Collaboration in Software Engineering: A Roadmap

Jim Whitehead
Who Am I?

• Created and led Internet Engineering Task Force working group on Web Distributed Authoring and Versioning
  ▶ WebDAV, DeltaV standards
  ▶ Built on top of HTTP

• Brought people together to collaborate (working group)
  ▶ …on the creation of a collaboration protocol (WebDAV)
  ▶ …that was infrastructure for a coordination protocol (DeltaV)
  ▶ WebDAV is now being used as infrastructure for CalDAV
    • Calendar access protocol
“The future is here, it’s just not evenly distributed yet.”
- Attributed to novelist William Gibson

Predicting the future by studying the present
Why collaborate?

• Human limitations drive need to collaborate
  ▶ At high levels of abstraction, we are slow and error-prone
  ▶ Hence, must work together to create large projects

• Working together creates new set of problems
  ▶ Natural language ambiguity
  ▶ Human memory cannot recall all project details
  ▶ Cannot track everyone’s activities in a large group
  ▶ Need to converge on a single architecture and design
Goals for Software Engineering Collaboration

- Establish scope and capabilities of a project
  ▶ What are we doing?
- Converge on architecture and design
  ▶ How are we doing it?
- Manage dependencies among activities, artifacts, and organizations
  ▶ Who is doing what, when?
- Reduce dependencies among engineers
  ▶ Reduce need for collaboration
- Identify, record, and resolve errors
  ▶ Engineers make errors. Deal with it.
- Record organizational memory
  ▶ Record what has been decided and done.
Meeting the goals: communication

- Software engineers have adopted every mainstream communication technology
  - Telephone, teleconferences, email, voice mail, discussion lists, Web, instant messaging, text messaging, video conferences, voice over IP
  - Useful at every stage of a project’s lifecycle
- Face to face
  - Meetings, informal discussions in hallways, doorways, and offices
- In general, this communication is unstructured
- Similar to collaboration in other disciplines
  - What makes software engineering collaboration unique?
Model-based collaboration

• What makes software engineering collaboration unique?
  ▶ Software enhanced collaboration around models
  ▶ In contrast to general study of computer supported collaboration which generally lacks this focus on models

• Software engineering collaboration is **model-based**
  ▶ production of new models
  ▶ creation of shared meaning around models
  ▶ elimination of error and ambiguity within models

• Much collaboration around formal and semi-formal artifacts
  ▶ requirements specification, architecture diagrams, UML, source code, bug reports
Future Directions for Collaboration in SE

• Shift to web-based development environments
  ▶ Improving access to artifacts

• Broadening participation in software projects
  ▶ Increasing number of collaborators

• Capturing design rationale as an argumentation structure
  ▶ A new artifact for improved organizational memory

• Use of 3D virtual environments (games)
  ▶ Adoption of a new mainstream communication technology
Shift to Web-based Environments
Historical Progression of Environments

• Up until the mid 90’s:
  ▶ Collaborating collections of tools
  ▶ Locally run, accessing local networked file system
  ▶ Remote collaboration involved synchronization of data to a remote site

• 1999
  ▶ Broad adoption of web-based environments: Sourceforge
  ▶ Mostly network services
    • SCM repository, mailing lists
  ▶ As well as some web-based applications
    • Bug tracking
  ▶ Requirements, design diagrams, editing, compiling: all still local
eRequirements, Gatherspace
Lorem ipsum dolor sit amet, consectetur adipiscing elit


Aliquam eu felis eu nibh euismod faucibus. Cras blandit facilisis nunc. Ut neque.

Today: Web 2.0 and Environments

- Requirements
  - Gatherspace, eRequirements
- UML diagrams
  - Gliffy supports UML diagrams, among many
- Inspections
  - Multiple web-based tools (1997-)
- Editors
  - Google Docs & Spreadsheets
    - Potential for rich web-based editors
Moving IDEs to the Web

• To enhance collaboration, and reap administrative benefits:
  ▶ Development environment capabilities will increasingly shift to the web
  ▶ Opportunity exists to create first complete lifecycle web-based development environment

• Desktop-based tools will not disappear
  ▶ Eclipse, Visual Studio, etc.
  ▶ Instead, will be tightly integrated into web-based environments
    • More than just SVN integration
Integrating Desktop and Web IDEs

- Need to create interface standards to allow within-application access to web-based data
  - Bug tracking
  - Requirements
  - Design diagrams and design rationale
  - Unit and system test results
  - SOA type interfaces?
- Permits delivery of rich assistance services (agent-based SE)
List Your Stuff in Marketplace

Looking for a couch? Trying to get rid of your textbooks? Selling your car? List what you have or want in Facebook Marketplace.

News Feed

Chris Ward added new photos. 2:36pm

Thé Party
28 photos
by Chris Ward
Location: Porter Quad

Belki Grinter and Mark Stringer are now friends. 10:24am

Chris Ward was tagged in an album. 3:41am

Flash Dance and Tea Party
38 photos
by Roby A. Behrens

Potential directions for Web-based IDEs

• Integration with network-of-relationship sites
  ▶ Facebook, MySpace, LinkedIn
  ▶ Improved team awareness and cohesion
    • IBM Dogear: bookmarking among social network of employees
  ▶ Expertise finding

• Integration with web-based code search engines
  ▶ Searching your own code (privately), or integrating found OSS code (with compatible license)
    • Krugle, Google Code Search, Splunk
  ▶ Integration with normal search engines too

• Access to analysis tools offered as a service
  ▶ SOA model for static analysis, test case generators, etc.

