An OO Design Exercise, cont: UI Design and Components

13 Mar 2009
CMPT166
Dr. Sean Ho
Trinity Western University

TRINITY WESTERN UNIVERSITY

Steps to OO design: wADes

- (Prereq: understand client requirements)
- System behaviour
  - Use-case scenarios
  - User interface mockups
- Components
  - Self-contained blocks with narrow interactions
- From components to classes
  - Attributes, methods

Example (midterm q. #10)

- Problem statement:

  Design a student enrolment database like we have at TWU

(1) Actors and actions

- Who are the actors? Who will interface with us?
  - *Student, Alumni, Other students/public, Registrar, Database, Advisor, Instructor*
- What are the actions? Scenarios of use?
  - *ask for GPA, change addr (stu->db)*
  - *add class, drop class (stu->db)*
  - *meet w/advisor (stu->advisor, advisor->db)*
  - *submit change advisor form (stu->registrar)*
  - *change advisor (registrar->db, advisor)*

Specifying a use case

- Each use case should have:
  - Short name
  - Goal: what does it achieve for its actors?
  - Names of actor(s) involved
  - Pre/post-conditions?
  - Basic flow: break down into steps (pseudocode!)
  - Alternate flows: what if user inputs something different from the usual?

Use case ex: advising meeting

- Name: Advising Meeting
- Actors: *...Student, Advisor, DB*
- Goal(s):
  - *...Student: choose right classes*
  - *Advisor: make sure sched works: appropriate credit load*
  - *DB: approve courses, check waitlist, security....*
- Pre-conditions: *...student registers for classes, advising window is open, ...*

Advising meeting: basic flow

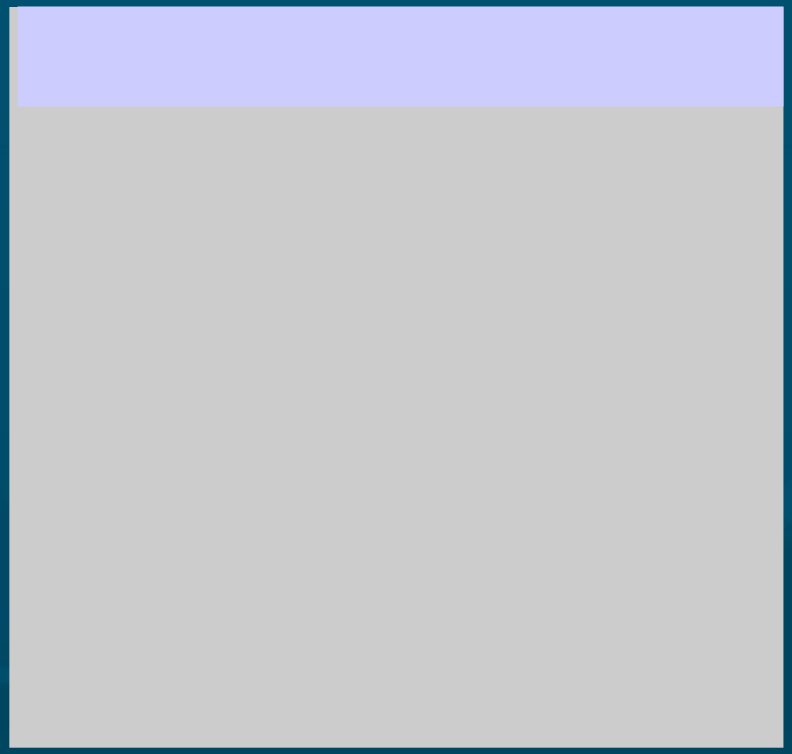
- Basic flow:
  - *...Student makes appt w/advisor*
  - *student and advisor talk about sched*
    - *advisor offers guidance: classes, goals*
  - *Advisor approves on DB*
  -

