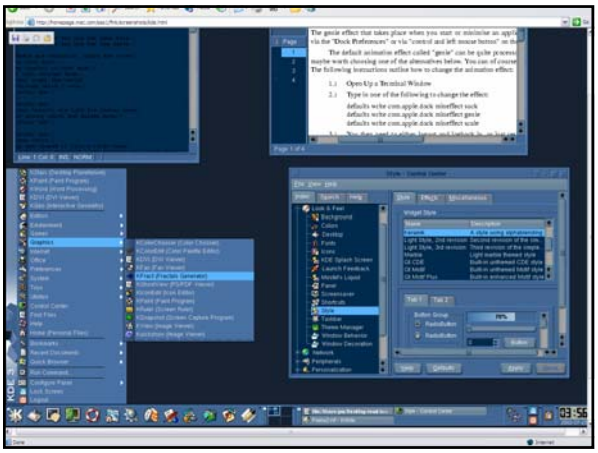
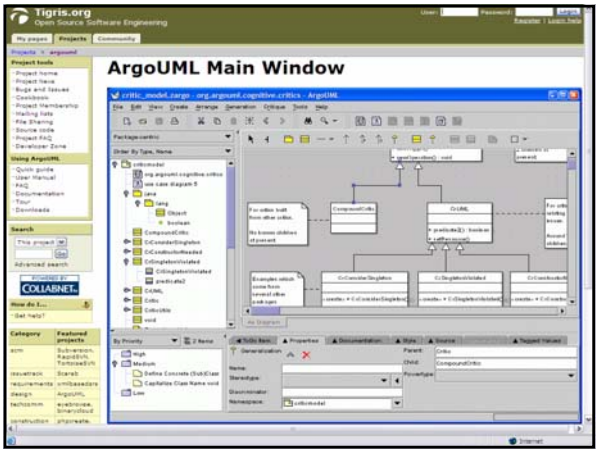
Institute for Software Research
UNIVERSITY OF CALIFORNIA, IRVINE

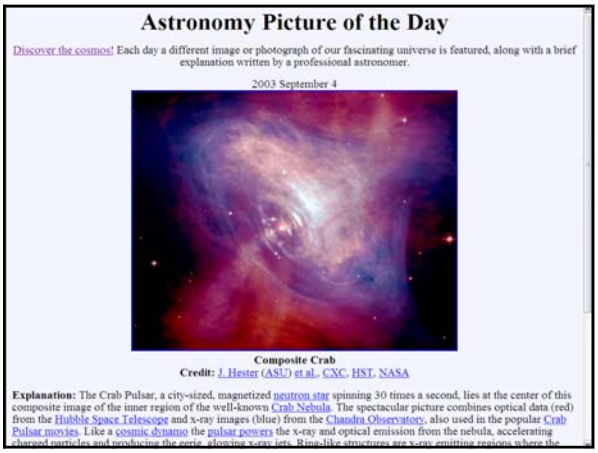
Transforming Organizations through Open Source Software

Walt Scacchi
Institute for Software Research
University of California, Irvine
Irvine, CA 92697-3425 USA
wscacchi@ics.uci.edu
http://www.ics.uci.edu/~wscacchi

15 June 2005







Institute for Software Research
UNIVERSITY OF CALIFORNIA, IRVINE

Overview

- Open source software development
- OSSD business models
- Organizational forms for OSSD
- Governance structures for OSSD
- Transforming organizations via OSS

6

Institute for Software Research
University of California, Irvine

Open Source Software Development

7

Institute for Software Research
University of California, Irvine

What do we know so far about OSSD?

- What is it, what is it not
- Who is participating or investing?
- Recent research results

8

Institute for Software Research
University of California, Irvine

What is OSSD, what is it not?

- Free (as in "freedom") vs. open source software
  - Freedom to access, browse/view, study, modify and redistribute the source code
  - Free is always open, but open is not always free
- F/OSSD is not "software engineering"
  - *Different*: F/OSSD can be faster, better, and cheaper than SE
  - SE "-ilities" may not apply in the same way to OSSD
- F/OSSD involves *more* software development tools, Web resources, and personal computing resources

9

Institute for Software Research
University of California, Irvine

Who is investing in OSS?

