New Course Development: 461L Software Engineering and Design Laboratory

Prof. Miryung Kim
Electrical and Computer Engineering
The University of Texas at Austin

Synopsis

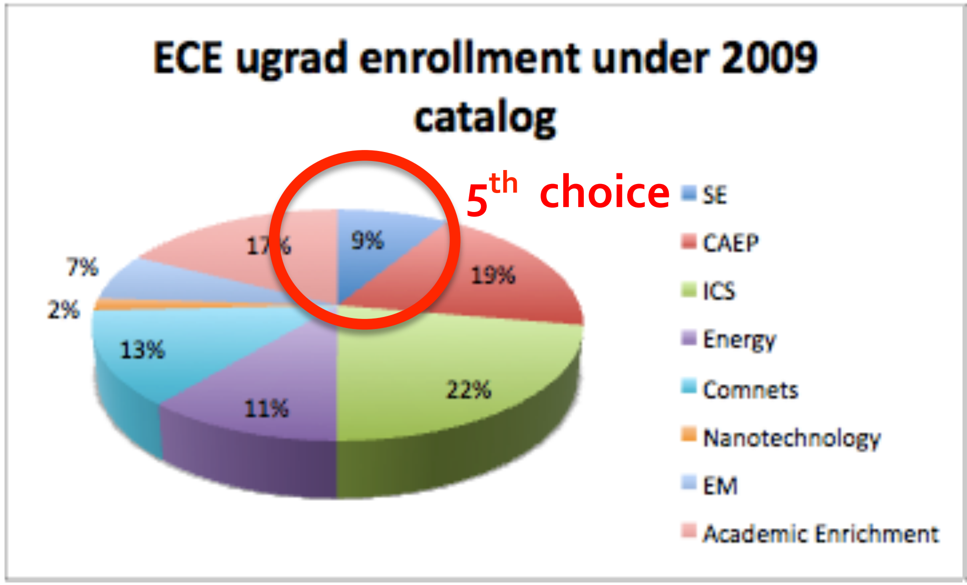
- A new junior level, software engineering & design laboratory class was created to meet the needs of our ECE undergraduates
- Class activities and self-paced tool tutorials helped students to engage in highly abstract subject matter and gain confidence in working with large software.

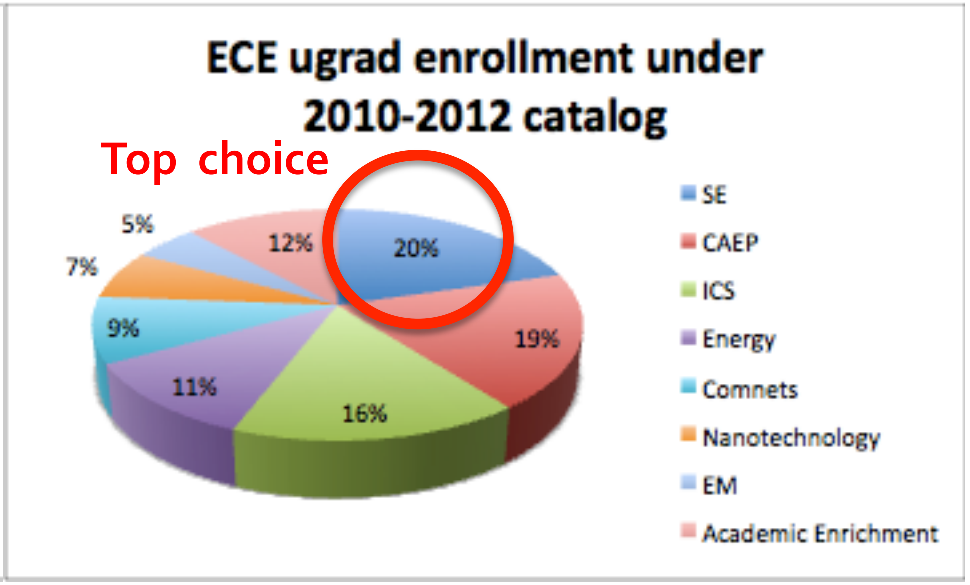
Outline

- **Motivation**
- Key Objectives
- Course Structure
- Example Instruction Materials and Methods
- Lessons Learned
- Conclusions

Motivation 1.
Tech Area Selection Trends in ECE

- Among ECE undergraduates, Software Engineering and Design Core (SE) has become the most popular technical area.





Motivation 2.
Lack of Core SE Laboratory Class

- SE tech area did not have its own core laboratory class in the old 2008 catalog.
- A lack of emphasis on hands-on experience in 422C, 360F, and 360C

EE	ComNets	445S Digital Signal Processing Lab
	ICS	438 Electronic Circuits
	Energy	462L Power Electronics
	EM	462L or 438
	Nanotechnology	440L Micro Elec Fabrication
CE	CAEP	445L Microprocessor Lab
	Soft. Engineering	**No lab course. Take 445L Instead**

Motivation 3. Prepare our students for professional careers in SE

Example Career Paths

Process Control Engineer	Software Engineer
Tech Program Manager=>Senior PM	Manager=>Senior Manager=> Director
Senior Product Manager => Senior Manager	CTO of his own start-up
Founder of his own start-up=> Director	Senior Vice President & Chief Digital Officer
BS in Applied Physics and Electrical Eng.	BS in Systems Engineering

Art produced by virtuosos with years of experience?
Business management (organization & planning)?

VS.

Science & Engineering

Outline

- Motivation
- **Key Objectives**
- Course Structure
- Example Instruction Materials and Methods
- Lessons Learned
- Conclusions

Key Objectives

- **Hands on experience**
- Teach tools required by industry
- Systematic engineering methods
- Realistic project tasks

Key Objectives

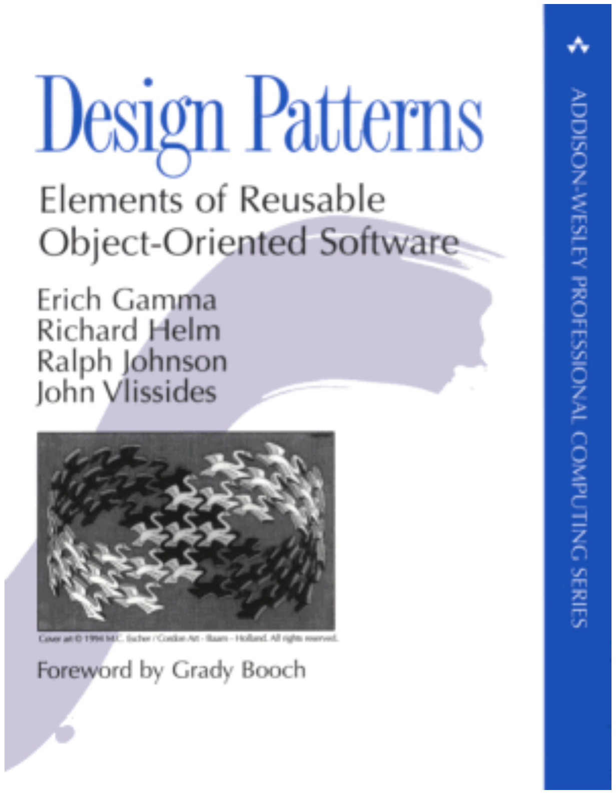
- Hands on experience
- **Teach tools required by industry**
- Systematic engineering methods
- Realistic project tasks

Version Control Tools, Unix Utilities & Shell Scripting, UML Modeling Tool, Build Mgmt Tool, Unit Testing Tool, Debugger, Profiler, etc.

Key Objectives

- Hands on experience
- Teach tools required by industry
- **Systematic engineering methods**
- Realistic project tasks

Design Patterns , Unit Testing, Regression Testing, Formal Methods, Static and Dynamic Program Analysis


Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

Key Objectives

- Hands on experience
- Teach tools
- Systematic engineering methods
- **Realistic project tasks**

Building a **small** project **from scratch** ✗

vs.

Evolving a **large system** ✓ through **feature additions**

Course Structure

Lectures

(3 hours per week)

Concepts, principles, & methods

Engaging students through ***class activities***, ***discussions***, and ***demos***

Tool Tutorials and Exercises

(lab: 3 hours)

