EECS 22L: Software Engineering Project in C Language

Lecture 7

Rainer Dömer

doemer@uci.edu

The Henry Samueli School of Engineering
Electrical Engineering and Computer Science
University of California, Irvine

Lecture 7: Overview

- Course Administration
  - Midterm course evaluation
  - New project setup
- Project 2
  - Focus points
  - Introduction
  - Application specification

Course Administration

- Midterm Course Evaluation: Results
  - Participation
    - 33 out of 95 students (34.74%)
    - ➢ Not really representative!
  - Student Feedback
    - Mostly very positive and encouraging
      - Overall letter grade: 19 “A”, 11 “A-”, 1 “B” (and 2 “N/A”?)
    - Specific comments
      - ➢ **`MidtermEvaluation_Report.pdf`**

Course Administration

- New Teams
  - New teams have been formed for Project 2
    - Almost all preferences have been met
      - ➢ A number of teams stay together unchanged
    - Very few cyclic or contradictory conflicts
  - ➢ 26 new teams, 5 or 6 members each
    - ➢ See individual team emails
- New Team Accounts
  - All team accounts have been cleared (all files deleted)
  - New passwords will be distributed in lab sessions by TAs
    - Use lab sessions this week to set up the new team account

Project 2

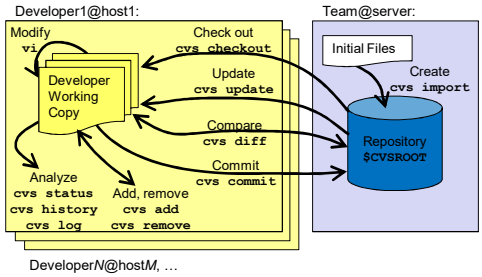
- New Focus Points
  - Graphical User Interface (GUI)
    - mandatory
  - Version control with CVS
    - mandatory
  - Testing
    - Unit test
    - System test
  - Client Server Application
    - TCP/IP communication via sockets

Project 2 Focus: GUI

- Graphical User Interface (GUI)
  - Differences to command-line interface
    - Input through events
    - Output through 2D pixel interface
    - Event-based main control loop
  - Available GUI libraries
    - GTK: GNOME or GIMP Toolkit
      - Recommended choice!
      - To be introduced in an upcoming lecture!
      - **`http://www.gtk.org/`**
    - SDL: Simple DirectMedia Layer
      - Refer to Lecture 4
      - **`http://www.libsdl.org/`**

Project 2 Focus: CVS

- Version Control using CVS
  - Source code management
    - For successful team work, version management is critical
    - Once properly set up, it's all a matter of '**`cvs update`**' and '**`make clean all`**'



  - ➢ Refer to Lecture 3 and online documentation

Project 2 Focus: Testing

- Perform Unit Tests and System Test
  - Test each module/component individually
    - With specific test data
    - Automatically
  - Run each module/component in debug mode
    - **`make test`**
  - ➢ Top-level **`Makefile`** should provide targets to run tests for each unit and the entire system
  - Unit tests:
    - **`% make test_gui`**
    - **`% make test_client`**
    - **`% make test_server`**
  - System test:
    - **`% make test`**

Project 2 Focus: Testing

- Perform Automated Unit Tests:
  - Test each module individually (with specific test data)
  - Use **`make test`** to run each module in debug mode
    - Example: Student records (see EECS 22, Lectures 14 ff.)

```
/* Student.c: maintaining student records */
...
#ifdef MAIN  /* test the student record functions */
int main(void)
{   STUDENT *s1 = NULL;
    s1 = NewStudent(1001, "Jane Doe", 'A');
    PrintStudent(s1);
    [...]
    return 0;
} /* end of main */
#endif /* MAIN */
/* EOF */
```

```
% vi Student.c
% make Student
gcc -Wall -ansi -g -c Student.c -o Student.o
gcc -DMAIN -Wall -ansi -g Student.c Student.o -o Student
```


Project 2 Focus: Client-Server App

- Not a stand-alone program any more…

➢ Application with Internet based communication!

- Client Server Software Architecture
  - Client App: sends requests to the server
  - Server Program: processes clients' requests
- Networked Inter-Process Communication
  - Internet protocol (IP)
  - TCP/IP communication via sockets
  - ➢ *Details will be introduced in Lecture 8*

Project 2

- Introduction
  - *Chat Tool*
    - Instant Text Messaging Application
  - A. User App
    - Register user's account name, password, IP address, etc.
    - Login, uppdate user's status: available, idle, or offline
    - List available users, add/delete contacts
    - Chat with text messages, images, …
  - B. Provider Service
    - Management of user accounts and contact information
    - Service users' login and requests
    - Maintain users' status, IP addresses, port numbers, …

Project 2: Software Specification

- Basic functions that are required:
  - Registration with provider (user account management)
  - Login
  - Add/delete users to contact list
  - View the status of other users in the contact list
  - Chat with online users in the contact list
  - Accept/reject non-contact users' invitations
  - Start/leave/restart chats
  - Keep track of chat threads, highlight unread messages

➢ *Text-based instant messaging with selected contacts!*

Project 2: Software Specification

- Advanced options that are desirable:
  - Transmit files, pictures, and/or videos
  - Save and display pictures
  - Apply filters to the pictures and/or videos
  - Customize fonts, colors, and other settings
  - GUI for the provider
  - Save a log of a conversation
  - Retrieve chat history
  - Message seen status
  - User typing status

  ➢ *Any other options that make the chat application irresistible to buy!*

Project 2: Software Specification

- Application Specification from the User's Perspective
  - Discussion on white board
  - Q & A