- Large corporations/enterprises:
  - IBM-*Eclipse*, Sun-*NetBeans* and *OpenOffice*, HP-*Gelato*, Apple-*Darwin*, Microsoft Research-*Rotor*, etc.
  - Barclays Global Investors, DKW, Merrill Lynch, etc.
  - DoD, DoE, NSF, NIH, NASA, etc.
  - MIT, Stanford, CMU, UC, UMichigan, etc.
- Mid-size corporations:
  - RedHat, Novell, Borland
- Small (start-up) companies:
  - ActiveState (now part of Sophos), Collab.Net, Jabber, Ximian (now part of Novell), JBoss, Compiere, etc.

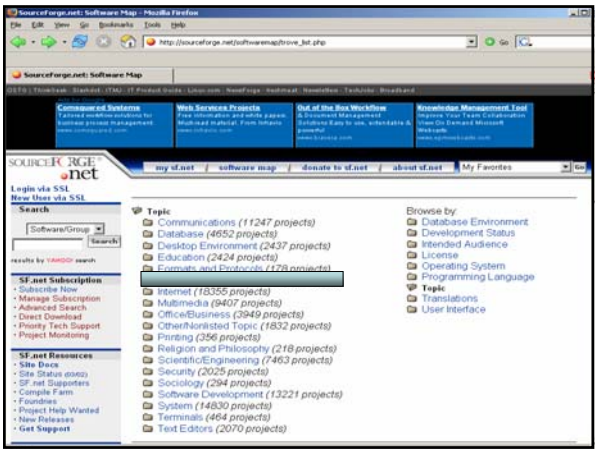
11

Institute for Software Research
University of California, Irvine

Recent research results

- OSS project demographics
- OSSD processes and practices
  - Software requirements and design
  - Architectural practices
  - Configuration management and work practices
  - Evolution dynamics
  - Project management and career contingencies
  - Software technology transfer

12



Institute for Software Research
University of California, Irvine

Findings from F/OSS Studies

- *CIO* Executive Survey--2002-2003:
  - OSS primarily for new system deployments
  - OSS benefits
    - enable lower Total Cost of Ownership
    - lower capital investment
    - greater reliability
  - OSS weaknesses:
    - lack of in-house skills or skills in labor market,
    - lack of vendor support or vendor viability
    - switching costs
- *Observation*: We lack a detailed and predictable understanding of the costs of developing, deploying, and sustaining OSS.

15

Institute for Software Research
University of California, Irvine

Findings from F/OSSD Studies

- Hars and Ou 2002:
  - >60% of F/OSS developers work on 2-10 F/OSS projects (i.e., socio-technical networking widespread)
- Madey, et al. 2003:
  - <5% of OSS projects on SourceForge.net sustained; >90% unsustained, and with less than 2 contributors (i.e., Power Law)
- Nichols and Twidale 2003:
  - Usability of F/OSS systems generally neglected
- Scacchi 2002-2004:
  - Largest F/OSSD projects sustain exponential growth; but most F/OSSD projects fail to grow to any sustainable effort

16

Institute for Software Research
University of California, Irvine

F/OSS Processes for Software Requirements or Design

- F/OSS Requirements/Designs
  - not explicit
  - not formal
- F/OSS Requirements/Designs are embedded within "informalisms"
  - Example OSS informalisms to follow (as screenshot displays)
- F/OSS Requirements/Design processes are different from their SE counterparts.