Tool installation & following **self-paced tutorials**

exercise tasks at each milestone

Team Projects +

New in Fall 2012

Homework

+

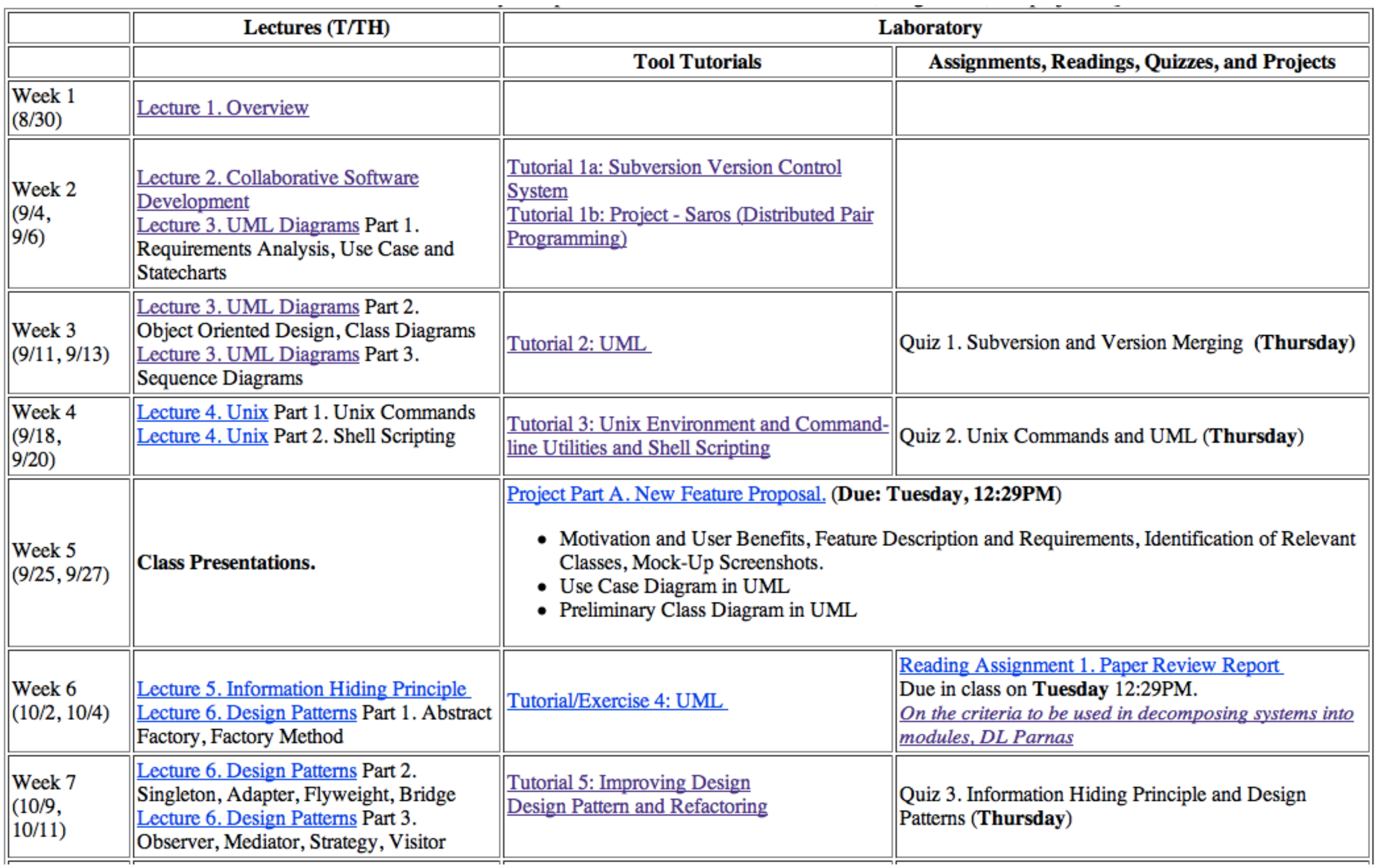
Paper Reviews

+

Quizzes +

Exams

Course URL

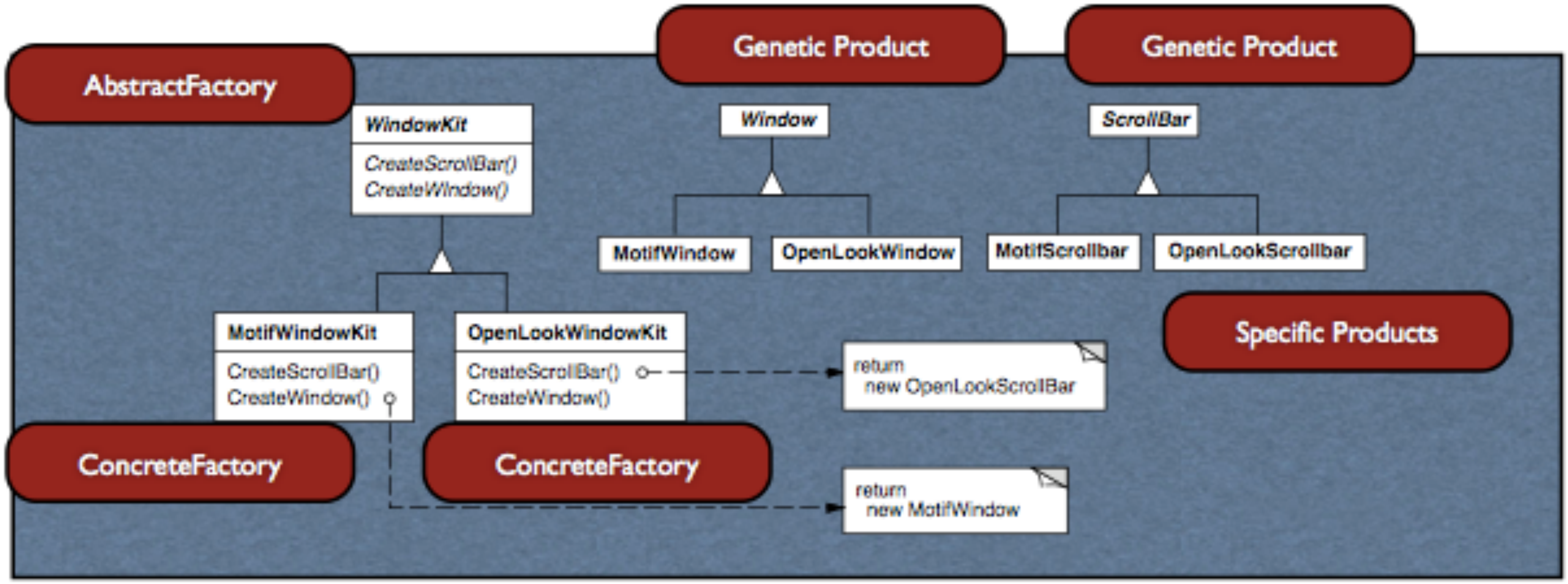


	Lectures (T/TH)	Laboratory	
		Tool Tutorials	**Assignments, Readings, Quizzes, and Projects**
Week 1 (8/30)	Lecture 1. Overview		
Week 2 (9/4, 9/6)	Lecture 2. Collaborative Software Development Lecture 3. UML Diagrams Part 1. Requirements Analysis, Use Case and Statecharts	Tutorial 1a: Subversion Version Control System Tutorial 1b: Project - Saros (Distributed Pair Programming)	
Week 3 (9/11, 9/13)	Lecture 3. UML Diagrams Part 2. Object Oriented Design, Class Diagrams Lecture 3. UML Diagrams Part 3. Sequence Diagrams	Tutorial 2: UML	Quiz 1. Subversion and Version Merging (**Thursday**)
Week 4 (9/18, 9/20)	Lecture 4. Unix Part 1. Unix Commands Lecture 4. Unix Part 2. Shell Scripting	Tutorial 3: Unix Environment and Command-line Utilities and Shell Scripting	Quiz 2. Unix Commands and UML (**Thursday**)
Week 5 (9/25, 9/27)	**Class Presentations.**	Project Part A. New Feature Proposal. **(Due: Tuesday, 12:29PM)** • Motivation and User Benefits, Feature Description and Requirements, Identification of Relevant Classes, Mock-Up Screenshots. • Use Case Diagram in UML • Preliminary Class Diagram in UML	
Week 6 (10/2, 10/4)	Lecture 5. Information Hiding Principle Lecture 6. Design Patterns Part 1. Abstract Factory, Factory Method	Tutorial/Exercise 4: UML	Reading Assignment 1. Paper Review Report Due in class on **Tuesday** 12:29PM. *On the criteria to be used in decomposing systems into modules, DL Parnas*
Week 7 (10/9, 10/11)	Lecture 6. Design Patterns Part 2. Singleton, Adapter, Flyweight, Bridge Lecture 6. Design Patterns Part 3. Observer, Mediator, Strategy, Visitor	Tutorial 5: Improving Design Design Pattern and Refactoring	Quiz 3. Information Hiding Principle and Design Patterns (**Thursday**)

Outline

- Motivation
- Key Objectives
- Course Structure
- **Example Instruction Materials and Methods**
- Lessons Learned
- Conclusions

