Introduction to UML

UML

The Unified Modeling Language, UML, is a language for specifying, visualizing, construction, and documenting the artifacts of software systems.

overview

- class diagrams
  - association
  - composition
  - aggregation
  - multiplicity
  - navigation
  - inheritance
- dynamics

Class Diagrams

Game

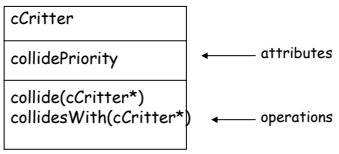
game is a domain concept

Class Diagrams

cGame

cGame is a POP class

Elaborated class diagrams



cCritter is also a POP class

using UML

Association

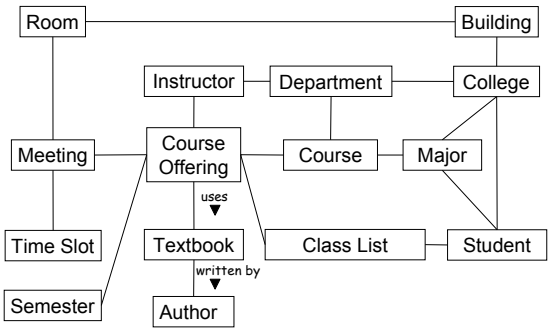
cGame is associated with cCritter

Association Names

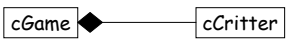
A is associated with B

- A has a B
- A has a method that returns a B
- vice versa
- etc.

Exercise: Identify Likely Association Names



Composition



cGame "has a" cCritter

A "has a" B

```
class cA
private:
        cB bObj;
```

instance member

composition

```
class cA
private:
        cB *pbObj;
```

reference member

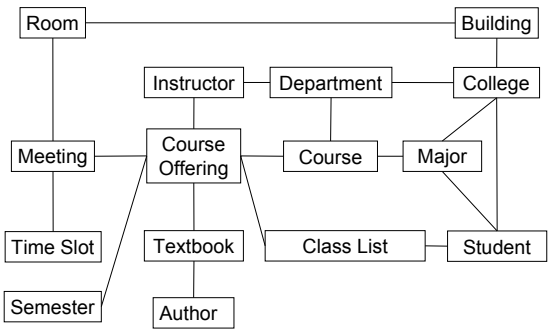
composition or aggregation

composition vs. aggregation

```
class cA
private:
        cB *pbObj;
```

- Composition: cA "owns" the cB object
  - cA initializes pbObj with new
  - cA destroys pbObj with delete
  - cascading delete
- Aggregation: cB is created/destroyed independently of cA

Exercise: Identify Likely Aggregations and Compositions

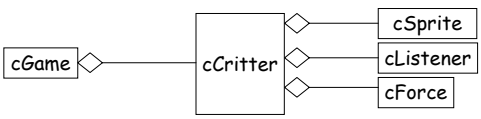


composition vs. aggregation

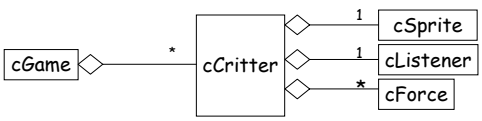
composition

aggregation (or composition or don't know)

Class Diagrams



Multiplicities



- cGame has zero or more cCritter
- cCritter has one cSprite
- cCritter has one cListener
- cCritter has zero or more cForce

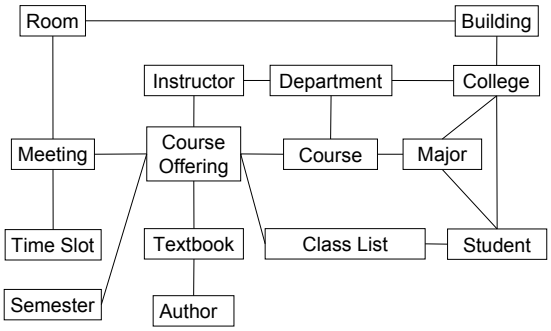
Multiplicities

- The default multiplicity is 1 or don't know.
- m..n means m through n (m and n fixed numbers).
- m..* means m or more.
- * means the same as 0..* (0 or more).
- a, b, c, ... means *one of* a, b, c ...
- 0,1 or 0..1 is a way of saying *optional*.

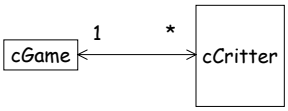
Association with Navigation

classB has a way to navigate to a classA object

Exercise: Identify Important Navigation Paths

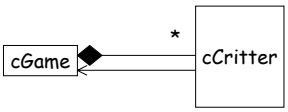


Class Diagrams

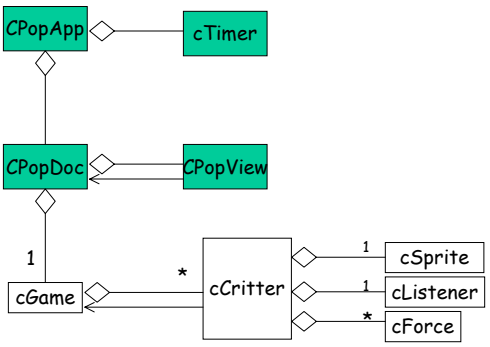


- cGame can access its cCritter objects
- cCritter can access its cGame

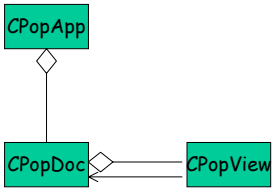
Class Diagrams



the bigger picture



document-view architecture (design pattern)



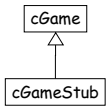
recap

- class diagrams
  - association
  - composition
  - aggregation
  - multiplicity
  - navigation
  - inheritance
- dynamics

overview

- class diagrams
  - association
  - composition
  - aggregation
  - multiplicity
  - navigation
  - inheritance ← next
- dynamics