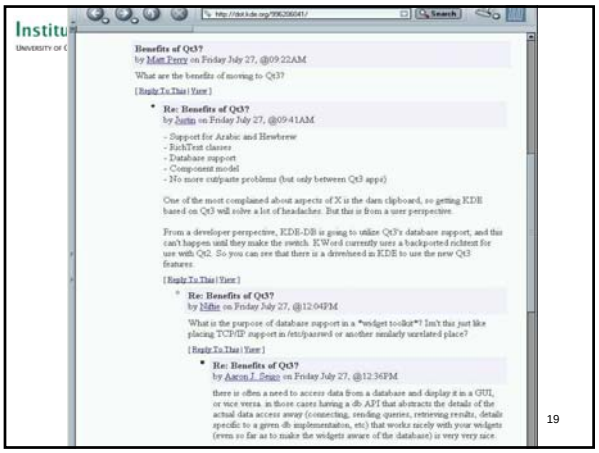
17

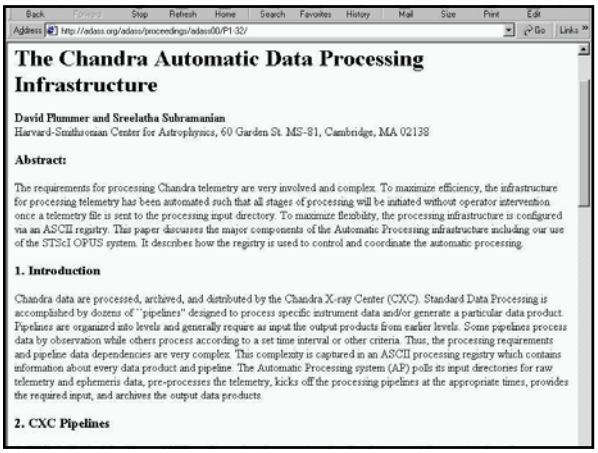
Institute for Software Research
University of California, Irvine

SE vs. F/OSS processes for Requirements

- Elicitation
- Analysis
- Specification and modeling
- Validation
- Communicating and managing
- *Post-hoc* assertion
- Reading, sense-making, accountability
- Continually emerging webs of discourse
- Condensing and hardening discourse
- Global access to discourse

18





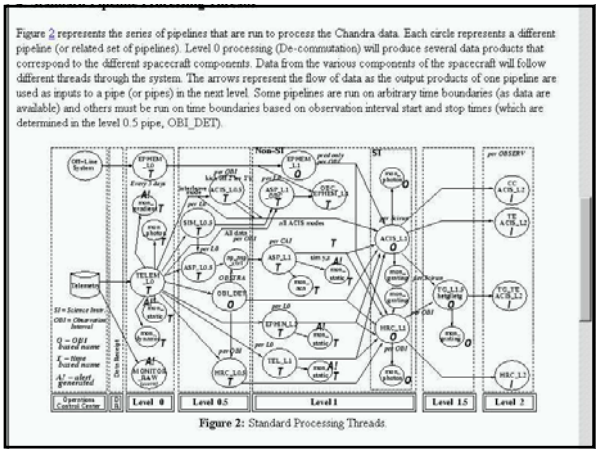


Figure 2 represents the series of pipelines that are run to process the Chandra data. Each circle represents a different pipeline (or related set of pipelines). Level 0 processing (De-commutation) will produce several data products that correspond to the different spacecraft components. Data from the various components of the spacecraft will follow different threads through the system. The arrows represent the flow of data as the output products of one pipeline are used as inputs to a pipe (or pipes) in the next level. Some pipelines are run on arbitrary time boundaries (as data are available) and others must be run on time boundaries based on observation interval start and stop times (which are determined in the level 0.5 pipe, OBI_DET).

Figure 2: Standard Processing Threads.

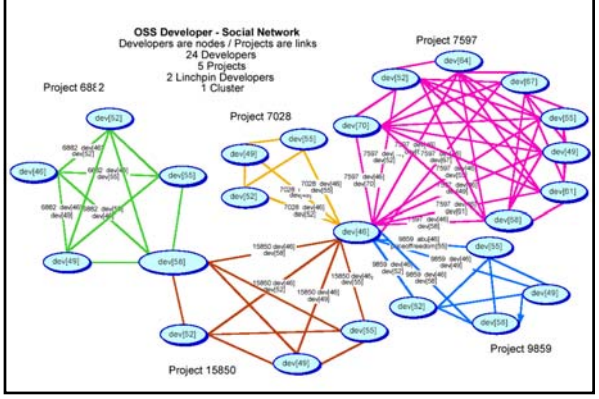
Institute for Software Research
University of California, Irvine