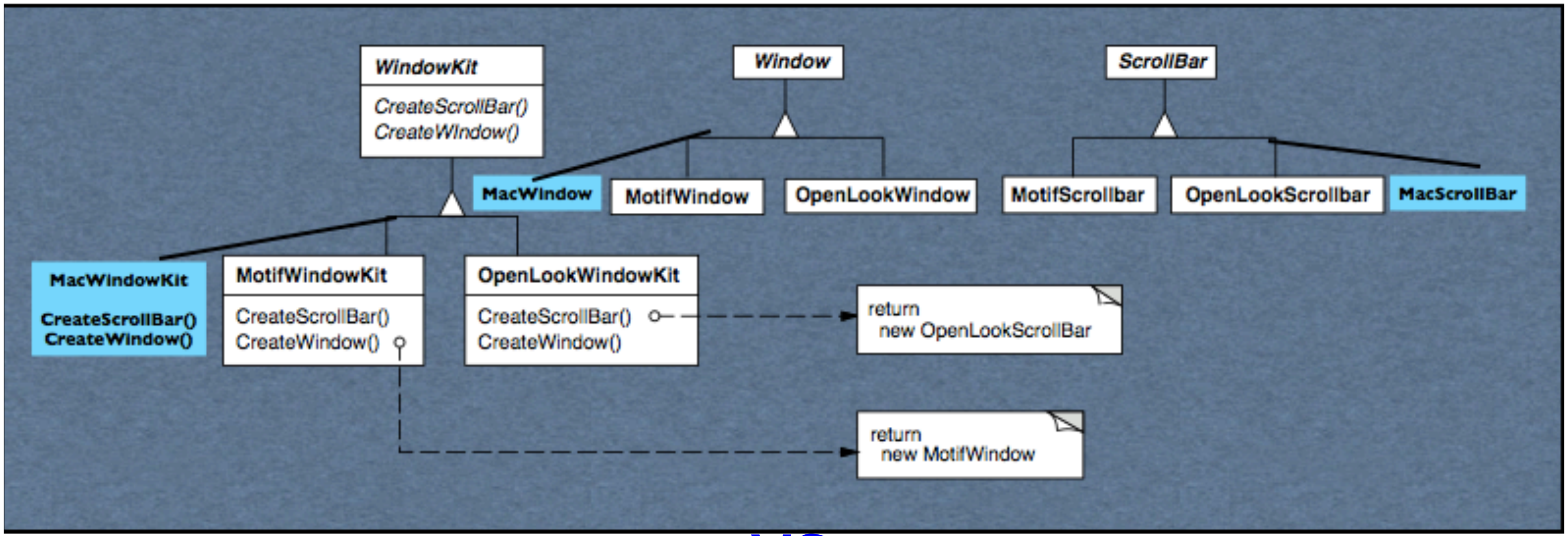
Lecture Example 1.
Design Pattern Critiquing



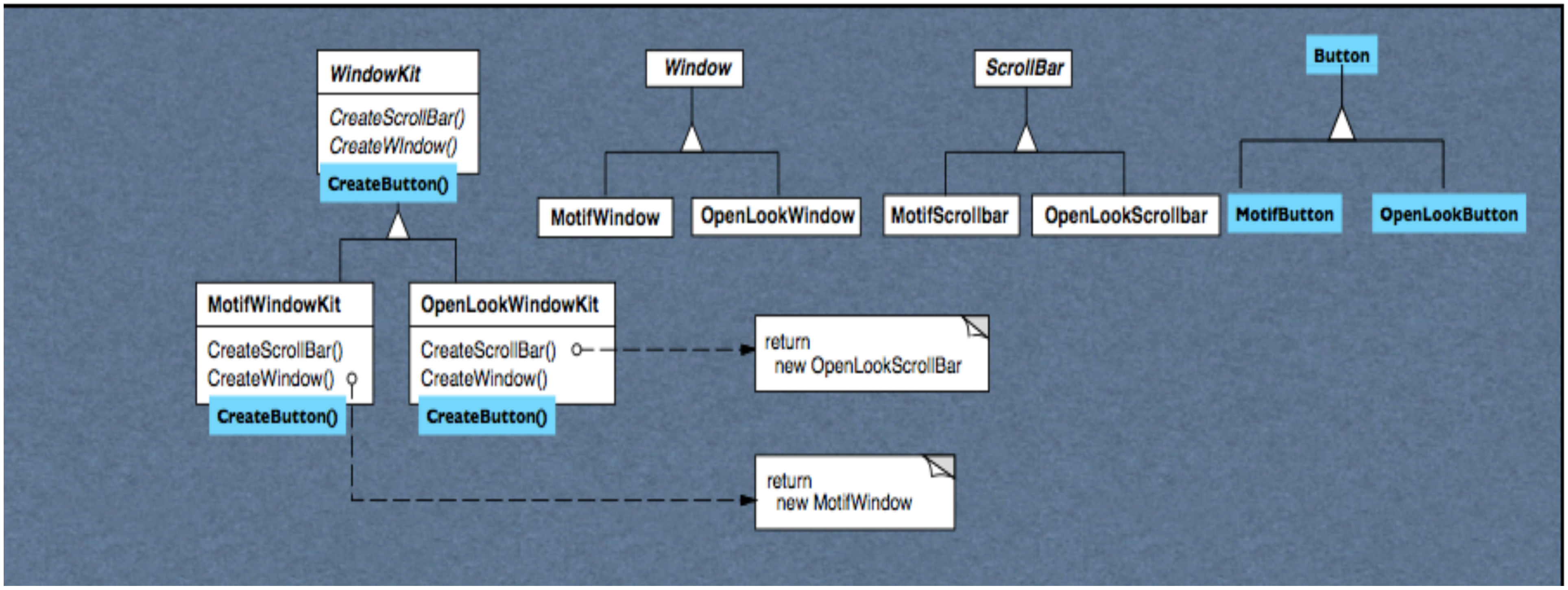
Extension Scenario 1.
How about adding a different look and feel such as **MacWindowKit**?

VS.

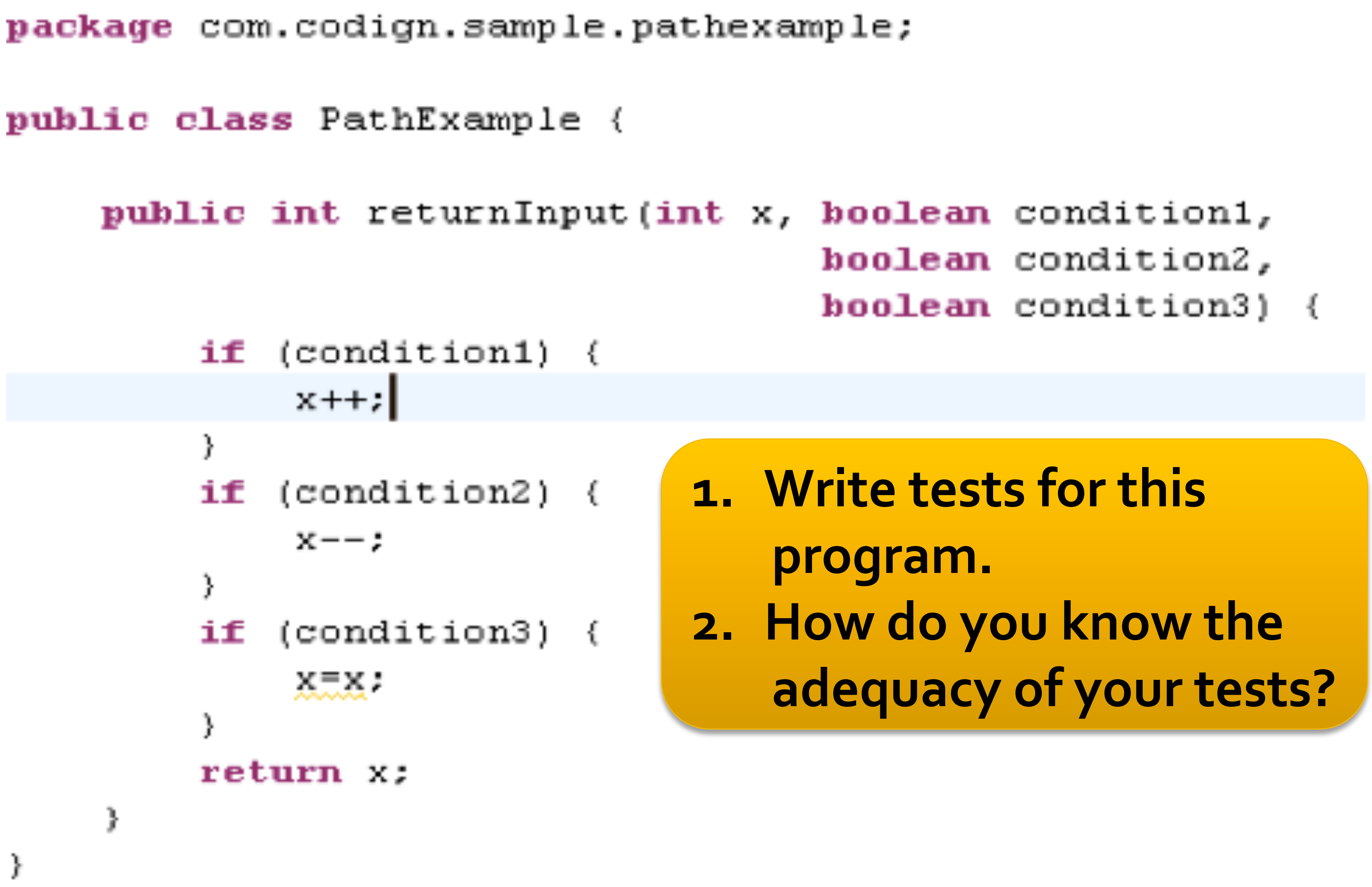
Extension Scenario 2.
How about adding a new type of an object, **Button**?


WindowKit
CreateScrollBar()
CreateWindow()
Window
ScrollBar
MacWindow
MotifWindow
OpenLookWindow
MotifScrollbar
OpenLookScrollbar
MacScrollBar
MacWindowKit
CreateScrollBar()
CreateWindow()
MotifWindowKit
CreateScrollBar()
CreateWindow()
OpenLookWindowKit
CreateScrollBar()
CreateWindow()
return new OpenLookScrollBar
return new MotifWindow

vs.


