EECS22L Discussion Week 2

Outline

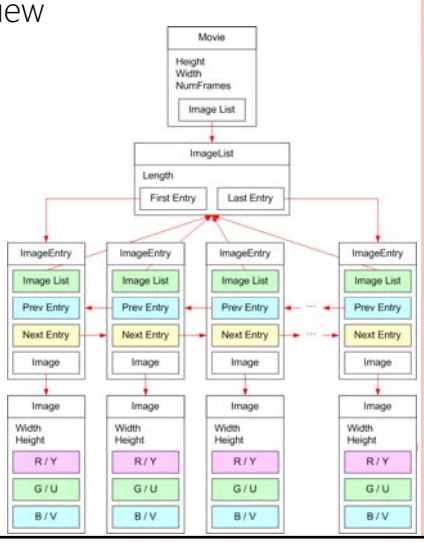
- Feedback of the first submission
- Second Submission
  - Chess_SoftwareSpec.pdf
- CVS review

Second Submission

- Name of the deliverable(s):
  - Chess_SoftwareSpec.pdf
- Due date:
  - Jan 19, 12pm (noon)
- Requirement:
  - Please refer to Grading Criteria on the course website

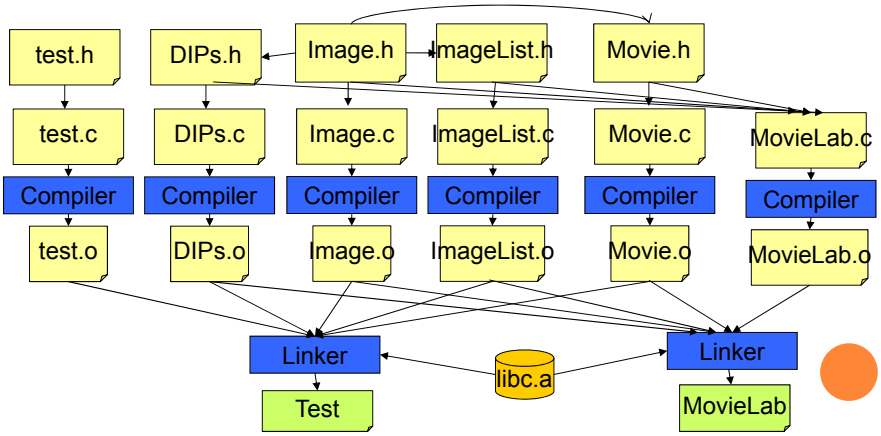
Chess_SoftwareSpec.pdf (using MovieLab as the example)

- Software Architecture Overview
  - Main data type and structure
  - Example:
    - Why double link list for a movie ?
    - Why R/G/B in struct IMAGE ?
  - How to represent Player?
  - How to represent PieceType?
  - How to represent a chess board?
  - How to represent a chess piece?
  - How to represent a position?
  - How to represent a movie?
  - How to represent a move list?
  - Please refer to handout 4.2



CHESS_SOFTWARESPEC.PDF (USING MOVIELAB AS THE EXAMPLE)

- Software Architecture Overview
  - Major software components

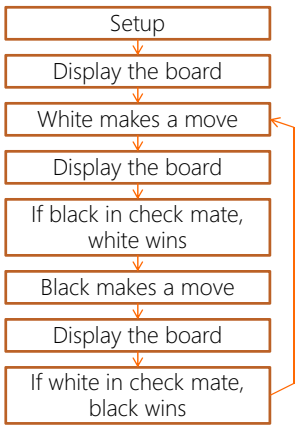


CHESS_SOFTWARESPEC.PDF (USING MOVIELAB AS THE EXAMPLE)