OSS architectural issues

- Source code as API
  - Unpredictable evolution of product line
- *Bricolage*
  - Unanticipated composition of heterogeneous software components
- Social networks and software configurations: Conway's law revisited
  - OSS are socio-technical systems

22

Institute for Software Research
University of California, Irvine



Institute for Software Research
University of California, Irvine

Configuration management and work coordination

- Use CM to coordinate and control who gets to update what part of the system/online artifacts
  - Many F/OSSD projects use **CVS** (single centralized code repository with update locks) and frequent releases (*daily releases* on active projects)
  - Collab.Net and Tigris.org: **Subversion** (CVS++)
  - Apache: Single major release, with frequent "*patch*" releases (e.g., "A patchy server")
  - Linux Kernel: **BitKeeper** (multiple parallel builds and release repositories)—*being replaced by Git*
  - GNU **arch** seeks to develop Free CM unification

24

25

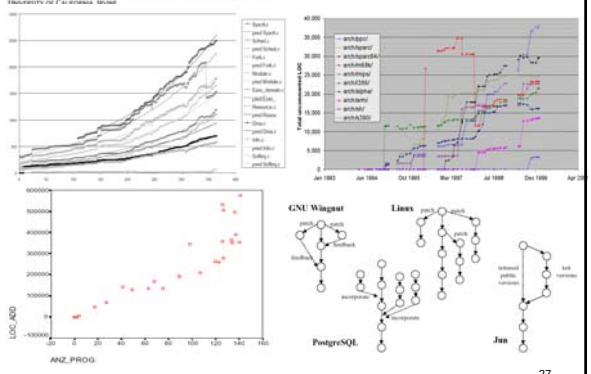
Institute for Software Research

Evolutionary redevelopment, reinvention, and redistribution

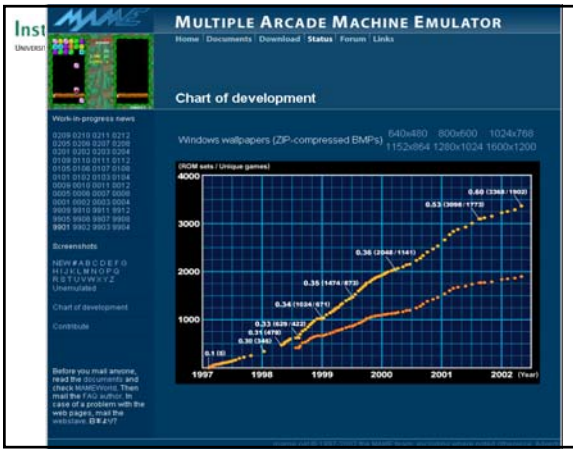
- Overall evolutionary dynamic of F/OSSD is *reinvention*
  - Reinvention enables continuous improvement
- F/OSS evolve through minor mutations
  - Expressed, recombined, redistributed via incremental releases
- F/OSS systems *co-evolve* with their development community
  - Success of one depends on the success of the other
- Closed legacy systems may be *revitalized* via opening and redistribution of their source
  - When enthusiastic user-developers want their cultural experience with such systems to be maintained.

26

Institute for Software Research



27



Institute for Software Research

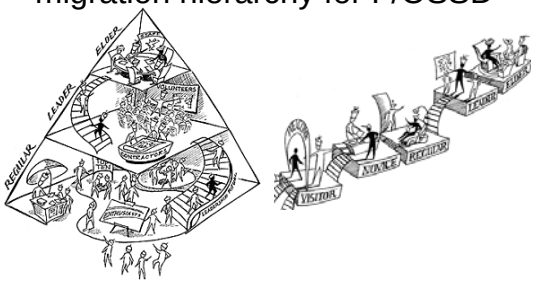
Project management and career development

- F/OSSD projects self-organize as a *pyramid meritocracy* via *virtual project management*
  - Meritocracies embrace incremental mutations over radical innovations
  - VPM requires people to act in leadership roles based on skill, availability, and belief in project community
- F/OSS developers want to learn about new stuff (tools, techniques, skills, etc.), have fun building software, exercise their technical skill, try out new kinds of systems to develop, and/or interconnect multiple F/OSSD projects.