WindowKit
CreateScrollBar()
CreateWindow()
CreateButton()
Window
ScrollBar
Button
MotifWindow
OpenLookWindow
MotifScrollbar
OpenLookScrollbar
MotifButton
OpenLookButton
MotifWindowKit
CreateScrollBar()
CreateWindow()
CreateButton()
OpenLookWindowKit
CreateScrollBar()
CreateWindow()
CreateButton()
return new OpenLookScrollBar
return new MotifWindow

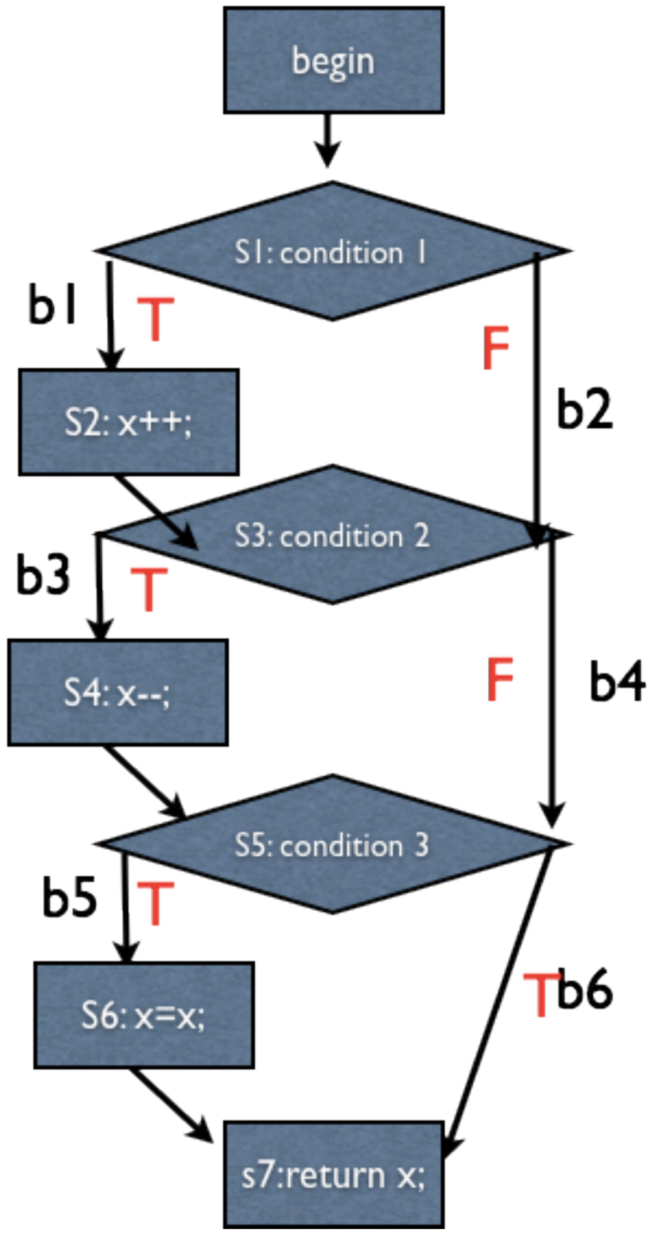
Lecture Example 2.
Test Coverage



```java
package com.codign.sample.pathexample;

public class PathExample {

    public int returnInput(int x, boolean condition1,
                                  boolean condition2,
                                  boolean condition3) {
        if (condition1) {
            x++;
        }
        if (condition2) {
            x--;
        }
        if (condition3) {
            x=x;
        }
        return x;
    }
}
```

1. Write tests for this program.
2. How do you know the adequacy of your tests?

Control Flow Graph



Fill out the following code coverage table by running the program with the following inputs

input	exercised statements	exercised branches	exercised paths
(cond1=true, cond2=true, cond3=true)	s1, s2, s3, s4, s5, s6, s7	b1, b3, b5	[b1, b3, b5]
Coverage			
(cond1=false, cond2=false, cond3=false)			
Coverage			
(cond1=false, cond2=true, cond3=true)			
Coverage			

Lecture Example 3. Software Inspection

which pre-condition should hold here?

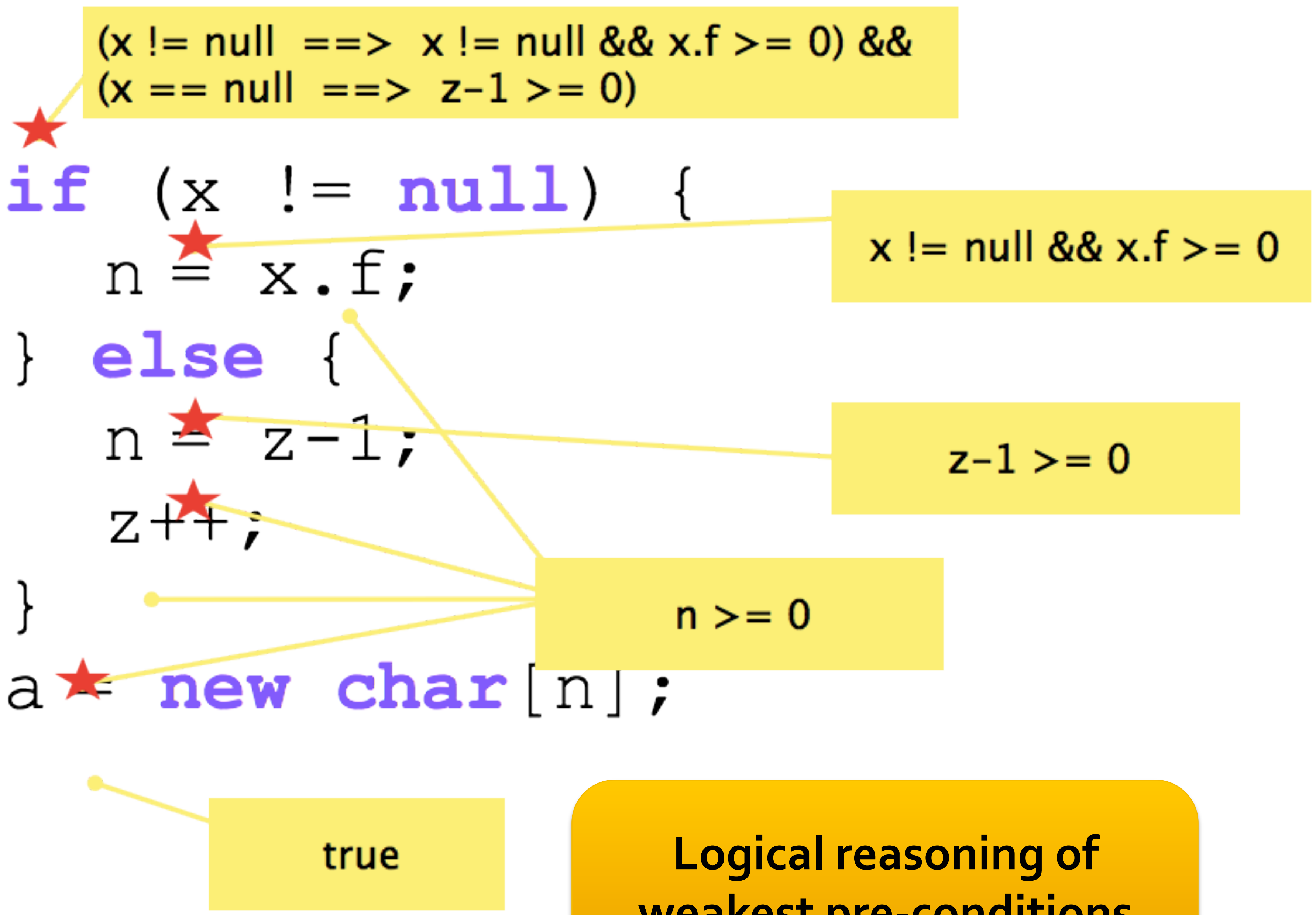
★

```
if (x != null) {
   n = x.f;
} else {
   n = z-1;
   z++;
}
a = new char[n];
```

