4156

remind to bring laptop

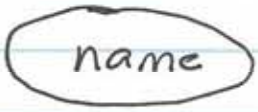
9/29/15

UML = Unified Modeling Language

standard graphical (visual) notation used in SW industry to depict
- use cases
- design - structural & behavioral
- some aspects of implementation & deployment

simplest UML is for use cases

oval

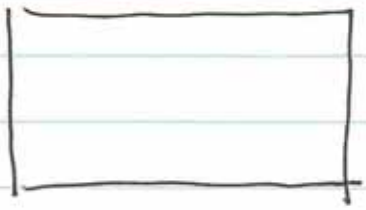


active voice present tense verb

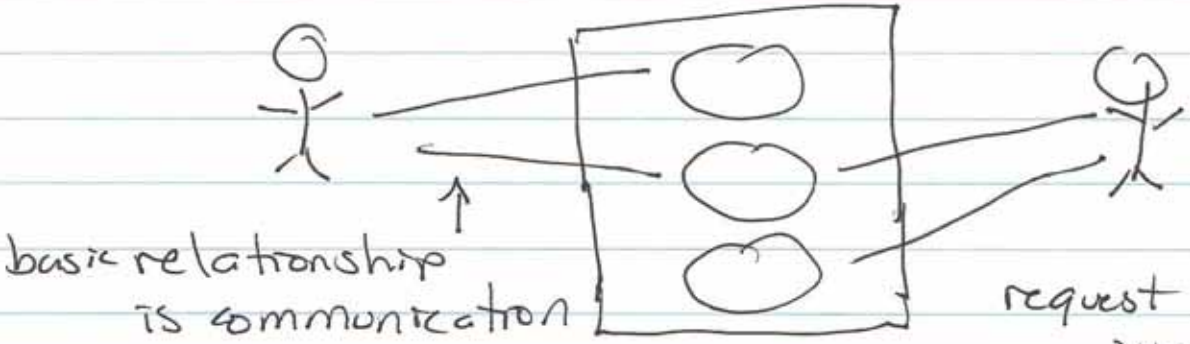
stick figure

actor role - human or system

rectangle

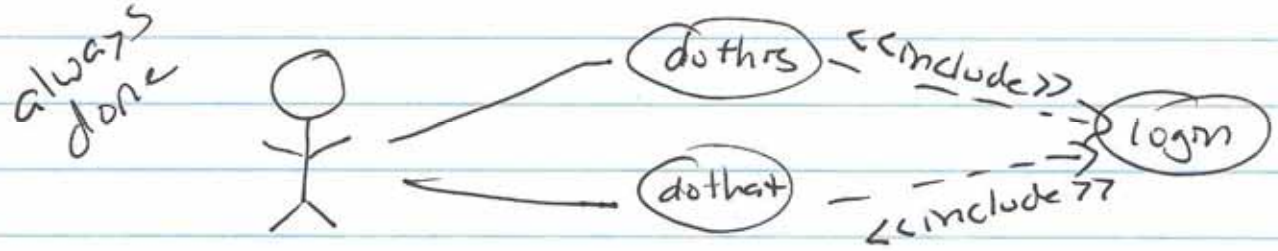


inside us. outside system boundary

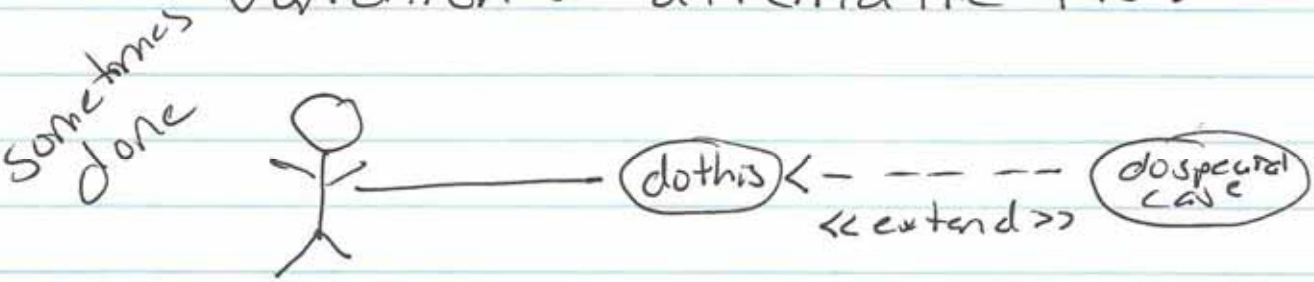


4156 CRUD for assgn 9/29/15

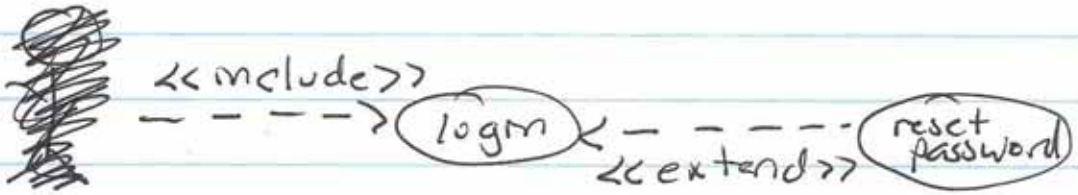
<<include>> relationship - flow of events "calls" the included use case