29

Institute for Software Research

A pyramid meritocracy and role migration hierarchy for F/OSSD



(images from A.J. Kim, *Community Building on the Web*, 2000)

30

31

Institute for Software Research
University of California, Irvine

Software technology transfer and licensing

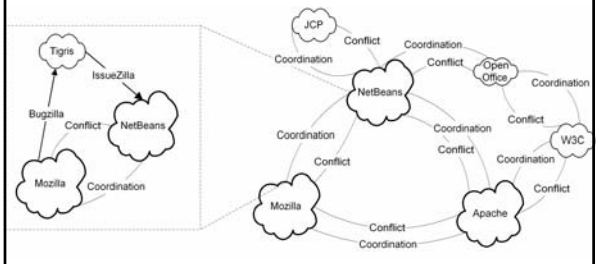
- F/OSS technology transfer from existing Web sites is a *community and team building process*
  - Not (yet) an engineering process
  - Enables unanticipated applications and uses
  - Enables F/OSSD to persist *without* centrally planned and managed corporate software development centers

33

Institute
University of Califo

34

Institute for Software Research
University of California, Irvine



An *OSS Ecosystem*: a socio-technical community and infrastructure for tech transfer and inter-project governance

35

Institute for Software Research
University of California, Irvine

Implications

- F/OSSD is a *community building process*
  - not just a technical development process
  - F/OSS peer review creates a *community of peers*
- F/OSSD processes often iterate *daily* versus infrequent singular (milestone) Software Life Cycle Engineering events
  - F/OSSD: frequent, rapid cycle time (easier to improve) *vs*.
  - SLC: infrequent, slow cycle time (harder to improve)

36

Institute for Software Research
University of California, Irvine

Conclusions regarding OSSD

- Developing F/OSS is *different* than software engineering
  - not better, not worse, but different and new
  - more social, more accessible, more convivial
- F/OSS developers may not seek the goals and benefits from classic software engineering

37

Institute for Software Research
University of California, Irvine

OSSD Business Models

38

Institute for Software Research
University of California, Irvine

Traditional software production business models

- Custom software product engineering
- Cost reduction
- Agile production
- Revenue maximization
- Profit maximization
- Market dominance

M. Cusumano, *The Business of Software : What Every Manager, Programmer, and Entrepreneur Must Know to Thrive and Survive in Good Times and Bad* , Free Press, 2004.

39

Institute for Software Research
University of California, Irvine

OSSD Business Models

- Free software (General Public License)
- Open source (BSD/MIT license)
- Dual license
  - F/OSS and proprietary licenses
- Corporate sponsored (SUN, IBM, Apple, HP)
- Corporate Source (Hewlett-Packard in-house)
- Shared Source (Microsoft Research)
- Community Source (SAKAI, Westwood, etc. in Higher Education)

40

Institute for Software Research
University of California, Irvine

Organizational Forms for OSSD

41

Institute for Software Research
University of California, Irvine

Common OSS organizational forms

- **One-off project** (>90%)
- **Replicated project** (Virtual product line?)
  - 500+ Linux distributions, 200+ OSS content management system projects, 400+ system build/make projects, etc.
- **Meta project**
  - Apache.org
- **Non-profit foundation**
  - Free Software Foundation, GNOME Foundation
- **Corporate sponsored**
  - Sun NetBeans, HP Gelato, IBM Eclipse, Apple Darwin, etc.
- **Consortium** and **Alliance**
  - Open Source Development Lab, Avalanche Cooperative
  - Asterisk VoIP PBX, SugarCRM, OpenAdaptor

42

Institute for Software Research
University of California, Irvine

Governance Structures for OSSD

43

Institute for Software Research
University of California, Irvine

Governance structures

- Collaboration
  - Guidelines, policies, separate concerns
- Leadership and Control
  - Transparency in decision-making
  - Consent in decision-making
  - Conflict Resolution