Inheritance

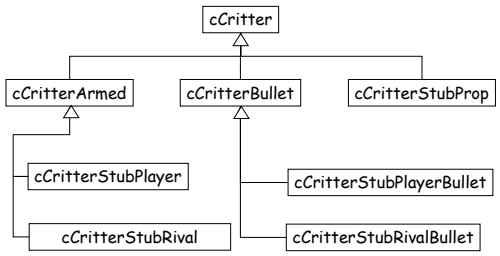


cGameStub is a cGame

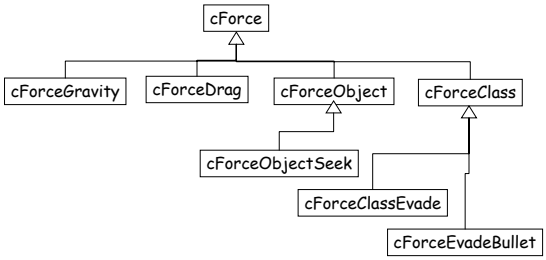
"is a"

class cGameStub : public cGame

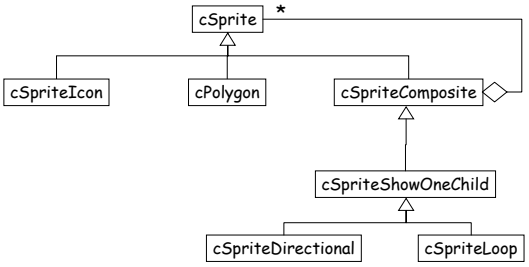
Class Diagrams: critters



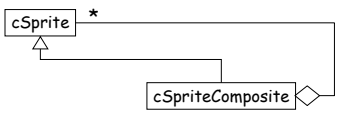
Class Diagrams: forces



Class Diagrams: sprites



design pattern



Class Diagrams

In project teams, construct a UML class diagram for your space invader game.

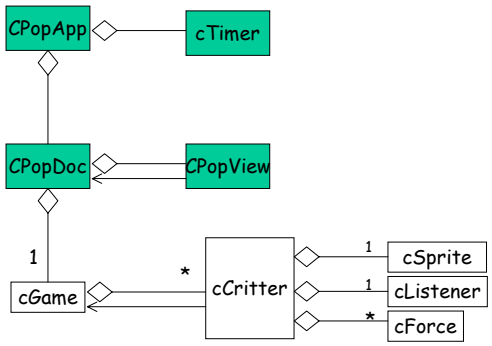
overview

- class diagrams
  - association
  - composition
  - aggregation
  - multiplicity
  - navigation
  - inheritance
- dynamics ← next

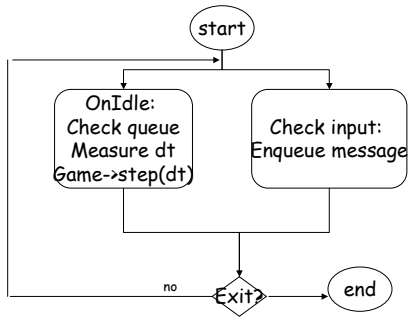
dynamics

a brief introduction ... we'll revisit this later

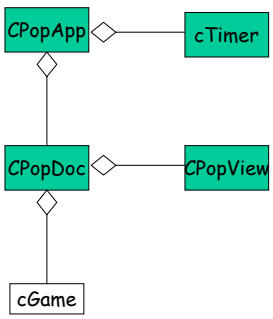
static view



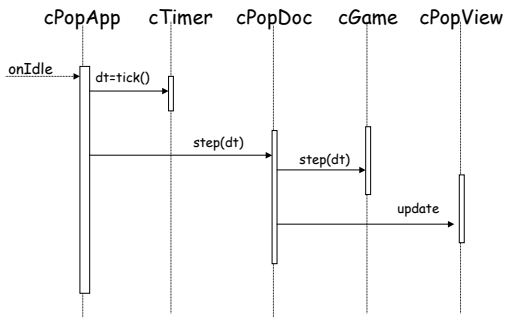
Activity Diagrams



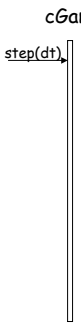
class interactions?



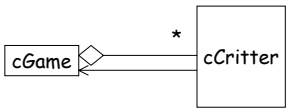
Sequence Diagrams



Sequence Diagrams

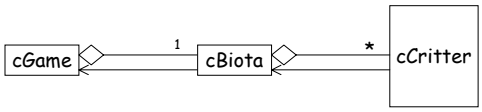


Class Diagrams



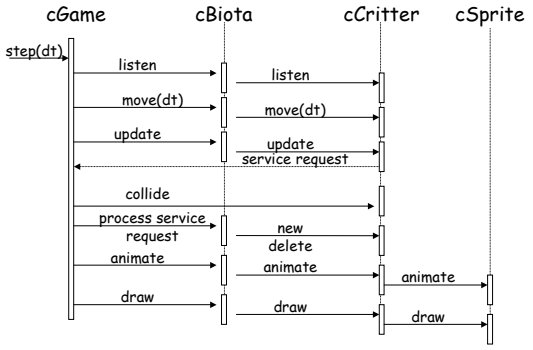
remember this?

Class Diagrams



refined view

Sequence Diagrams



project 1

- prototype due one week from today
- version 1 due two week from today
- all documentation/reports due two weeks from today

grades

feedback

- design doc will be "graded" by Thurs.
- risk analysis will be "graded" by Thurs.
- for Thurs. submit class diagram for project 1

misc

- how do i find out about POP?
  - program by example
  - ask Ed Heaney
  - ask me
- source control
  - tortoiseCVS
- groups/space on Turing