First Model, then Function
— on the value of objects

Want to be a consultant for CS1110?

If you are doing well in the course and are interested, talk to Laurie Buck in Upson 303

Online course evaluations are starting

Completion counts toward your course grade like a quiz

Just before the final, we get a list of students who completed it. Don't see actual evaluations until grades have been submitted

First Model, then Function
— on the value of objects

Gries's revision of lecture by
Michael E. Caspersen
University of Aarhus
Denmark

The foundation of the system should be a model of relevant concepts and phenomena from the problem domain

Object-oriented design

The foundation of the system should be a model of relevant concepts and phenomena from the problem domain

People at a university. Organize into categories

university member
- student
  - grad
  - undergrad
  - nondegree
- faculty
  - prof
  - …
- staff
- …

is-a relation
Make B a subclass of C if every B is a C

Subclass principle
Structure classes so that behavior common to several classes can be defined in a superclass of those classes

Overview

Function first, or model first?

- One problem
- Two solutions: a bad and a good
- Very black and white
- Exaggeration promotes understanding

The Coming of Showboat Inc., (2)

Showboat Inc. and Contractors

Daniel von Schneider

own, manage company

At the beach, rent out boats

Sailor Sam

John H. Acker

freelance programmer

promising employee of software firm

Scarlett Olivia Oakley Patton (Scoop)

Requirements

- von Schneider requests a decision support system for strategic planning
- The system must be able to deliver a daily report
  - number of sessions (hires) of the day
  - average duration of each session
- von Schneider asks for bids ... John H. Acker wins

Function First

- John H. Acker is smart, very smart!
- Realizes he can get by with a two-variable state space
  - *n*: number of finished sessions
  - *totalTime*: sum of duration of finished sessions
- Maintaining *n* is easy
  - finish a session? Increment *n*
- Maintaining *totalTime* is less trivial...

"Clever" Calculations...

s_i: start time for session *i*
e_i: end time for session *i*
duration of session *i*: $(e_i - s_i)$

totalTime
= { definition }
$(e_1 - s_1) + (e_2 - s_2) + \ldots + (e_n - s_n)$
= { remove parentheses }
$e_1 - s_1 + e_2 - s_2 + \ldots + e_n - s_n$
= { re-ordering }
$e_1 + e_2 + \ldots + e_n - s_1 - s_2 - \ldots - s_n$

Start of session: *totalTime* = *totalTime - currentTime*
End of session: *totalTime* = *totalTime* + *currentTime*

Use of the System

Sailor Sam

von Schneider

Control panel

Start | End | Day over

```
*************************************
*     S H O W B O A T   Inc         *
*************************************
Daily Report    21 Apr 2008     22:29

Number of Sessions:                5
Average rental time:          00:12:02
*************************************
```

J. H. Acker's Software Architecture

User Interface
run printDailyReport

System Functions
beginSession endSession DailyReport

Model
Time

Time:
general class in a class library;
not domain specific

t0

Time
- Time ()
- Time (int t)
- Time add (Time t)
- Time subtract (Time t)
- Time divide (int n)
- ...

System Functions

System state

```
private int n= 0; // no of sessions
private Time totalTime= new Time(0); // …
```

```
/** Start a session at the current time */
public void beginSession() {
  Time now= new Time();
  totalTime= totalTime.subtract(now); }
```

```
/** End a session at the current time */
public void endSession() {
  Time now= new Time();
  totalTime= totalTime.add(now); n= n + 1; }
```

```
/** Print a a report */
public void doDailyReport() {
  Time avgTime;
  if (n != 0) avgTime= totalTime.divide(n);
  else avgTime= new Time(0);
  printDailyReport(n, avgTime); }
```

User Interface

```
/** ShowBoat Session Manager … */
public static void run() {
  char command= '.';
  int sessionNr= 0;

  command= getChar();
  while (command != '.') {
    if (command == 's')
        f.beginSession();
    if (command == 'e')
        f.endSession();
    command= getChar();
  }
  printDailyReport(f.doDailyReport());
}
```