44

Institute for Software Research
University of California, Irvine

Transforming Organizations via OSS

45

Institute for Software Research
University of California, Irvine

Transforming organizations

- Concepts
- Methods
- Tools
- Case studies

46

Institute for Software Research
University of California, Irvine

Concept

- Transforming organization target from *as-is* condition to *to-be* form via *here-to-there* transformations
- Transformation goal: Aligning organizational form, processes, strategy, and information technology
  - Organizational forms
  - Open source processes
  - Business model strategy
  - Web-centric OSS infrastructure

47

Institute for Software Research
University of California, Irvine

Methods

- Change management
  - Empowering organizational participants to embrace, lead, and affect organizational transformation
- Process engineering
  - Proving participants with the concepts, techniques, tools, and authority to transform their organization in a process-centric manner

48

Institute for Software Research
UNIVERSITY OF CALIFORNIA, IRVINE

Tools (1)

- Organizational process modeling, analysis, simulation, visualization, re-enactment, and redesign
  - Supports capture, description, and application of causal and interrelated knowledge about what can affect software development (from empirical studies).
  - Requires an underlying computational model of process states, actions, plans, schedules, expectations, histories, etc. in order to answer dynamic "what-if" questions.

Institute for Software Research
UNIVERSITY OF CALIFORNIA, IRVINE

Tools (2)

Organizational transformation *collaboratory*

- An immersive environment where participants can take a first-person view of simulated organizational processes, before, during, and after transformation
- Using either an organizational process simulator, or a game-based computer supported collaborative organizational learning environment (CSCOLE)

Institute for Software Research
UNIVERSITY OF CALIFORNIA, IRVINE

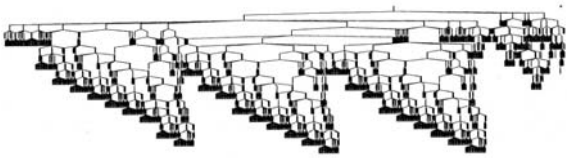
Case Studies

- Core process transformation at ONR
  - Process engineering using organizational transformation laboratory
- Supporting long-term, large-scale transformation
  - MMORPG-based computer supported collaborative organizational learning environment
    - Massively, multi-player online role playing game
- Transforming a software productivity crisis at "BigSysCo"
  - Process engineering and process simulator

Institute for Software Research
UNIVERSITY OF CALIFORNIA, IRVINE

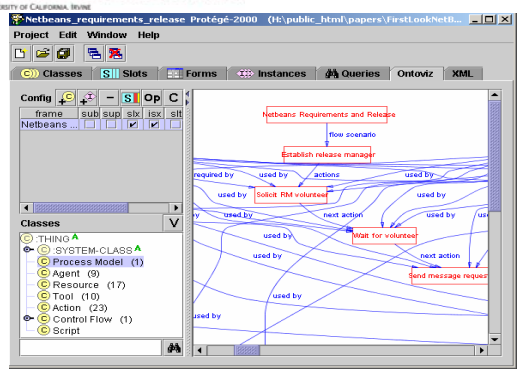
A complex software production process:

a *decomposition-precedence* relationship view
(19 levels of decomposition, 400+ tasks)

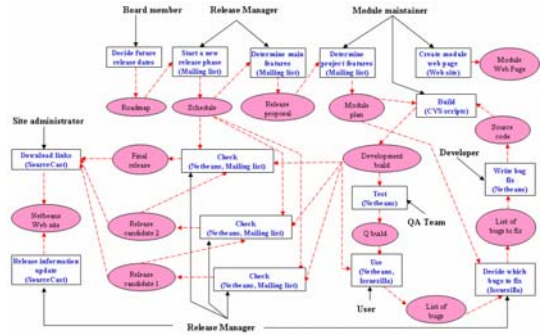


W. Scacchi, Experience with Software Process Simulation and Modeling, *J. Systems and Software*, 46(2/3):183-192,1999.

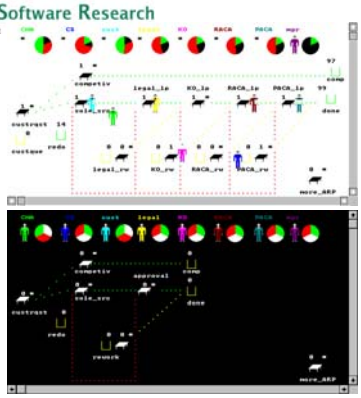
Institute for Software Research
UNIVERSITY OF CALIFORNIA, IRVINE