- Software Architecture Overview
  - Module interface
  - Please refer to handout 4.2
  - Example:
  - ```
    unsigned char GetPixelR(IMAGE *image,
                            unsigned int x,
                            unsigned int y);
    /* function description ...*/
    ```
  - ```
    ILIST *NewImageList(void);
    /* function description ...*/
    ```
  - ```
    void AppendImage(ILIST *l, IMAGE *image);
    /* function description ...*/
    ```
  - ```
    void ReverseImageList(ILIST *l);
     /* function description ...*/
    ```
  - ```
    ...
    ```
    ```

CHESS_SOFTWARESPEC.PDF

- Software Architecture Overview
  - Overall program control flow
  - Please refer to the handout 4.3
  - Illustrate the flow with a control flow graph in your SoftwareSpec
  - Yours should come with more detail and description than this example.



CHESS_SOFTWARESPEC.PDF

- Unlike the installation in UserManual, in this one you should describe the comment you use to build, compile, execute, clean, and test your program.
- Installation
  - How to compile your program
    - make
  - How to run your program
    - Ex: % ./bin/chess
  - How to run the test
    - make test
  - How to clean the working directory
    - make clean

CHESS_SOFTWARESPEC.PDF

- Think carefully of the features you want to implement, as well as the modules, APIs, data structure of them.
- Describe the detail as much as you can in the SoftwareSpec
- Partition the tasks (Team member responsibilities)
- Be sure to reference 3rd party images, etc.

CVS-SET ENVIRONMENT VARIABLE

- Set the environment variable CVSROOT
  - setenv CVSROOT :ext:account@machine:path

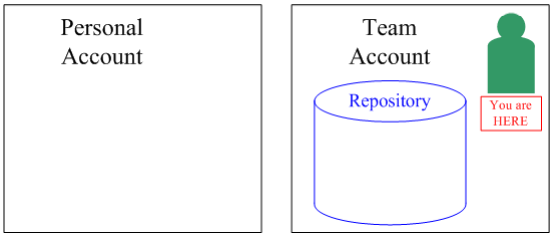
example : set CVSROOT to /users/ugrad2/2013/winter/teamX/cvs_rep
> setenv CVSROOT :ext:teamX@ladera.eecs.uci.edu:/users/ugrad2/2013/winter/teamX/cvs_rep

- P.S. the words in RED is the information you have to provide.
  In this example, account = teamX
  machine = ladera.eecs.uci.edu
  path = /users/ugrad2/2013/winter/teamX/cvs_rep

CVS-CREATE REPOSITORY

- Create a repository
  - cvs –d ~path init

example : create repository at /users/ugrad2/2013/winter/teamX/cvs_rep
> (login to teamX)
> cvs –d ~/cvs_rep init

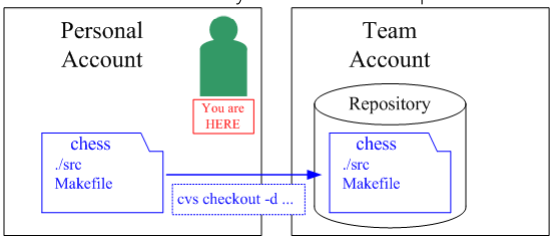


CVS-IMPORT PROJECT

- Import project to repository
  - cvs import –m "message" project vender version

example : put project "chess" to repository
> (login to personal account and cd into your chess directory)
> (set environment variable CVSROOT if you have not done it yet. See page 2)
> cvs import –m "chess alpha" chess chewei start

P.S. after import the project into the repository you can delete your local directory, but better to do that after you confirm it is imported successfully

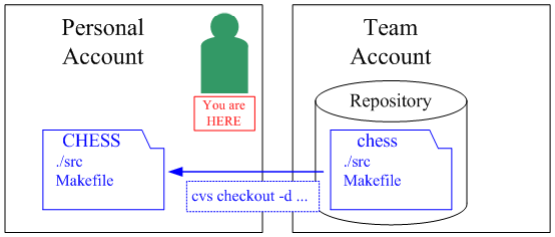


CVS-CHECKOUT PROJECT

- Checkout a project from repository to local
  - cvs checkout –d path project

example : checkout project "chess" from repository to directory "CHESS"
> (login to your personal account)
