cs121 - software development
project organization 1

alexandre r.j. francois
visiting associate professor of computer science
outline

deliverables
logistics
schedule
teamwork
deliverables

collective - class
  design document (10%)
  requirements document (10%)
  software design description document (10%)
  website and publicity material (5%)

collective - team
  Goals and subgoals reports (5%)
  Code with documentation and tests (15%)
  Reviews (packages and results) (5%)

individual
  portfolio (40%)
create a Google Doc
  title: “CS121 portfolio - <your name>”
  use the name you like to be called...
invite alexandrefrancois@gmail.com as collaborator
at the top, include some info about you
  picture, name, class, major, your CS username
document your project activities at least once per week
  really add on everything you do for the project
this will be used for:
  monitoring your involvement during the semester
determine your individual project grade
these documents will not be public by default!
Trac repository

https://www.cs.hmc.edu/trac/cs121fa2010

present key information, such as

  - project goals
  - milestones
  - tickets

(documentation will be in the form of a separate site created with Doxygen)
team organization

who does what?
  collaborative teams (concurrent vs. parallel)
  fluid membership (?)

communication and documentation are key!

management vs. leadership
  possible distinction:
    management = do things right
    leadership = do the right thing
  common sense helps, but not enough!
at every meeting

begin by having each member/team report progress from the week before

discuss issues, risks, plans

finish by having each member/team indicate his/her action items or tasks for the coming week

assign responsibility for key roles:

  moderator

  primary author of report
at every meeting

everyone should speak
not speaking can indicate lack of interest, lack of contributions, general disinterest
the team leader (if applicable) should not do all of the talking for everyone
everyone should be taking notes throughout the meeting
everyone should later contribute to the written documentation of the meeting
take notes even if you never read them again. It helps maintain attention
communication pathways

change proposal and control protocols
design documents
requirements document
software architecture
development plan
milestone schedules
top 10 risk list
communication tools

collaborative authorship
   GoogleDocs, Wiki
version control systems (code)
   CVS, Subversion (SVN), git
integrated tools
   Trac, sourceforge.net
architectural framework / design language
   SAI/MFSM
   UML
reasons teams fail

methodology shortcuts
worthless methodologies
loss of scope
unrealistic schedule
poor estimations
too much enthusiasm