★

true

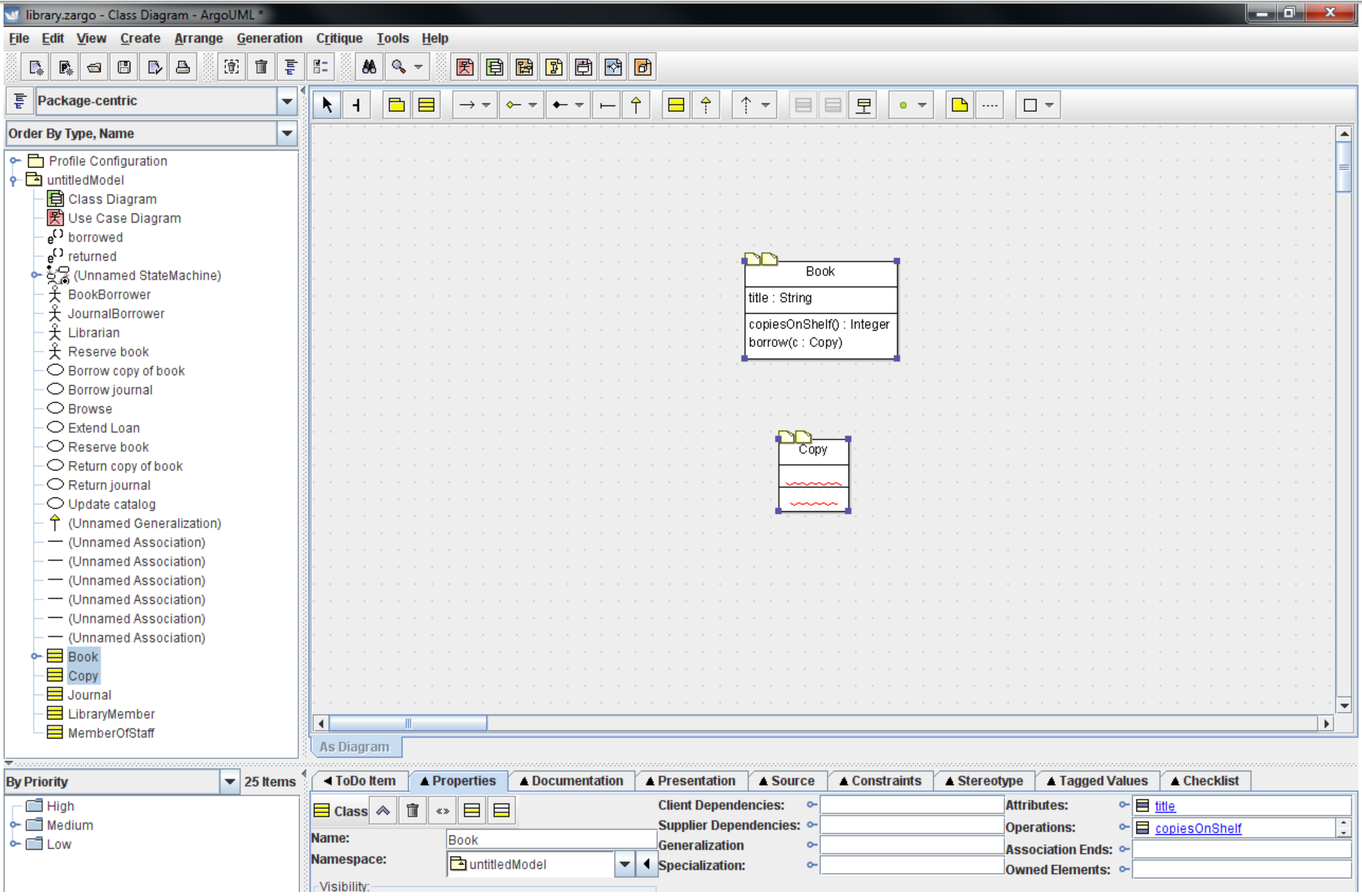
1. Is this program correct?
2. Which pre-condition does this program need to satisfy?



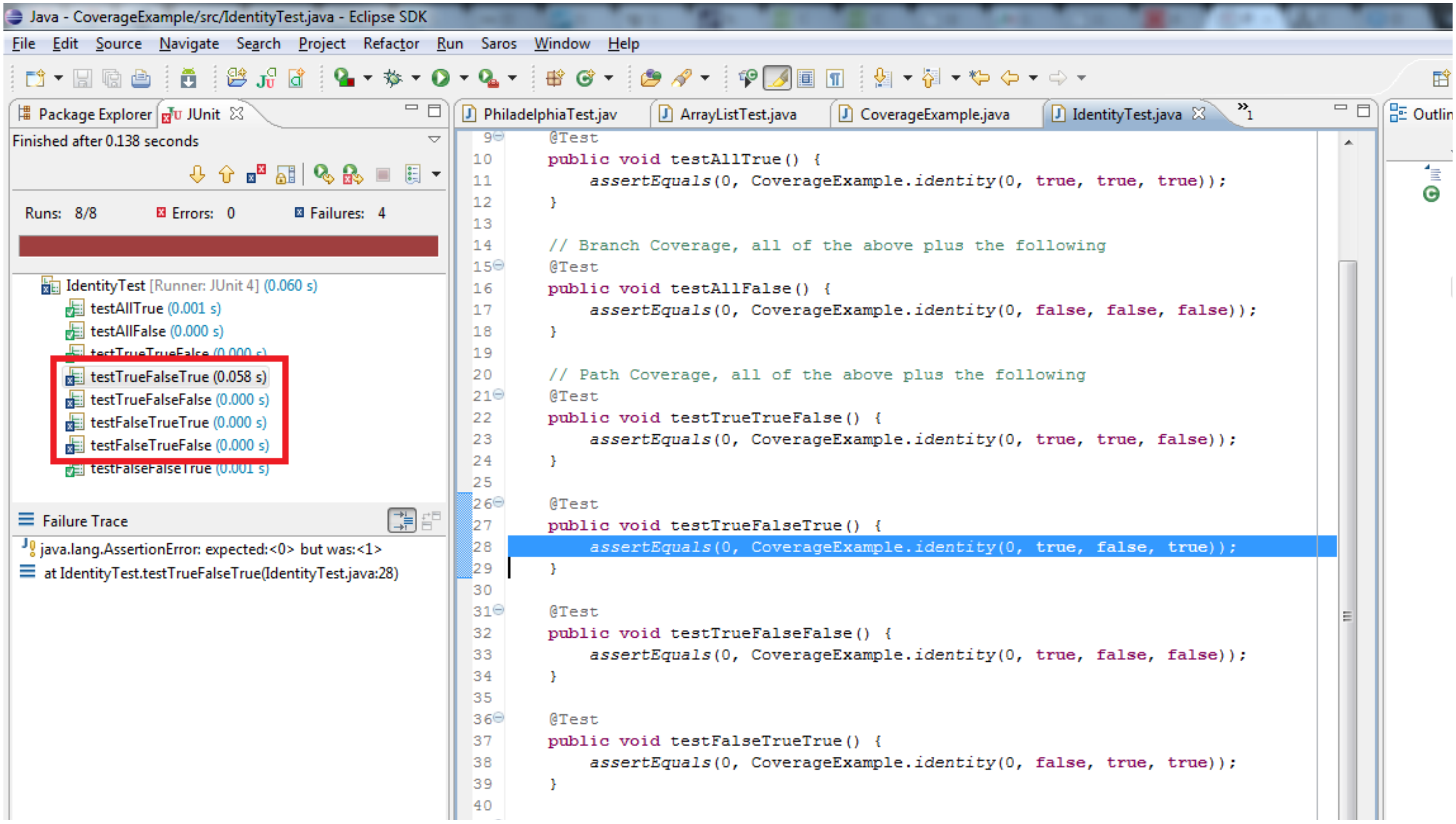
```
if (x != null) {
    n = x.f;
} else {
    n = z-1;
    z++;
}
a = new char[n];
```

Logical reasoning of weakest pre-conditions and loop invariant

Self-paced Tutorial Example 1. UML Design and Modeling Tool



Self-paced Tutorial Example 2. Test Coverage with JUnit


Java - CoverageExample/src/IdentityTest.java - Eclipse SDK
File Edit Source Navigate Search Project Refactor Run Saros Window Help
Package Explorer
JUnit
PhiladelphiaTest.jav
ArrayListTest.java
CoverageExample.java
IdentityTest.java
Outlin
Failure Trace

Finished after 0.138 seconds

Runs: 8/8 Errors: 0 Failures: 4

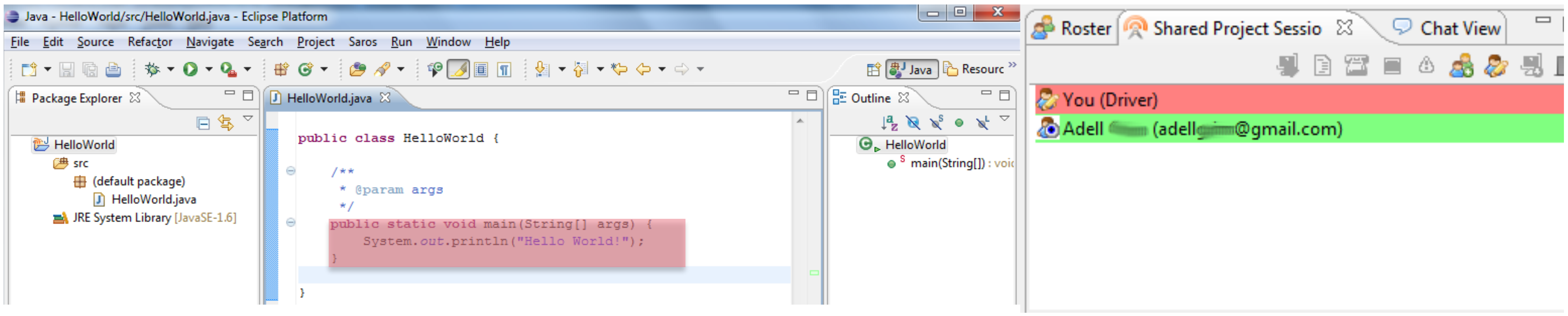
```
IdentityTest [Runner: JUnit 4] (0.060 s)
    testAllTrue (0.001 s)
    testAllFalse (0.000 s)
    testTrueTrueFalse (0.000 s)
    testTrueFalseTrue (0.058 s)
    testTrueFalseFalse (0.000 s)
    testFalseTrueTrue (0.000 s)
    testFalseTrueFalse (0.000 s)
    testFalseFalseTrue (0.001 s)
```

```
java.lang.AssertionError: expected:<0> but was:<1>
at IdentityTest.testTrueFalseTrue(IdentityTest.java:28)
```

```java
    @Test
    public void testAllTrue() {
        assertEquals(0, CoverageExample.identity(0, true, true, true));
    }

    // Branch Coverage, all of the above plus the following
    @Test
    public void testAllFalse() {
        assertEquals(0, CoverageExample.identity(0, false, false, false));
    }

    // Path Coverage, all of the above plus the following
    @Test
    public void testTrueTrueFalse() {
        assertEquals(0, CoverageExample.identity(0, true, true, false));
    }

    @Test
    public void testTrueFalseTrue() {
        assertEquals(0, CoverageExample.identity(0, true, false, true));
    }

    @Test
    public void testTrueFalseFalse() {
        assertEquals(0, CoverageExample.identity(0, true, false, false));
    }

    @Test
    public void testFalseTrueTrue() {
        assertEquals(0, CoverageExample.identity(0, false, true, true));
    }