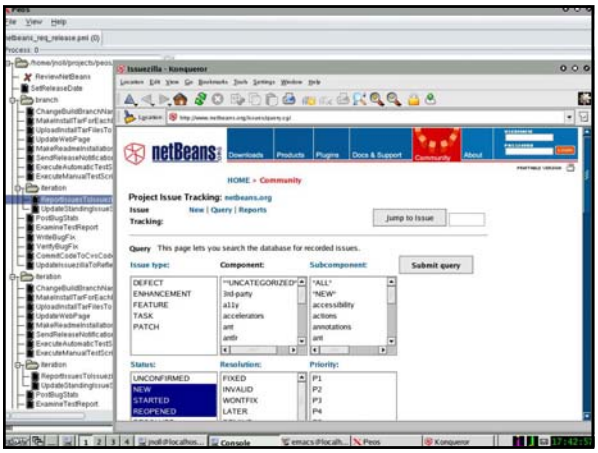


Institute for Software Research
UNIVERSITY OF CALIFORNIA, IRVINE



Institute for Software Research
UNIVERSITY OF CALIFORNIA, IRVINE





Institute for Software Research
UNIVERSITY OF CALIFORNIA, IRVINE

Identify and cultivate software productivity drivers

- Why are software developers so productive in the presence of technical and organizational constraints?
- Software developers must realize the potential for productivity improvement.
  - The potential for productivity improvement is not an inherent property of new software development technology.
  - Technological impediments and organizational constraints can nullify this potential.
- Thus, a basic concern must be to identify and cultivate *software productivity drivers.*
  - Examples include workplace incentives and alternative software business models

Institute for Software Research
UNIVERSITY OF CALIFORNIA, IRVINE

Open source *processes*

- Free/open source software *does not* embody the processes for how to develop, deploy, use or sustain them
  - Deploying free/open source software is low-cost, but often inefficient and sub-optimal
- Closed source software development, deployment, use, and support is also inefficient and sub-optimal
  - Explicit open source processes could also help closed source systems.

- Example of an open source process model of a *proposal submission* process, specified in a Process Markup Language, PML
- J. Noll and W. Scacchi, Specifying Process-Oriented Hypertext for Organizational Computing, *Journal of Network and Computer Applications*, 4(1):39-61, 2001.

Institute for Software Research
UNIVERSITY OF CALIFORNIA, IRVINE

Questions?

Institute for Software Research
University of California, Irvine

Acknowledgements

- *Project collaborators*:
  - Mark Ackerman, UMichigan, Ann Arbor
  - Les Gasser, UIUC
  - John Noll, Santa Clara University
  - Margaret Ellliot, Chris Jensen, UCI-ISR
  - And others at UCI-ISR
- *Funding support*:
  - National Science Foundation, ITR#-0083075, ITR#-#0205679, ITR#-0205724, and ITR#-#0350754
    - No endorsement implied.

62

Institute for Software Research
University of California, Irvine

References

- W. Scacchi, Understanding the Requirements for Developing Open Source Software, *IEE Proceedings--Software*, 149(1), 24-39, 2002.
- W. Scacchi, Free/Open Source Software Development Practices in the Computer Game Community, *IEEE Software*, 21(1), 59-67, January/February 2004.
- C. Jensen and W. Scacchi, Experiences in Discovering, Modeling, and Reenacting Open Source Software Development Processes, *Proc. Software Process Workshop*, Beijing, China, May 2005.
- W. Scacchi, Understanding Free/Open Source Software Evolution, in N.H. Madhavji, M.M. Lehman, J.F. Ramil and D. Perry (eds.), *Software Evolution*, John Wiley and Sons Inc, New York, (to appear).
- More at www.isr.uci.edu/research-open-source.html

63