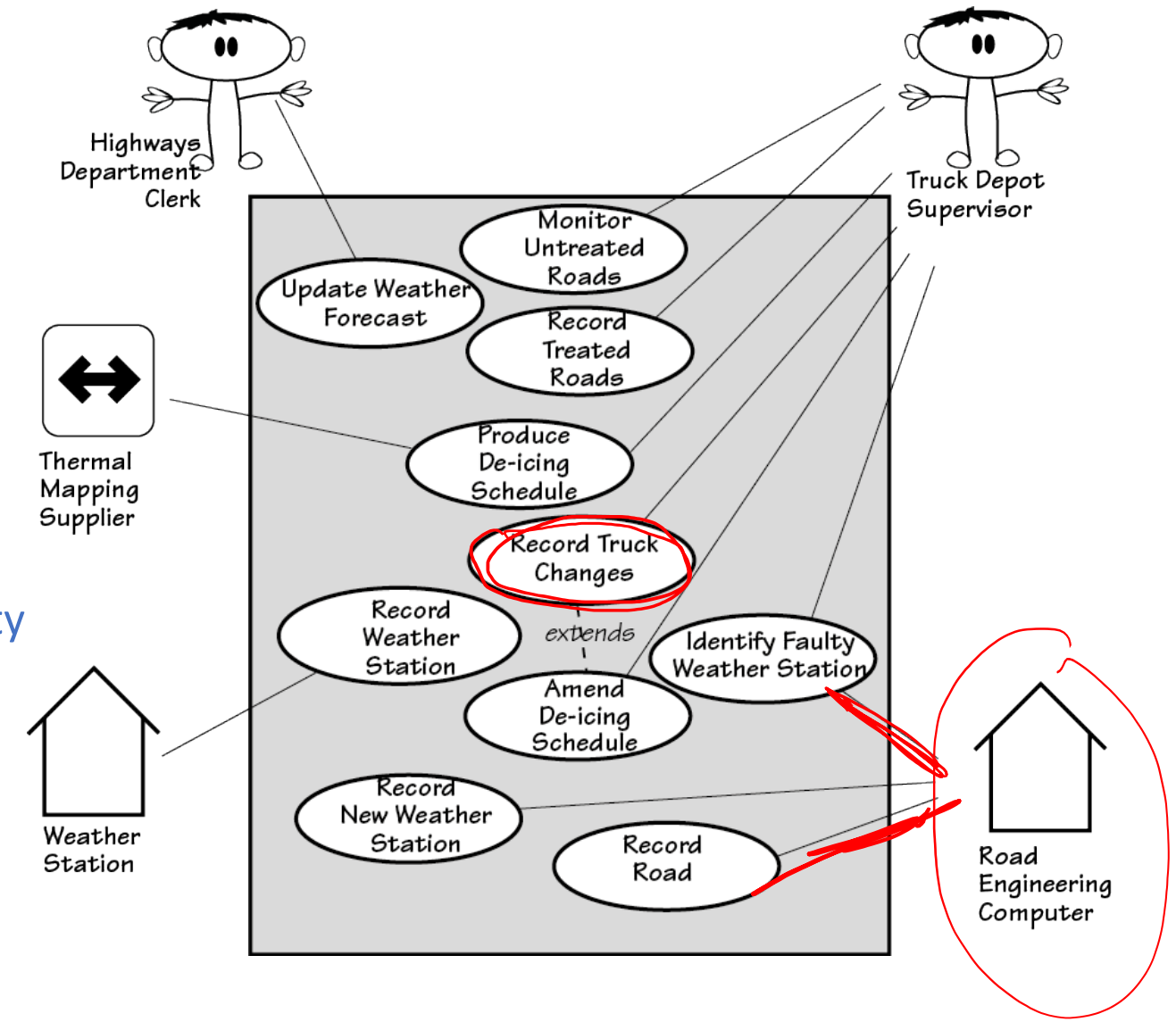


wikimedia

Homework 1 due next Thurs, 9/3

Figure 4.12

The product use case diagram for the IceBreaker product, showing the product use cases, the actors involved in each product use case, and the product's boundary. The different notation used for the actors indicates the way they interact with the product. (These distinctions are explained in Chapter 8, where we look at starting the product.)



From Lecture 3: IceBreaker's **product use case diagram**

Use cases are units of functionality the product needs

Actors are adjacent systems that Interact with the product

(Icons for adjacent systems: pp. 190-194)

Product Use Cases → Scenarios → Functional Requirements

- Chapter 6: Product Use Case → Scenario
- What is a scenario?
  - It "tells the story" of a use case
  - Scenarios break down each product use case into a series of stakeholder-recognizable **steps** (Glossary, p. 515)
  - Build at least one scenario for each product use case
- Advantage:
  - Fast way to find/record/revise functional requirements in meetings with stakeholders

Scenario template (pp. 198-199) for product use case.

1. **Product use case name**: passenger checks into airline flight
2. **Trigger:** passenger activates the machine
3. Preconditions:
4. Interested stakeholders:
5. **Actor:** passenger
6. **Steps:*** (1) software asks for passenger's name . . . (9) software prints baggage tag
7. Outcome:

* 3-10 steps recommended

Beyond normal scenarios

paths: are there alternative scenarios?

choices, so subsequent steps branch based on that choice

ow OR save cart for later?

there "exception"al/misuse/negative/what-if scenarios?

deviations from normal

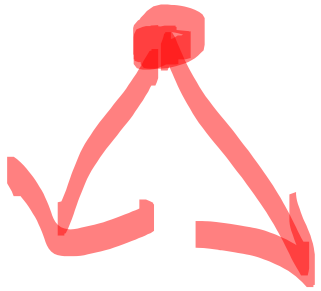
forgets password; malicious user enters wrong data; user is

ep, ask "what can go wrong"?

carried away thinking of too many of these too early

systems, use Jackson's principle of commensurate care:

care = probability of an exception × seriousness of damage if it occurs



Beyond scenarios

- Storyboard: shows the steps of a scenario as a sequence of pictures
  - Good for User Interfaces, displays, dashboards
- Limitations of scenarios [Wiegers & Beatty]:
  - Partial coverage of possible system behaviors
  - Thus, misses some functional requirements
  - Too-early use risks over-specification & unnecessary constraints on order, parallelization of steps
  - Doesn't capture the "why" of interactions (rationales)
  - Doesn't capture non-functional requirements (quality attributes)
- Check your understanding with the worked-out example on pp. 196-199
  - the software to be developed will automate checking a passenger in for a flight