```

Team Project

- 4 to 5 person team
- Adding a feature to an open source project
- SAROS is an Eclipse plug-in for distributed collaborative programming, developed by Lutz Prechelt at Freie Univ. Berlin in Germany



```
public class HelloWorld {

    /**
     * @param args
     */
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

You (Driver)
Adell [redacted] (adell[redacted]@gmail.com)

Team Project

Part A: New Feature Proposal

Motivation
User Benefits
Feature Descriptions
Mock-up Screenshot
Preliminary Design in UML

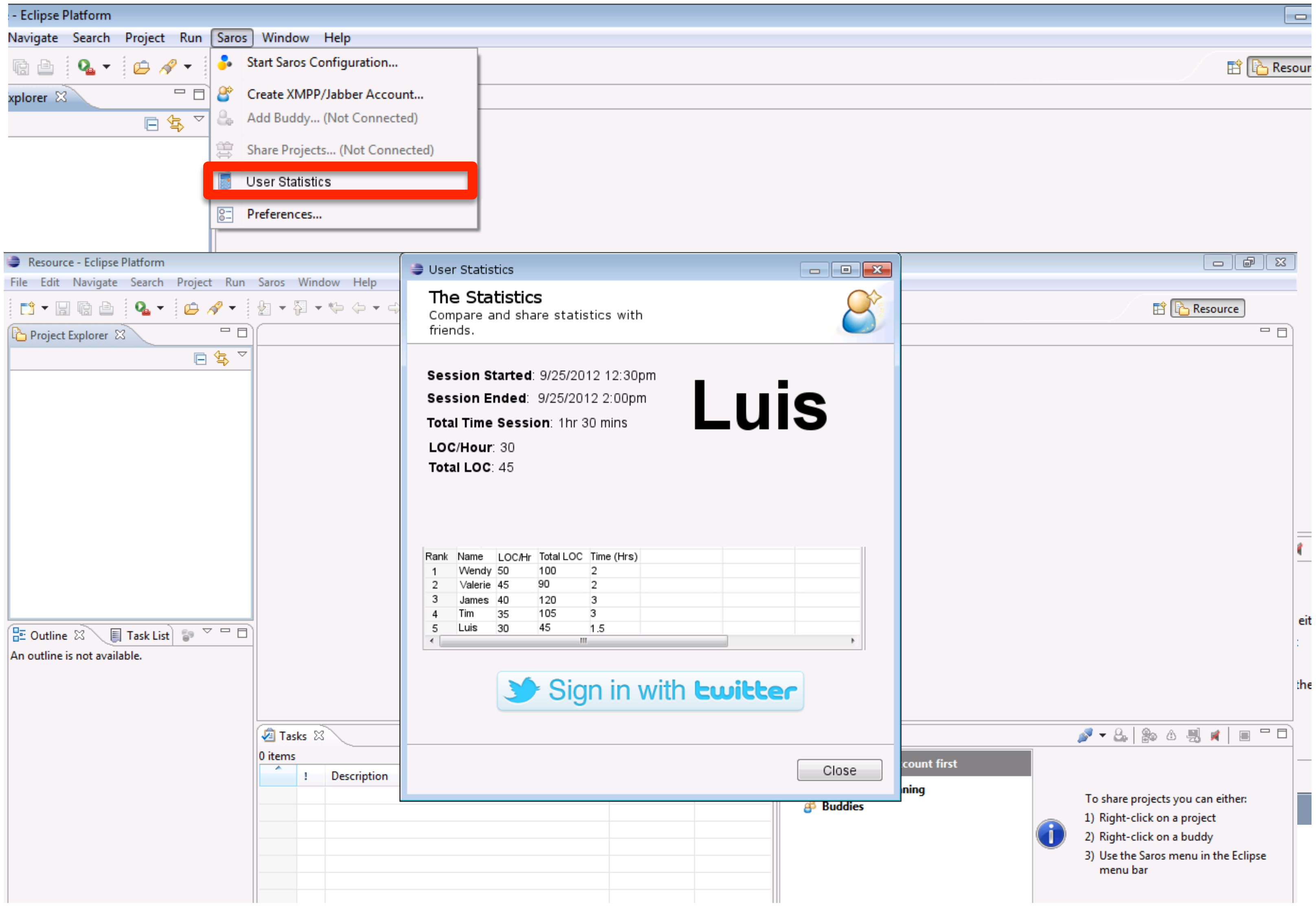
Part B: Implementation Progress

Design Extension in UML
API Descriptions
User Interfaces
Test Scenarios and Test Cases in JUnit

Part C Final Feature Demonstration

Design Extension in UML
API Descriptions
User Interfaces
Test Scenarios and Test Cases in JUnit
User Manuals

Example: SAROS User Statistics with Logging & Tweeting Features


- Eclipse Platform
Navigate Search Project Run Saros Window Help
Start Saros Configuration...
Create XMPP/Jabber Account...
Add Buddy... (Not Connected)
Share Projects... (Not Connected)
User Statistics
Preferences...
Resource - Eclipse Platform
Project Explorer
Outline
Task List
An outline is not available.
Tasks
0 items
Description
Buddies
User Statistics
Sign in with twitter
Close
To share projects you can either:
1) Right-click on a project
2) Right-click on a buddy
3) Use the Saros menu in the Eclipse menu bar

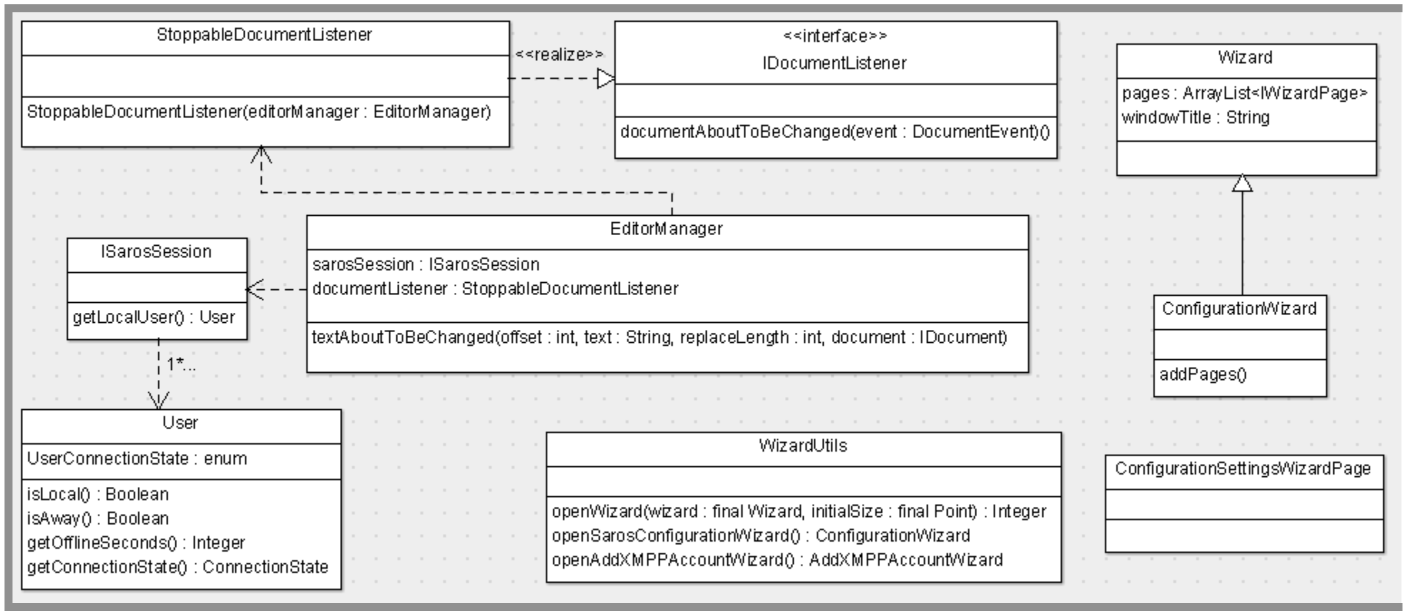
The Statistics
Compare and share statistics with friends.

Session Started: 9/25/2012 12:30pm
Session Ended: 9/25/2012 2:00pm
Total Time Session: 1hr 30 mins
LOC/Hour: 30
Total LOC: 45

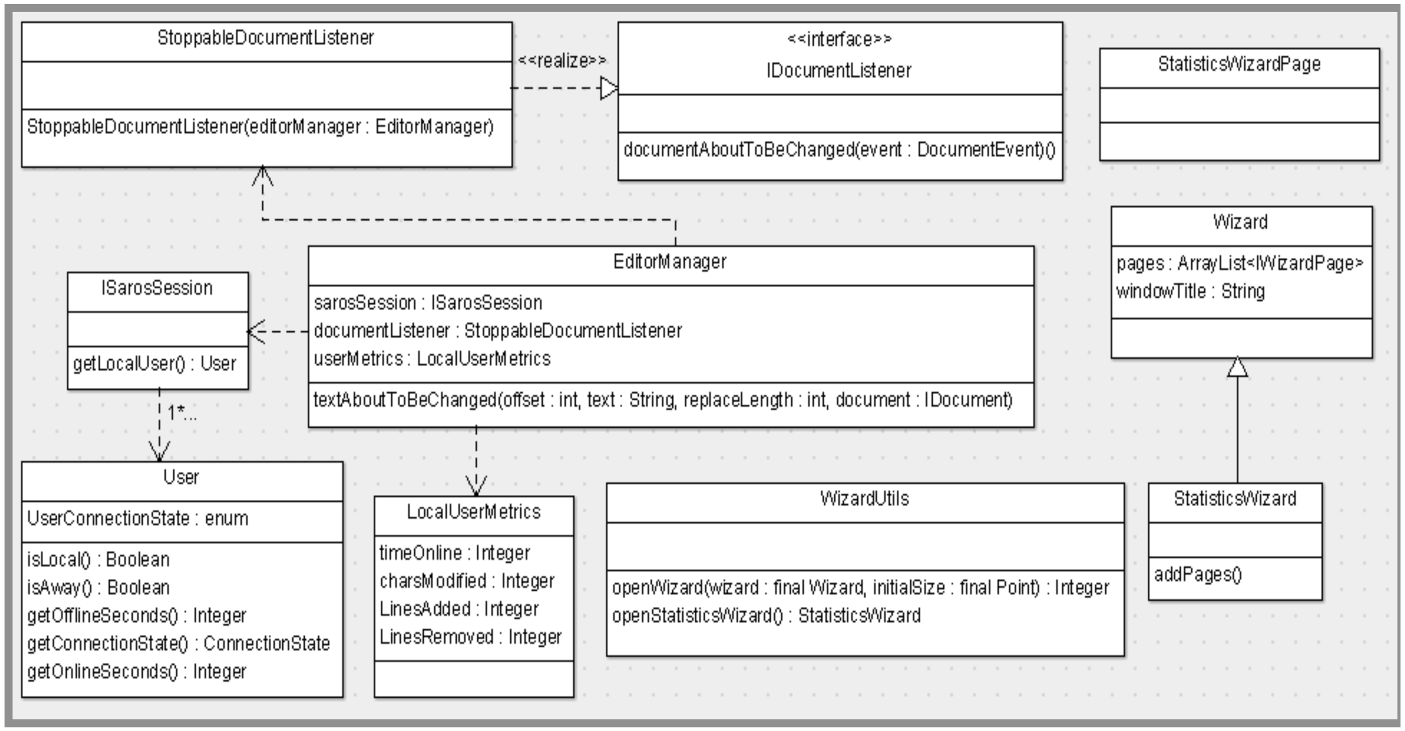
Luis

Rank	Name	LOC/Hr	Total LOC	Time (Hrs)
1	Wendy	50	100	2
2	Valerie	45	90	2
3	James	40	120	3
4	Tim	35	105	3
5	Luis	30	45	1.5

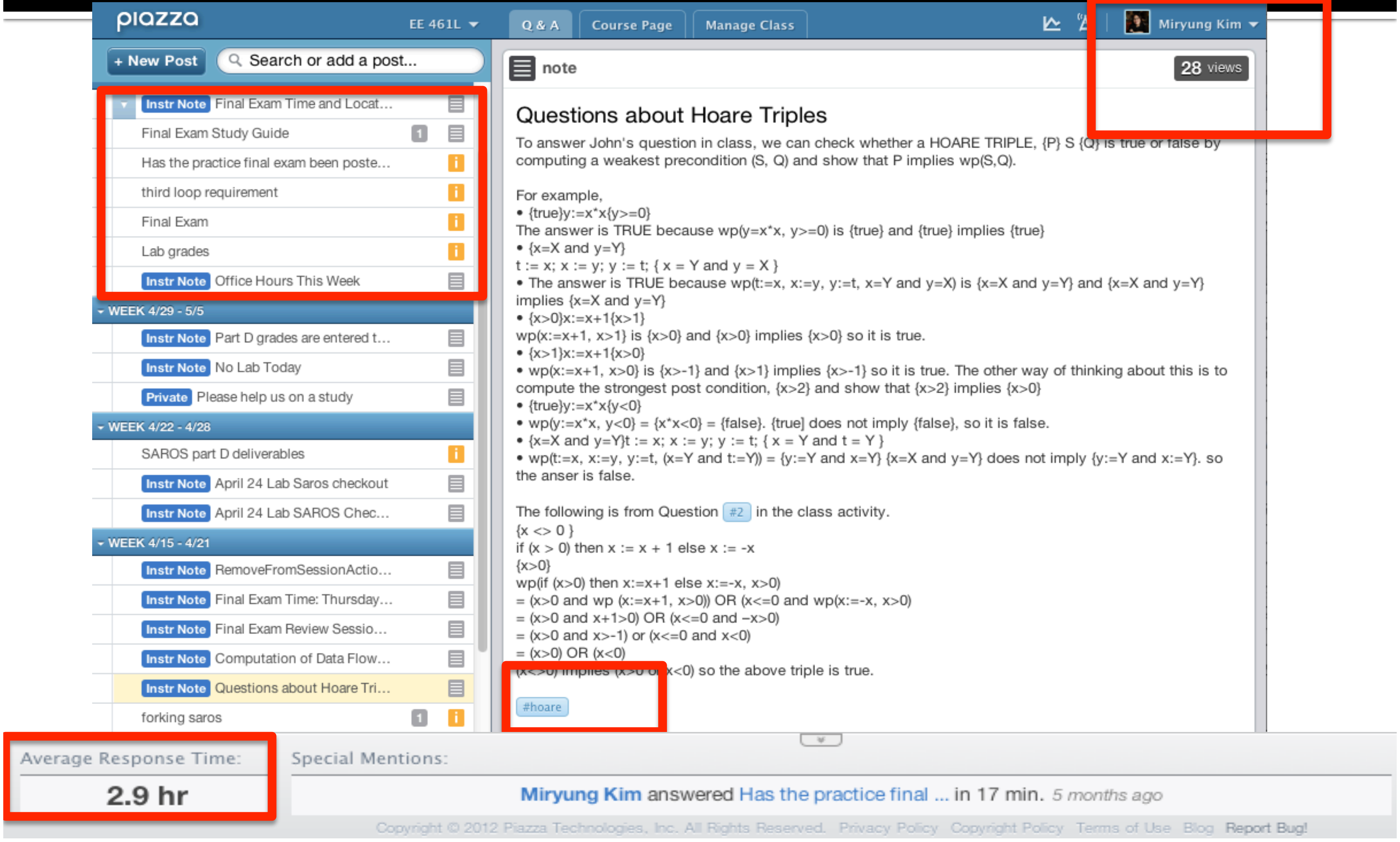
Current UML


StoppableDocumentListener
StoppableDocumentListener(editorManager : EditorManager)
<<realize>>
<<interface>>
IDocumentListener
documentAboutToBeChanged(event : DocumentEvent)()
Wizard
pages : ArrayList<IWizardPage>
windowTitle : String
EditorManager
sarosSession : ISarosSession
documentListener : StoppableDocumentListener
textAboutToBeChanged(offset : int, text : String, replaceLength : int, document : IDocument)
ISarosSession
getLocalUser() : User
1*...
ConfigurationWizard
addPages()
User
UserConnectionState : enum
isLocal() : Boolean
isAway() : Boolean
getOfflineSeconds() : Integer
getConnectionState() : ConnectionState
WizardUtils
openWizard(wizard : final Wizard, initialSize : final Point) : Integer
openSarosConfigurationWizard() : ConfigurationWizard
openAddXMPPAccountWizard() : AddXMPPAccountWizard
ConfigurationSettingsWizardPage

Extension in UML


StoppableDocumentListener
StoppableDocumentListener(editorManager : EditorManager)
<<realize>>
<<interface>>