• Requirements integrated with CRM
  ▶ Trace requirement back to individual customer
Broadening Project Participation
Broadening Project Participation

• Many types of software lock-in their users
  ▶ Organizations dependent on this software have strong incentive to ensure it meets their needs
• Currently:
  ▶ Customers consulted during requirements…
  ▶ … then get to give feedback during beta testing
• Opportunity:
  ▶ Develop means to engage customers during all phases
  ▶ Allow them to actively ensure needs are met
Challenge: What to control

- Most organizations do not want to cede total control over software to customers
  - Not complete open source
- Selective opening of projects
  - Choose portions of the code to make available
    - No need to reveal trade secrets, distinctive IP
    - Foreign language translations
  - Give customer participants control over some decisions, but not all
    - Or, reserve veto authority
  - Or, just capture how users are using the software
    - Analyze use logs to improve software
- Ideal
  - Customers add features and fix bugs most relevant to them, increasing satisfaction
Capturing Project Argumentation
Capturing Project Argumentation

• Even experienced engineers disagree on portions of the requirements and design of large systems
  ▶ “Design for change” means “predict the future”
    • An argument ensues

• Architecture and design can be viewed as argumentative processes
  ▶ Resolution of differences of prediction and interpretation among skilled practitioners to develop system structure
  ▶ Since only one model can prevail, architecture and design are simultaneously cooperative and competitive
Capturing Project Argumentation Structure

• Need to record why a decision was made
  ▶ Primary argument
  ▶ Supporting facts

• Also important to record alternatives considered, but rejected
  ▶ Why another engineer thought it could be done differently
  ▶ As assumptions change, these alternatives may become relevant

• Argumentation structure gives insight into both the design as-is, as well as design variations
Robert Horn: Can Computers Think?

1 Can Computers Think?
The History and Status of the Debate — Map 1 of 7

Can computers have emotions?

Should we pretend computers will never be able to think?

Does God prohibit computers from thinking?

Can computers understand arithmetic?

Can computers draw analogies?

Are computers inherently disabled?

Can computers be creative?

Is the brain a computer?

Can computers reason scientifically?

Can computers be persons?
Opportunity: Web-based Argumentation

- Design argumentation Wiki
  - Instead of static posters…
    - Dynamic, living argumentation structures
  - Within-Wiki support for argument structure
    - Major/minor points
    - Pro/Con arguments
    - Supporting facts
    - Ability to link to (extract out) requirements
    - Activity awareness, hotspots
  - Analysis of argumentation structure
    - Summarization of arguments
    - Consistency checking
  - Support software variability analysis
    - Why select a specific feature? Here’s why, and why not.
- Some ongoing work in this direction
  - Compendium [Shum et al. 2005]
Collaboration in Networked 3D Virtual Worlds
Software engineers have a long history of adopting novel coordination technologies:

- Better communication
  - Email, instant messaging, text messaging, voice over IP
- Improved presence information
  - Instant messaging
- Multi-way discussions
  - Wikis, collaborative editors
- Change tracking, collision avoidance
  - Software configuration management

If a new communication technology emerges, it is reasonable to expect it to be adopted too.
Networked 3D Virtual Worlds (MMOs)

• Since 2000, a proliferation of Massively Multiplayer Online (MMO) worlds
  ▶ World of Warcraft (WoW), Everquest, Eve Online
  ▶ Millions of users
  ▶ Revenue for WoW in 2006: $471million
  ▶ Player controls avatar, performs quests in the world
    • Towns, cities, dungeons, etc.
  ▶ Communication features:
    • Instant messaging
    • Email-like services
    • Presence (seeing another player’s avatar)
    • Often coupled with use of third-party voice conferencing
  ▶ Most explicitly framed as entertainment spaces
World of Warcraft
Second Life

• MMO that is aimed at primarily adult audience
  ▶ Not framed as a fantasy space
  ▶ Lack of explicit game framing makes professional use possible
  ▶ User generated content
    • Programming language and environment
    • Within world P2P commerce encouraged

• Businesses are exploring use
  ▶ IBM has business unit for 3D virtual worlds
    • Appx. 1000 IBM employees active in Second Life
    • Sun, Dell also have presence
    • Sweden has embassy in Second Life
Scenes from Second Life
Potential Benefits

• A space that can help merge local workers and remote workers

• 3D topology can provide a landscape for structuring online interactions
  ▶ Virtual meeting rooms
    • Can show people gathered for a meeting
  ▶ Project spaces
    • Browsing of project information space
    • Know that people in a vicinity are interested in/working on a project

• Many issues still to address…
Early Days

- Australian Broadcast Commision (ABC) victim of virtual terrorism attack on May 22, 2007
  - Perhaps by “Second Life Liberation Army”
Novel Narrative and Reward Structures

- What makes MMOs so compelling?
  - Narrative structure of quests
  - Compelling reward structure
    - Leveling up
    - Completing a quest
    - Officer of a guild

- An advance in **reward structure engineering**
- Can this reward structure be harnessed for engineering project work?
  - Work in real world on a project might yield benefits in a parallel game world?
  - Apply leveling up idea to professional education?
    - I’m a level 70 software engineer
  - For many young people, these reward structures are as familiar as traditional hierarchic advancement
Students at UCSC collaborated to recreate a scene from Donkey Kong using 8800 Post-It notes on the Engineering 2 building (April 2007)