> (set environment variable CVSROOT if you have not done it yet. See page 2)
> cvs checkout –d CHESS chess

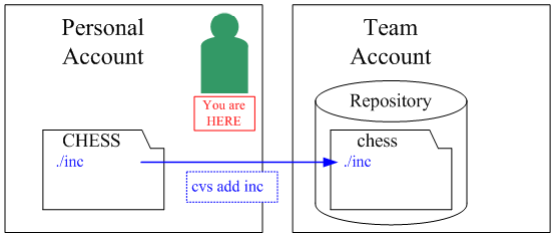
P.S. [–d path] is just optional.



CVS-ADD NEW DIRECTORY

- Add a new empty directory to the repository
  - cvs add directory

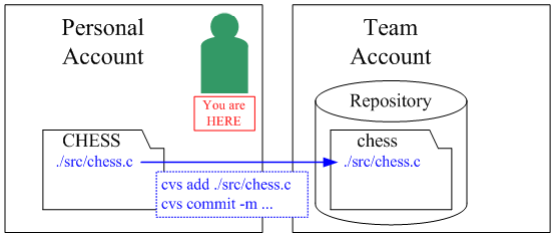
example : add directory "inc" to the repository
> (login to your personal account and cd into your cvs checkout directory)
> (set environment variable CVSROOT if you have not done it yet. See page 2)
> cvs add inc



CVS-ADD NEW FILE

- Add a new file to the repository
  - cvs add file
    cvs commit –m "message"

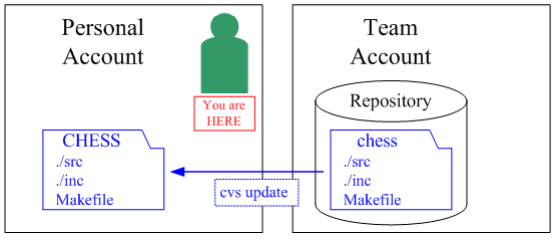
example : add file "./src/chess.c" to the repository
> (login to your personal account and cd into your cvs checkout directory)
> (set environment variable CVSROOT if you have not done it yet. See page 2)
> cvs add ./src/chess.c
> cvs commit –m "put chess.c in repository"



CVS-UPDATE PROJECT

- Update your working directory
  - cvs update

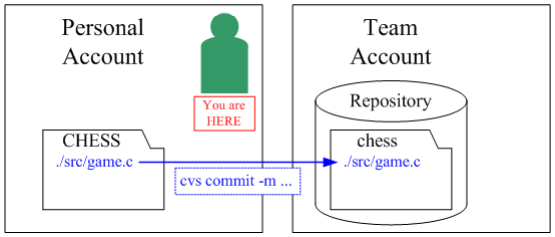
example : update working directory "CHESS" from repository
> (login to your personal account and cd into your cvs checkout directory)
> (set environment variable CVSROOT if you have not done it yet. See page 2)
> cvs update



CVS-COMMIT

- Commit changes to CVS repository
  - cvs commit –m "message" file

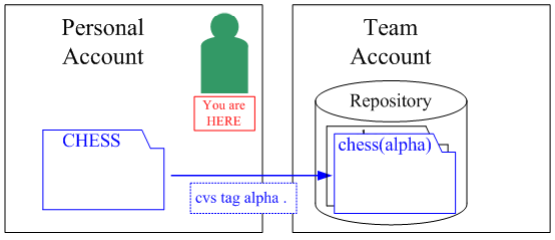
example : commit change in file ./src/game.c to the repository
> (login to your personal account and cd into your cvs checkout directory)
> (set environment variable CVSROOT if you have not done it yet. See page 2)
> (modify ./src/game.c and save the change)
> cvs commit –m "game.c modification" ./src/game.c



CVS-TAG PROJECT

- Tag the project in CVS repository
  - cvs tag version

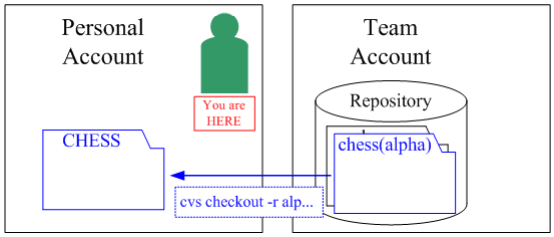
example : create branch "alpha" for your chess program
> (login to your personal account and cd into your cvs checkout directory)
> (set environment variable CVSROOT if you have not done it yet. See page 2)
> cvs tag alpha .



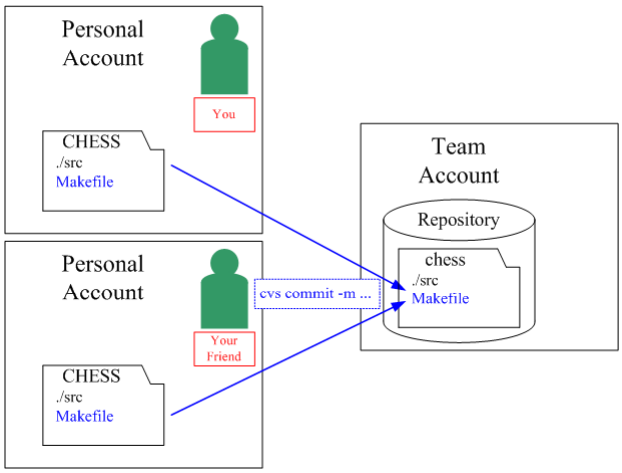
CVS-CHECKOUT TAGGED PROJECT

- Checkout a tagged project
  - cvs checkout –r version project

example : checkout branch "alpha" of chess from CVS
> (login to your personal account and cd into your cvs checkout directory)
> (set environment variable CVSROOT if you have not done it yet. See page 2)
> cvs checkout –r alpha chess



CVS-RESOLVE CONFLICT



CVS-RESOLVE CONFLICT (CONT)

- Resolve the conflict

example : you and your friend both modify chess.l and commit it to CVS.
your friend commited it before you did.
> (you will not be able to commit the your change due to the conflict)
> cvs update
> vi chess.l
> (find the symbols inserted by CVS and remove it)
> (the symbols will be (<<<<<<< filename, ========, and >>>>>>> version)
> (example :
<<<<<<< chessl. ← Inserted by CVS
Your Documentation
======= ← Inserted by CVS
Your friend' s Documentation
>>>>>>> 1.2 ← Inserted by CVS
)
> cvs commit –m "conflict resolved" chess.l