IDocumentListener
documentAboutToBeChanged(event : DocumentEvent)()
StatisticsWizardPage
Wizard
pages : ArrayList<IWizardPage>
windowTitle : String
EditorManager
sarosSession : ISarosSession
documentListener : StoppableDocumentListener
userMetrics : LocalUserMetrics
textAboutToBeChanged(offset : int, text : String, replaceLength : int, document : IDocument)
ISarosSession
getLocalUser() : User
1*...
User
UserConnectionState : enum
isLocal() : Boolean
isAway() : Boolean
getOfflineSeconds() : Integer
getConnectionState() : ConnectionState
getOnlineSeconds() : Integer
LocalUserMetrics
timeOnline : Integer
charsModified : Integer
LinesAdded : Integer
LinesRemoved : Integer
WizardUtils
openWizard(wizard : final Wizard, initialSize : final Point) : Integer
openStatisticsWizard() : StatisticsWizard
StatisticsWizard
addPages()

On-Line Education Technologies: Piazza



- Instr Note Final Exam Time and Locat...
- Final Exam Study Guide 1
- Has the practice final exam been poste...
- third loop requirement
- Final Exam
- Lab grades
- Instr Note Office Hours This Week

WEEK 4/29 - 5/5
- Instr Note Part D grades are entered t...
- Instr Note No Lab Today
- Private Please help us on a study

WEEK 4/22 - 4/28
- SAROS part D deliverables
- Instr Note April 24 Lab Saros checkout
- Instr Note April 24 Lab SAROS Chec...

WEEK 4/15 - 4/21
- Instr Note RemoveFromSessionActio...
- Instr Note Final Exam Time: Thursday...
- Instr Note Final Exam Review Sessio...
- Instr Note Computation of Data Flow...
- Instr Note Questions about Hoare Tri...
- forking saros 1

note — 28 views

Questions about Hoare Triples

To answer John's question in class, we can check whether a HOARE TRIPLE, {P} S {Q} is true or false by computing a weakest precondition (S, Q) and show that P implies wp(S,Q).

For example,
• {true}y:=x*x{y>=0}
The answer is TRUE because wp(y=x*x, y>=0) is {true} and {true} implies {true}
• {x=X and y=Y}
t := x; x := y; y := t; { x = Y and y = X }
• The answer is TRUE because wp(t:=x, x:=y, y:=t, x=Y and y=X) is {x=X and y=Y} and {x=X and y=Y} implies {x=X and y=Y}
• {x>0}x:=x+1{x>1}
wp(x:=x+1, x>1} is {x>0} and {x>0} implies {x>0} so it is true.
• {x>1}x:=x+1{x>0}
• wp(x:=x+1, x>0} is {x>-1} and {x>1} implies {x>-1} so it is true. The other way of thinking about this is to compute the strongest post condition, {x>2} and show that {x>2} implies {x>0}
• {true}y:=x*x{y<0}
• wp(y:=x*x, y<0} = {x*x<0} = {false}. {true] does not imply {false}, so it is false.
• {x=X and y=Y}t := x; x := y; y := t; { x = Y and t = Y }
• wp(t:=x, x:=y, y:=t, (x=Y and t:=Y)) = {y:=Y and x=Y} {x=X and y=Y} does not imply {y:=Y and x:=Y}. so the anser is false.

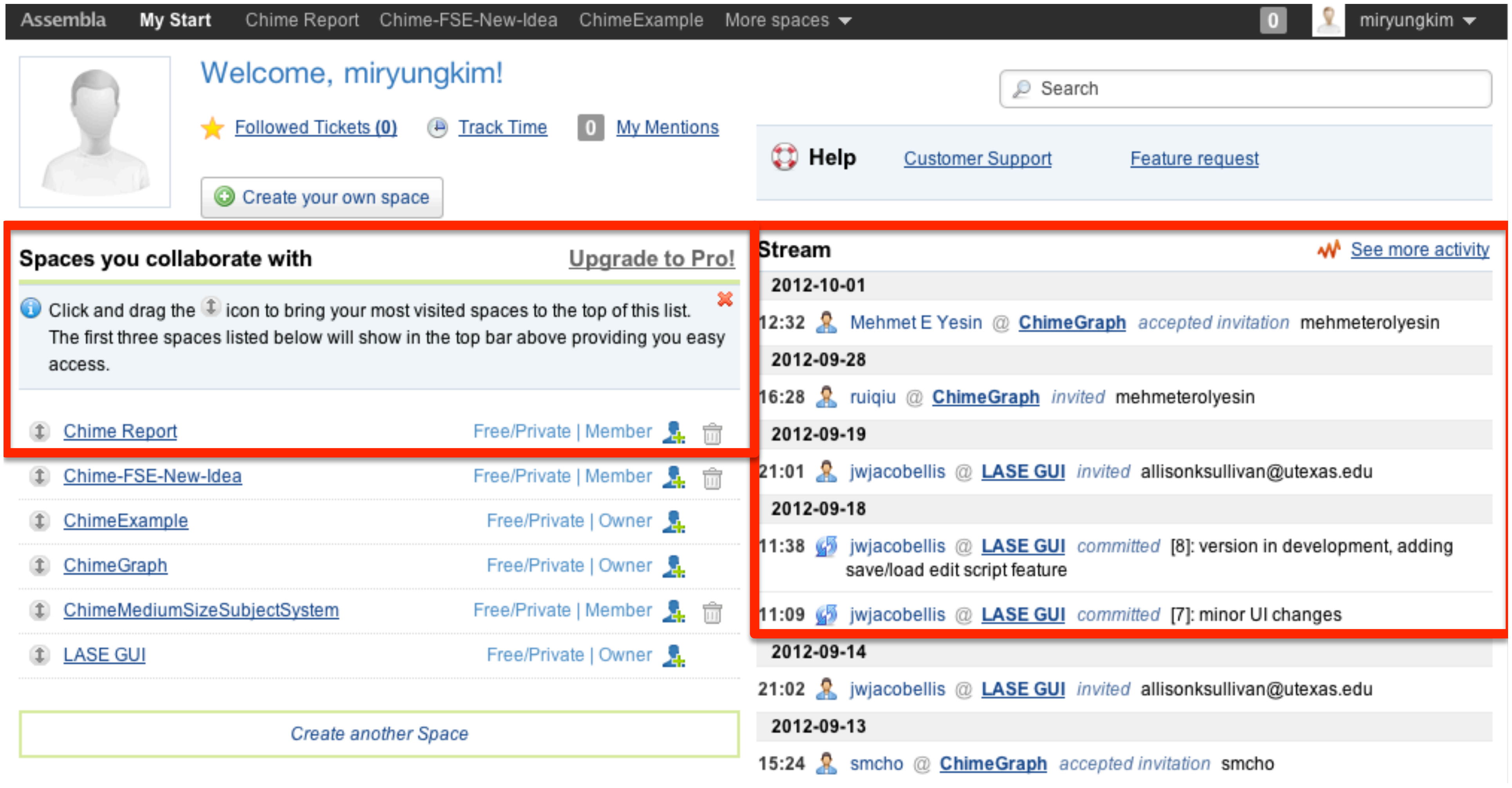
The following is from Question #2 in the class activity.
{x <> 0 }
if (x > 0) then x := x + 1 else x := -x
{x>0}
wp(if (x>0) then x:=x+1 else x:=-x, x>0)