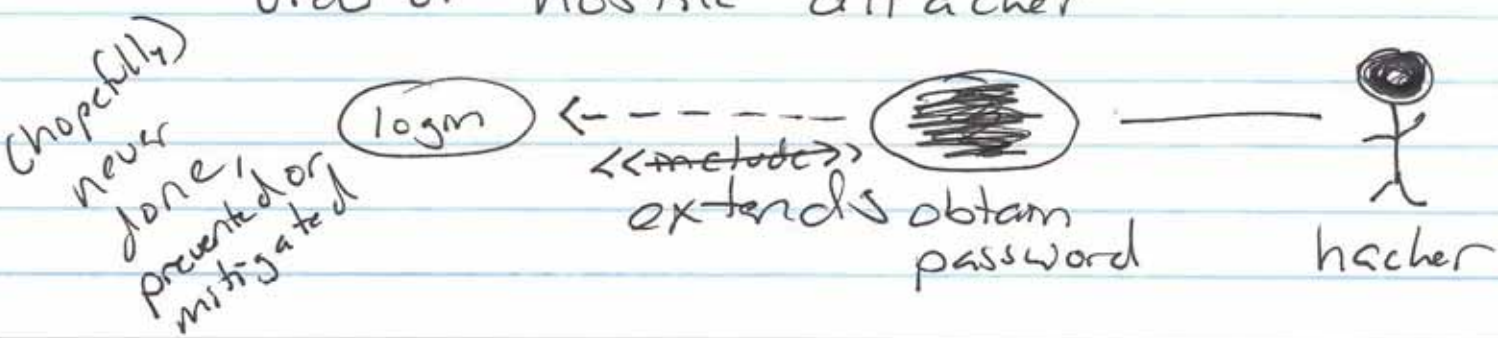
<<extend>> relationship - exceptional variation or alternative flow



note arrow goes other direction



misuse case - use case from point of view of hostile attacher



4156

9/29/15

trp: avoid CRUD
create, read, update, delete

focus on what actor really wants to do, not underlying DB

ex. university registration system
creates, deletes, modifies student schedules -
but student just wants to register for classes

trp: do not reference UI elements like pages or buttons

trp: avoid architectural details

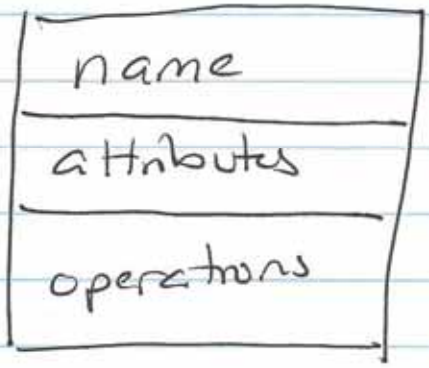
ex. student schedule stored in MySQL

Head First appendix i.1 - i.2
does not address use case diagrams
only class diagrams & sequence diagrams