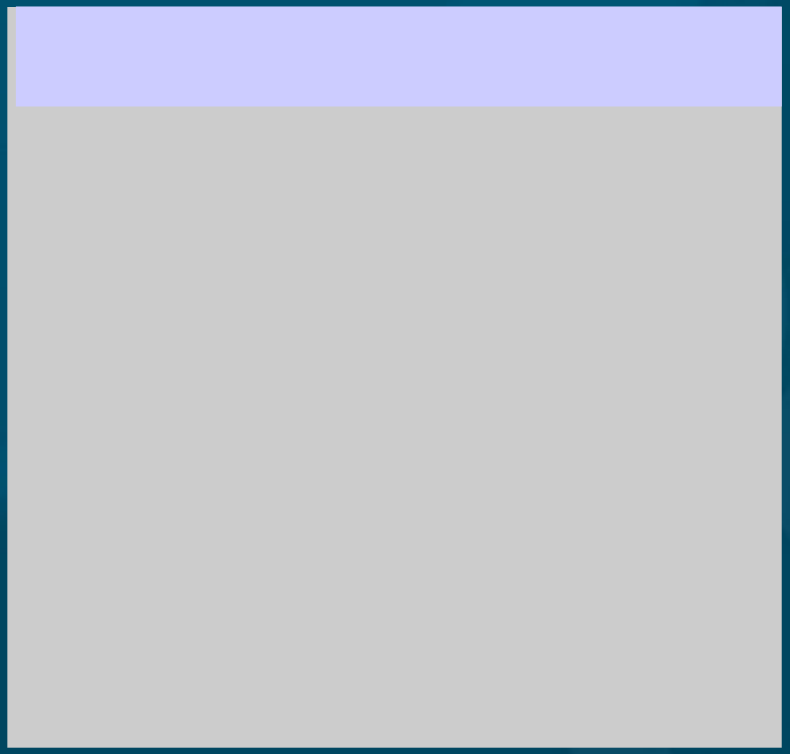
Advising mtg: alternate flows

- What might not go according to the basic flow?
  - *...fire, computer going down, etc.*
  - *advisor too busy: can't make appt*
  - *student picks course that doesn't exist*
    - *or time conflict, or doesn't meet prereqs*
  - *advisor doesn't check off /approve sched*

(1) UI mockup

- For each use-case (action), describe/mockup what the user interface will be like:
  - Text Q&A? Windows? Interactivity?

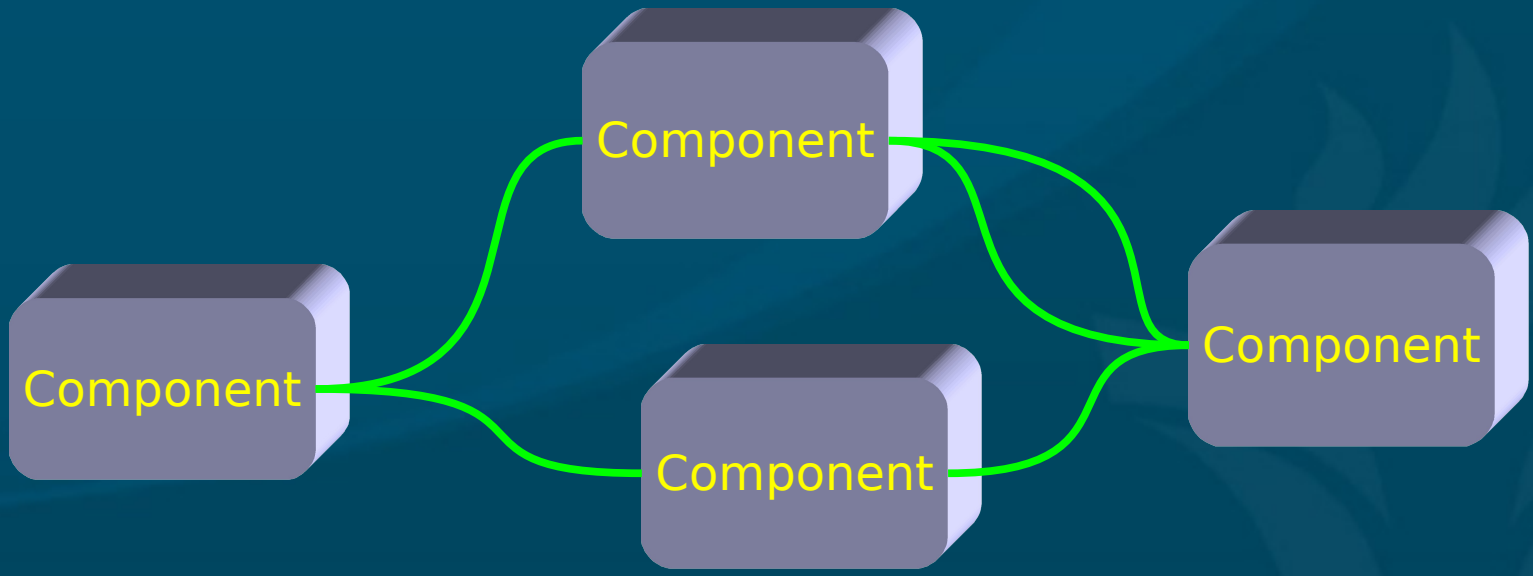




UI mockup: advising meeting

(2) Component design

- This is often the hardest part!
- Components need not be classes
- Thinly coupled: describe all interfaces between components



Component: Advisee

- Name: *...Advisee component*
- Description: *Student info from DB*
- Interface to *(Notes)*:
  - *...*
- Interface to *(main DB)*:
  - *...get name, ID#, major*

Component: ....

- Name: *...*
- Description: *...*
- Interface to *(component)*:
  - *...*
- Interface to *(component)*:
  - *...*

(3) From components to classes

- Each component may need several classes to implement it
- Component: ...
  - Class: ...
    - Attributes: ...
    - Methods: ...
  - Class: ...
    - Attributes: ...
    - Methods: ...