= (x>0 and wp (x:=x+1, x>0)) OR (x<=0 and wp(x:=-x, x>0)
= (x>0 and x+1>0) OR (x<=0 and –x>0)
= (x>0 and x>-1) or (x<=0 and x<0)
= (x>0) OR (x<0)
(x<>0) implies (x>0 or x<0) so the above triple is true.

#hoare

Average Response Time: 2.9 hr

Special Mentions: Miryung Kim answered Has the practice final ... in 17 min. *5 months ago*

On-Line Education Technologies: Assembla



Welcome, miryungkim!

Spaces you collaborate with

Click and drag the icon to bring your most visited spaces to the top of this list. The first three spaces listed below will show in the top bar above providing you easy access.

Chime Report	Free/Private | Member
Chime-FSE-New-Idea	Free/Private | Member
ChimeExample	Free/Private | Owner
ChimeGraph	Free/Private | Owner
ChimeMediumSizeSubjectSystem	Free/Private | Member
LASE GUI	Free/Private | Owner

Stream

2012-10-01

12:32 Mehmet E Yesin @ ChimeGraph *accepted invitation* mehmeterolyesin

2012-09-28

16:28 ruiqiu @ ChimeGraph *invited* mehmeterolyesin

2012-09-19

21:01 jwjacobellis @ LASE GUI *invited* allisonksullivan@utexas.edu

2012-09-18

11:38 jwjacobellis @ LASE GUI *committed* [8]: version in development, adding save/load edit script feature

11:09 jwjacobellis @ LASE GUI *committed* [7]: minor UI changes

2012-09-14

21:02 jwjacobellis @ LASE GUI *invited* allisonksullivan@utexas.edu

2012-09-13

15:24 smcho @ ChimeGraph *accepted invitation* smcho

Student Feedback

- “I think (in class) activities are great, they help me a lot to understand the concepts that are taught in class.”
- “Dr. Kim has designed a highly interactive course. The skills we learn during labs have helped me become stronger in my software skills.”
- “It is useful to learn different tools. The self-paced tutorial is a nice way to do it.” “It is also good to feel like I am in an open forum and can ask for help at any time”
- “It should be a sophomore level class because this is material you need before internships! I had a hard time learning this on the job.”
- “I think this course can benefit from having weekly or biweekly homework.”
- “Reading academic papers would be cool.”

Lessons Learned

- Provide early and frequent feedback
- Incentivize rather than offer unsolicited advice
- Clear communication on course management / expectations
- Bring out creativity and ownership for class projects

Conclusions

- 461L intends to provide hands-on experience in working with large software systems and to prepare for professional careers in SE.
- In class activities and self-paced tutorials helped students to engage in highly abstract subject matters.
- Early and frequent feedback through tests and assignments and clear communications on course expectation are needed.

Acknowledgment

- Software Engineering Area Faculty in ECE
- Academic Development Fund from Cockrell School of Engineering
- ECE Department
- TAs: Evan Grim, Rui Qui, and Kevin Boos