The Cathedral and the Bazaar

Kenneth M. Anderson
Foundations of Software Engineering
CSCI 5828 - Spring Semester, 2001
Guest Lecture

Today's Lecture

- Discuss Background of the Paper
- Discuss Raymond's "Rules"
- Open Discussion on the Open-Source Approach

The Cathedral

- The Cathedral
  - equated to the traditional software life cycle
  - characterized by few releases with the goal being to reduce the bugs encountered by users
  - "I believed that important software needed to be built like cathedrals, carefully crafted by individual wizards or small bands of mages working in splendid isolation, with no beta to be released before its time"
  - Linux overturned this belief which was based on Raymond's experience with GNU software

GNU Software

- GNU's Not Unix (Recursive Acronym)
  - GNU is an effort by the Free Software Foundation to create free software tools better than the commercial tools they replace
  - Accompanied by the GNU copyleft (as opposed to a copyright)
    - Modifications can be made to the software as long as the modifier releases both the software and the source code of the modifications (This is my understanding)
  - Began work in the mid-1980s

The Bazaar

- "… rather, the Linux community seemed to resemble a great babbling bazaar of differing agendas and approaches (…) out of which a coherent and stable system could seemingly emerge only by a succession of miracles"
- software development characterized by
  – release early and often
  – delegate as much as possible
  – be open to feedback "to the point of promiscuity"

Background to the Paper

- Raymond joined the Linux community and formed an understanding of why it works
  – To test his theory, he consciously choose to run a smaller scale software project using the Bazaar-style of development
  – Most of the paper concerns his experience with fetchmail (previously popclient)

Users as co-developers

- Treat your users as co-developers
  – "least-hassle route to rapid code development and effective debugging"
  – Release early. Release often. And listen to your customers.
    - users are "stimulated and rewarded" by constant improvement
    - maximize the number of person-hours spent debugging
  – Problem transparency
    - With a large set of people, a problem discovered by one has a solution that is often transparent to another

Linus' Law

- Given enough eyeballs, all bugs are shallow
  – Debugging is parallelizable [Jeff Dutky]
- Or as Brooks would say
  – The task can be partitioned with minimal communication between parties
  – As such, this brings it closer to the realm in which workers and months can be interchanged!

Raymond's Process

- In order to test his open source theory
  - fetchmail was released early and often (every 10 days)
  - Anyone who contacted Raymond about fetchmail was added to the beta-list (300 people at its largest)
  - Each release was accompanied by a "chatty" announcement that encouraged participation
  - Raymond listened to his users; especially important: design decisions were "voted" on, developers who sent in patches/feedback were "stroked"

Some results

- "If you treat your beta-testers as if they're your most valuable resource, they will respond by becoming your most valuable resource"
- A critical design decision was submitted by one of the users
  - "Often, the most striking and innovative solutions come from realizing that your concept of the problem was wrong"
    - The same holds for optimizing code (you often have to switch to a new algorithm to get major speed increases)

Necessary Preconditions

- The Bazaar Style requires
  - An existing software system
    - You can't code from the ground up
    - Plus, you need to give something to your users that motivate them to participate
  - A coordinator must
    - be able to recognize good ideas
    - have a base level of design and coding skill (peer pressure prevents projects from starting otherwise)
    - have good people and communication skills

Social Concerns

- "Many people would expect a culture of self-directed egoists [hackers] to be fragmented, territorial, wasteful, secretive, and hostile."
- Raymond asserts that open source avoids this because the "economics" of this society is based on maximizing ego satisfaction and reputation among other hackers

Counter-Claim to Brook's Law

- Provided the development coordinator has a medium at least as good as the Internet, and knows how to lead without coercion, many heads are inevitably better than one.
- Use of Brooks throughout paper
  - "I quoted several bits from Fred Brooks's classic The Mythical Man-Month because, in many respects, his insights have yet to be improved upon."

Parallels to Brooks

- "Plan to Throw One Away"
- Conceptual Integrity
  - The coordinator acts as a chief architect
- "More Users find More Bugs"
- No Silver Bullet
  - What essential difficulties are being addressed by the open source movement?