UML has many others, e.g., activity, state, but we'll focus on those 2

4156 9/29/14

UML class diagrams



everything except class name is optional

attributes – both static fields & instance variables

name : type = initial value

operations – public (+) & private (-) methods

name (params) : return type

only need design level information
omit details like getters & setters, helper methods

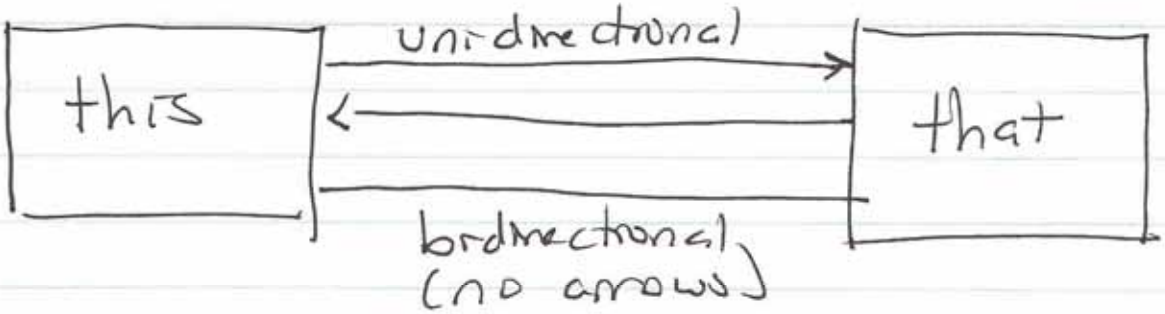
show static structure of individual classes

plus relationships among classes

4156 9/29/15

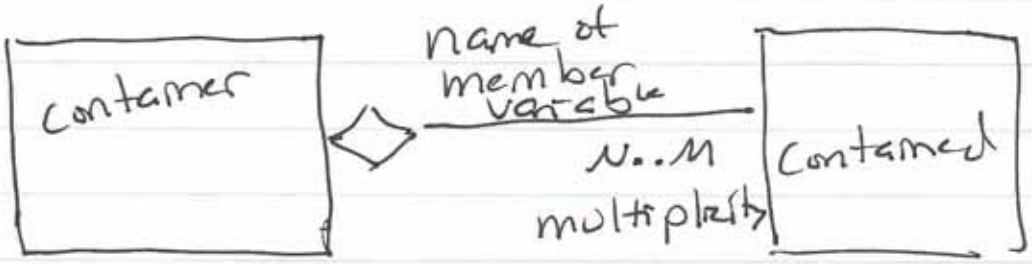
association - most general relationship

instances of one class "know about" or "communication with" instances of another class (or the same class)



many special cases of association

open aggregation HAS-A

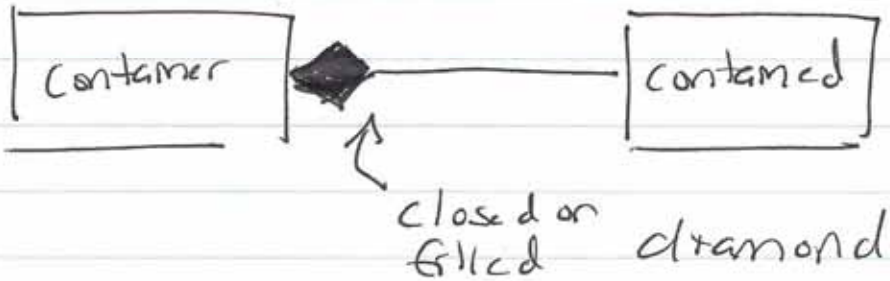


N..M → 0..*, 1..5, etc.

