For personal use only,
please do not
distribute on
non-UBC domains.

10b - Modelling: Scenarios

CPSC 310

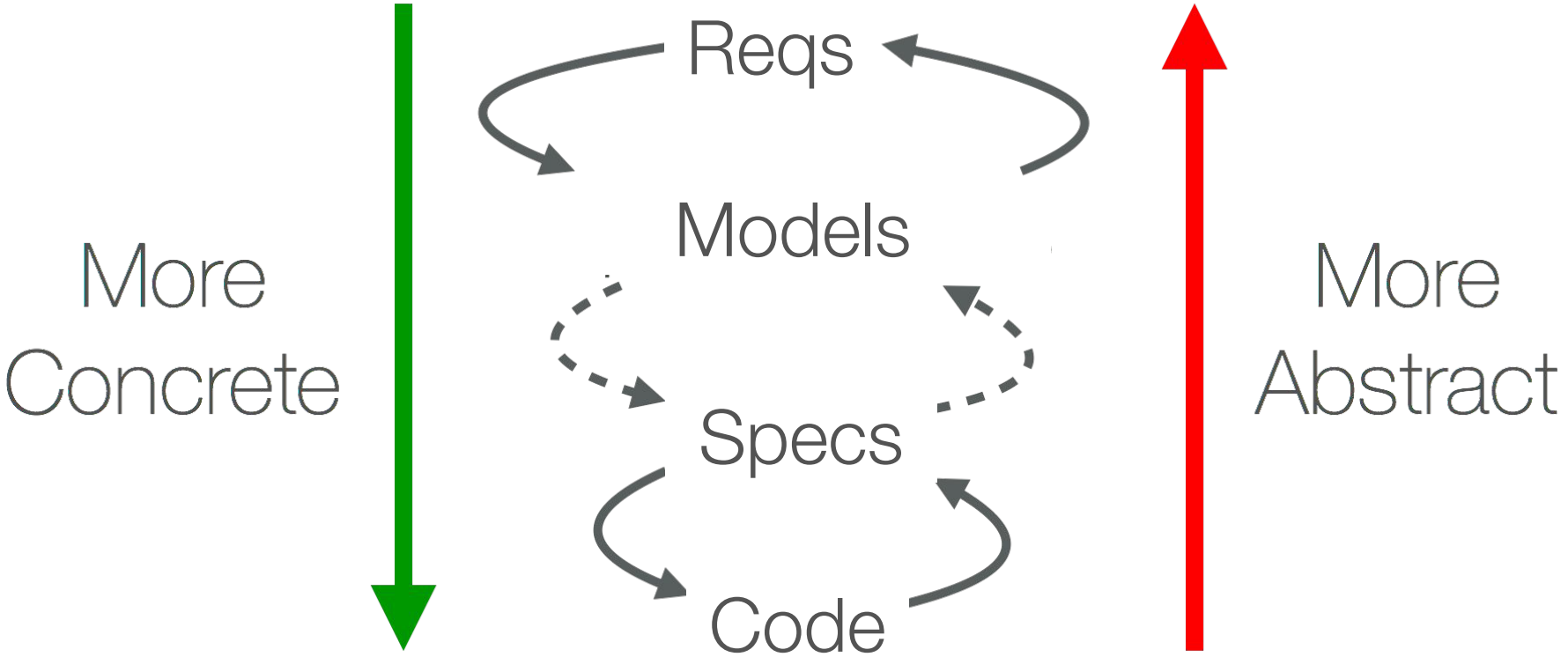
Nick Bradley & Kyle Chin

Learning goals

By the end of this lecture, you should be able to:

- Create requirement scenarios in terms of system state
- Identify operations that change system state
- Reason about valid and invalid (or unspecified) system states
- [Next lecture] Revisit how to model state in terms of data & relations

Moving Between Representations



Scenario walkthroughs

Scenarios are concrete examples of the requirements in action.

Each one describes what the system should do in a specific situation:

- **Given**: the state of the system before the action
- **When**: an action an actor (e.g., a user) performs
- **Then**: the state of the system after the action

Writing scenarios helps reveal ambiguous or missing requirements and suggest the data we will need eventually need to model in our system.

Each section has a fixed number of seats. Once it's full, students can join the section's waitlist, which holds at most 50 students. A student can't join the waitlist for a section they're already enrolled in, and can't join the same waitlist twice. Students can leave the waitlist whenever they like, and can check their position on it at any time.

When a seat frees up, the first student in line is offered it and has 24 hours to accept. If they accept, they're enrolled and come off the waitlist. If they decline, or let the offer expire, they come off the waitlist and the seat goes to the next student in line.

Students' names and email addresses live in the Student Information System, not with us. The same student is often waiting for several sections at once.

Scenario #1: Joining waitlist

Given:

- Section is full
- Waitlist is not full
- Student is not enrolled
- Student is not on the waitlist

When:

- Student joins the waitlist

Then:

- Student is added at the end of waitlist

Scenarios surface hidden assumptions

- Requirements never state every rule
- Writing scenarios exposes assumptions hidden in the text
- Once an assumption is surfaced, you can make a decision and document it
- Fill gaps with reasonable decisions where you can
  - Go back to the client to ask/verify

Tips for writing scenarios

- **Stay in the problem domain.** Describe what happens in client terms, not how it's built.
- **Use words from the requirements.** Say "waitlist," "offer," "enrolled," not your own synonyms.
- **Be consistent.** List the same state items in the same order, so the differences between scenarios stand out.
- **Cover the space.** For each action, work through every valid starting state. Being systematic helps find edge cases.
- **Keep it short.** *Declare* what is true, not the steps to get there.

Conflicting Requirements

- Occur when more than one *Then* state is valid for the same *Given/When* pair
- Example:
  - Given: A seat has freed up and is offered to the first student on the waitlist (offer pending)
  - When: A different student tries to enroll
  - Then (option 1): The new student is enrolled, because section isn't full
  - Then (option 2): The new student can't enroll, because the seat is being held for the student who was offered it
- To resolve: decide which requirement takes priority, and confirm with the client if the choice matters to them

Scenario #8: User: Student; Action: Accept Offer

A. Given:

1. Student has offer
2. Offer is not expired
3. Student is not enrolled
4. Section is not full

B. When:

Student accepts offer

C. Then:

Student is enrolled

Student A accepts offer, then immediately unenrolls. Student B receives offer. Student A accepts offer again.