Two Months Later

- von Schneider is pleased with system but quickly gets new ideas...
- von Schneider asks John H. Acker to make a few harmless extensions to the daily report to make it more ... interesting:
  - duration of the longest session of the day :-(
  - an extra report at noon :-(
  - distribution of load during the day :-(
  - ... :-(:-(:-(
  - the maximum number of concurrent sessions :-)

John H. Acker Rules **No More**

Scarlett Olivia Oakley Patton's Object-Oriented Perspective

- John H. Acker's system: developed solely by focusing on functional requirements of the system
- No software representation of concepts and phenomena in the problem domain
- **Foundation of system should be a model of relevant concepts and phenomena from the problem domain**
- **Relevance**: determined by the requirements specification
- But what are the relevant concepts...?

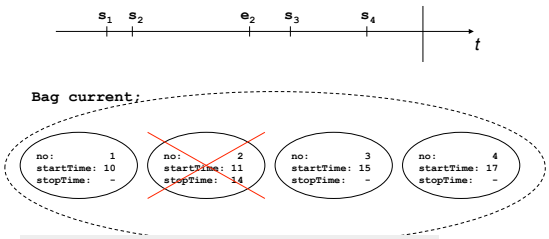
The Key Concept: Session

- Relevant concepts: found in requirements spec (functional requirements)
  - no. **sessions** (hires) per day
  - average **session** duration
- Both requirements expressed in terms: **session**
- Central property of a session **duration**
- Must be able to **start** and **stop** a session
- Must be able to **identify** sessions (unique id)

```
class Session {
  /** Constr: session with id n,
      start/stop times of 0 */
  public Session (int n)

  /** = session id */
  public int id()

  /** Start session now */
  public void start()

  /** End session now */
  public void stop()

  /** = duration of session.
    Pre: session is stopped */
  public Time duration()
}
```

Session Objects

A session object for each unfinished session:



Bag: Like a set but can have more than one occurrence of an element

Use of the System

Sailor Sam

von Schneider

Control panel

Start End Day over

1 2 3
4 5 6
7 8 9
0

```
*************************************
*     S H O W B O A T   Inc         *
*************************************
Daily Report    21 Apr 2008     22:29

Number of Sessions:                5
Average rental time:          00:12:02
*************************************
```

Scarlett's Software Architecture

User Interface
run printDailyReport

System Functions
beginSession endSession doDailyReport

Model
Time *Session* *Report* *Bag*

User Interface

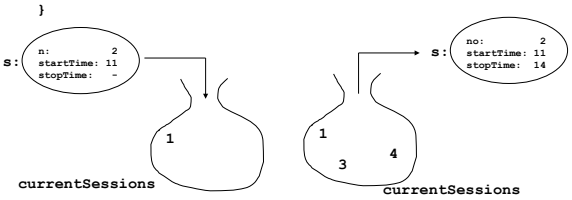
```
public static void run() {
  char command= '.';
  int sessionNo= 0;

  command= getChar();
  while (command != '.') {
    switch (command) {
      case 's': case
        sessionN                    ssion(sessionNo); break;
      case 'e
        el
                                    tInt(); f.endSession(no); break;
                                    ;
    }
    command= getChar();
  }
  printDailyReport(f.doDailyReport());
}
```

As before (almost)

System Functions

```
public static void
  beginSession(int n) {
  …
}
```

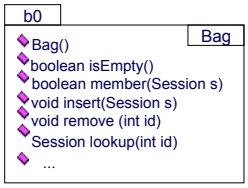
```
public static void
    endSession (int n) {
}
```




```
doDailyReport()
is as before
```

Bag

Bag is a general class in a class library;
like Time, it is not a domain specific class



An object, from pt of view of the user

Two Months Later

- von Schneider is pleased about the system but quickly starts getting new ideas...
- Calls Scarlett OOP and asks her to make a few harmless extensions to the daily report to make it more ... interesting:
  - duration of the longest session of the day :-)
  - an extra report at noon :-)
  - distribution of load during the day :-)
  - ... :-) :-) :-)
  - the maximum number of concurrent sessions :-)

Scarlett Olivia Oakley Patton Still Rules

A (logical) Three Tier Architecture

Interface Component (View)	Function Component (Controller)	Model Component (Model)

more dynamic ⟷ more static