open means contained objects exist independently

4156 9/29/15

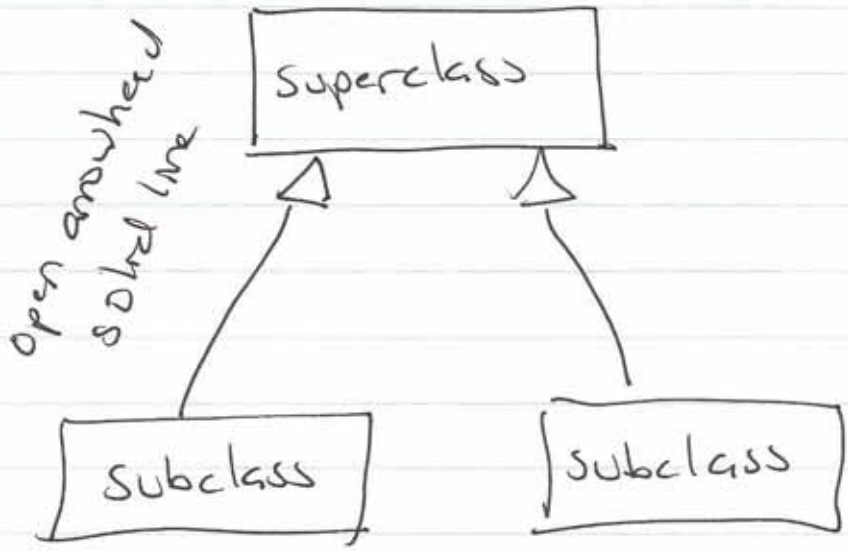
closed aggregation - strong lifecycle relationship



means deleting parent also deletes all children, or perhaps all children must be deleted first before parent can be deleted

other special cases, e.g., for hashmaps

inheritance

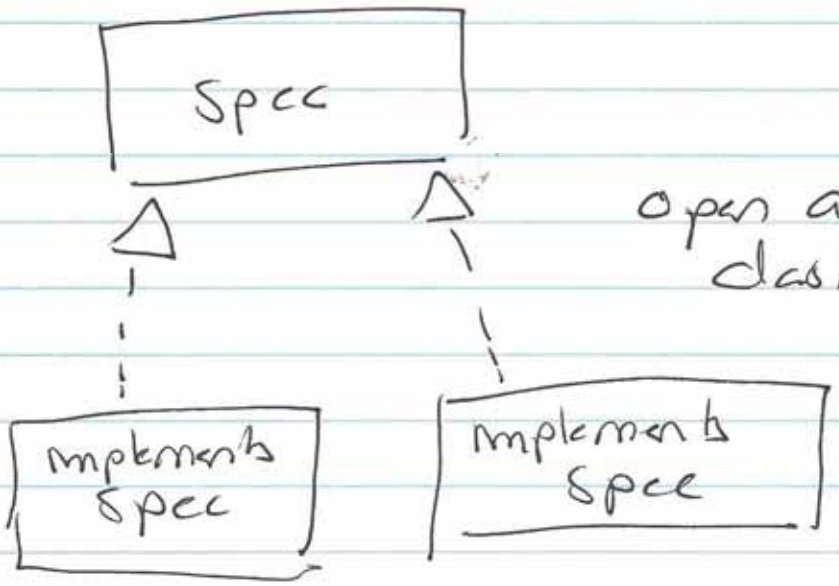


abstraction
generalization
base class

specialization
derived class

4156 9/29/15

implements interface - realization



various editors for UML diagrams

Violet
Argo
vanilla drawing program
others

some only support subset of UML
but all I know of support at
least class diagrams

