Lecture 29
Comments on
The Cathedral and the Bazaar

Kenneth M. Anderson

Foundations of Software Engineering

CSCI 5828 - Spring Semester, 2000

# Today's Lecture

- Discuss other papers that resulted from the Cathedral and the Bazaar

  

# The size of a Bazaar Project

- Forrest J. Cavalier, III (What a Name)
  - Defines the terms
    - total size
      - actual number of participants
    - effective size
      - number of participants contributing to a particular activity
    - effective power
      - effective size * average contribution (person hours/week)

  

# Size implications on Debugging

- In order for a bug to be reported
  - it must be reachable
  - it must be triggered
  - it must be noticed (!)
  - it must be reported
- Cavalier claims that the effective size of a bazaar project decreases for each condition
- This is important since it implies that bazaar projects may require large "effective sizes"

  

# Debugging Alternatives

- Cavalier suggests open-source versions of
  - code review
    - Of course, open source always had informal code review, Cavalier is asking that it be formalized
  - unit testing
    - Cavalier is asking that open source projects to choose code structures that lend themselves to unit testing (which is a first step towards integration testing)



# Evolvable Systems

- Raymond points to a paper on evolvable systems as one aspect relevant to the bazaar style of development
- A paper by Clay Shirky
  - describes Web protocols as a "practical joke"
    - Ignored almost all existing hypertext research
    - Inefficient use of the network
    - No support for load balancing, security risks everywhere, etc.
    - Inconsistent client implementations
  - yet it "won", why?



# Evolvable vs. Centrally Controlled

- Shirky describes the early 90's
  - Three major ways of indexing/accessing information
    - anonymous ftp, gopher, wais
  - Homogeneous network services
    - AOL, Compuserve, Prodigy, etc.
- None of these things could interoperate
- When I first started using the Web, the "big win" in my eyes was that it presented a uniform interface to the Internet



# Evolvable vs. Centrally-Controlled, continued

- It was more difficult for the "centrally-controlled" standards to evolve
  - They had a high entry barrier to use
    - At least higher than the Webs
  - There designs made assumptions that hindered evolution
- The Web in contrast had something that partially worked and could be evolved in several areas simultaneously

# Shirky's definition of evolvable

- Those that proceed not under the sole direction of one centralized design authority but by being adapted and extended in a thousand small ways in a thousand places at once.



# Three rules for Evolvable Systems

- Only solutions that produce partial results when partially implemented can succeed.
  - They have room to grow
- What is, is wrong
  - Evolvable systems are currently addressing past needs but are quickly evolving to address current needs (faster than non-evolvable systems)
- Evolution is cleverer than you are
  - Systems that evolve are better able to adapt to new requirements



# Capacity for Change

- Centrally designed protocols start out strong and improve logarithmically. Evolvable protocols start out weak and improve exponentially. It's dinosaurs vs. mammals, and the mammals win every time. The Web is not the perfect hypertext protocol, just the best one that's also currently practical. Infrastructure built on evolvable protocols will always be partially incomplete, partially wrong and ultimately better designed than its competition.



# Web Success via Open Source

- Shirky (in a separate paper) asserts
  - The "View Source" command of Web browsers places the Web in the realm of open source techniques
  - He attributes this as one factor in the Web's overall success

# Homesteading the Noosphere

- Raymond's second paper on open-source
  - Examines the hacker culture and its implications on open source
  - The paper was written because Raymond perceived a contradiction in what hackers believe and what they do
  - He characterized this as the difference between open source ideology and its actual practice



# Zealotry and AntiCommercialism

- High Zealotry
  - "open source is my life"
- Medium
  - "open source is a good thing"
- Low
  - "open source is ok, sometimes"
- High AC
  - "Commercial software is evil"
- Medium
  - Commercial software is good and bad
- Low
  - Commercial software is OK



# Types continued

- Nine combinations are possible
  - and represented in the open-source community
- Free Software Foundation
  - Characterized as High Zealotry and High AC
- Other communities
  - Perl, tcl, BSD Unix, Python, Linux
  - Didn't quite buy into the FSF philosophy



# Open Source Definition

- <http://www.opensource.org/>
- Defines guidelines for open source licenses
  - An open-source license must protect an unconditional right of any party to modify (and redistribute modified versions of) open-source software.
- In theory, this allows projects to be split into many different directions
  - But this rarely happens… why?

# Ownership in Open Source

- In fact (and in contradiction to the anyone-can-hack-anything consensus theory) the open-source culture has an elaborate but largely unadmitted set of ownership customs. These customs regulate who can modify software, the circumstances under which it can be modified, and (especially) who has the right to redistribute modified versions back to the community.
- [Quoted from Homesteading…]



# Open Source Taboos

- There is strong social pressure against forking projects. It does not happen except under plea of dire necessity,with much public self-justification, and with a renaming.
- Distributing changes to a project without the cooperation of the moderators is frowned upon, except in special cases like essentially trivial porting fixes.
- Removing a person's name from a project history, credits or maintainer list is absolutely *not done* without the person's explicit consent.
- [Quoted from Homesteading…]



# More on Ownership

- Ownership is acquired in three ways
  - Start the project
  - Have ownership transferred to you
  - Take control of an orphaned project
- The latter two has "acceptable" ways that they are accomplished to be "legitimate"
  - public notice in many forums, demonstration that you can make productive improvements, ...



# Similar to Homesteading

- Raymond draws a parallel with these customs to a concept called "land tenure"
- To acquire ownership of land in this system
  - Be the first to homestead it
  - Transfer of title
  - Adverse Possession
- This style of ownership developed in areas where there was no or weak central authority
  - 'Lockean' theory of property after John Locke

# The Noosphere

- Lockean systems tend to arise where the expected return of a resource is greater than the expected cost of defending it
- The Noosphere is the space of all ideas
  - Open source hackers have evolved a Lockean theory of property in the subset of the Noosphere covering all programs
- Raymond asserts that "reputation" is the expected return of leading an open source project





# Gift Culture

- Raymond then asserts that the hacker culture surrounding open source resembles a gift culture
  - one where survival resources are plentiful
  - gift giving is a way of achieving status
- The rest of his paper then examines sub-issues related to this claim (which I will not go into further…)