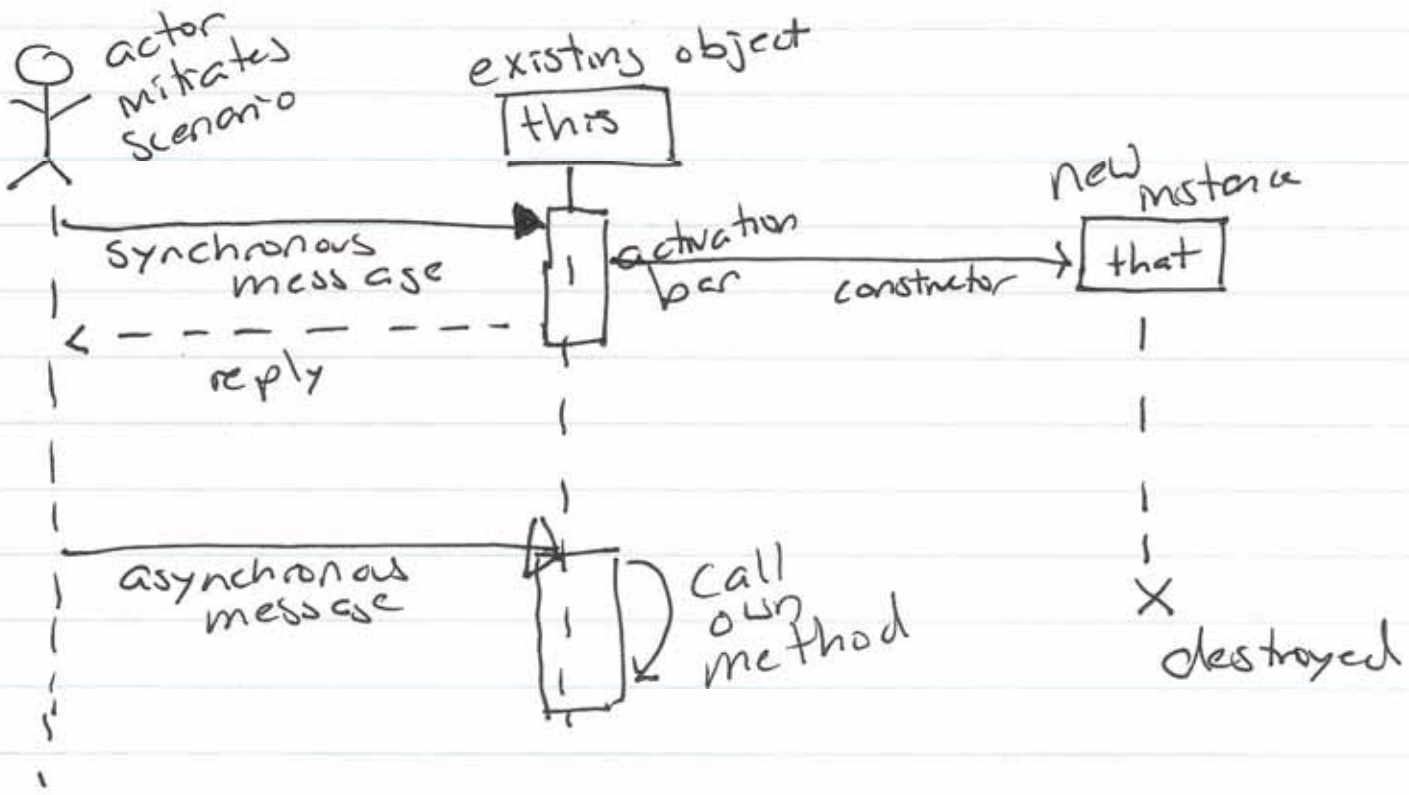
Continue here

4156 9/29/15

besides class diagrams showing state structure, various kinds of dynamic behavioral diagrams

example: sequence diagrams (there are many others)

show how classes work together visual order of events (e.g., method invocation, message)



lifeline (time progress)

message of form name(params): return type
or just name

4156 9/29/15

can show other details such as conditionals & iteration

Where do classes & messages come from?

design- discuss next week

classes - nouns in user stories or use cases

messages- verbs

subject/object of verbs should be actors or classes