The Corporate Origins of Open Source

Thomas Haigh

The Haigh Group/
University of Wisconsin, Milwaukee

4S, Vancouver,
November 2006

Structure of Talk

1. Review of canonical accounts of the origins of open source/free software
   - Linus Torvalds and Linux
   - Raymond Stallman and GNU
   - The Hacker Culture and Bell Labs
2. Examination of the role of the IBM SHARE scientific user group in the 1950s
   - Part of larger project on mathematical software
3. Some preliminary conclusions

1: Origins of Open Source Software – Three Fables

Open Source Idea?

- The **basic idea behind open source** is very simple: When programmers can read, redistribute, and modify the source code for a piece of software, the software evolves. People improve it, people adapt it, people fix bugs.

From OpenSource.org homepage

- “Open Source” concept attributed to 1998 meeting, Eric S. Raymond

Version 1: Finland, 1991

- Linus Torvalds sends a message to the comp.so.minix newsgroup…
- Linux was project of Linus Torvalds
  - Begun in 1991 as undergrad in Finland
- Now a leading serve operating system

```
From:
torvalds@klaava.Helsinki.FI
(Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: Gcc-1.40 and a
posix-question
Message-ID:
<1991Jul3.100050.9886@klaava
.Helsinki.FI>
Date: 3 Jul 91 10:00:50 GMT

Hello netlanders,
Due to a project I'm working
on (in minix), I'm
interested in the posix
standard definition. Could
somebody please point me to
a (preferably)
machine-readable format of
the latest posix rules? Ftp-
sites would be
nice.
```

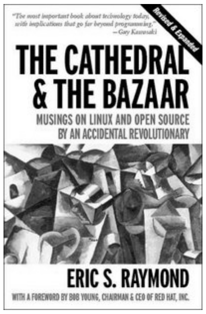
Power of the Internet

- Similar recent success for Firefox browser
- The story
  - Genius young programmer starts visionary project
  - Promising but incomplete versions posted on internet attract community of user/developers
  - A virtuous circle leads to exponential growth

HOW CLEAN, GREEN ATOMIC ENERGY WILL STOP GLOBAL WARMING
WIRED
THE FIREFOX REVOLT
At 14, he started hacking Netscape. At 19, he unveiled Firefox, a blazing-fast browser that was downloaded 1 million times the first day. Now it's the hottest thing on the Internet. (Watch your back, Bill Gates.)
+
WHY RIGHT-BRAINERS WILL RULE THE FUTURE
THE REVOLUTION IN BATTLEFIELD MEDICINE
COMEDY CENTRAL'S R&D LAB

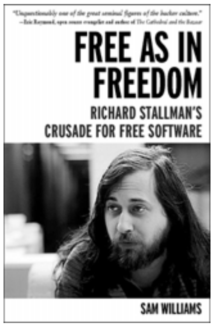
Bazaar Model

- Characteristics include
  - Users as co-developers
    - Projects start with personal problems to solve
    - Users debug systems – “many eyes make bugs shallow”
  - Early and frequent releases
  - High modularization
  - A “benevolent dictator” to lead project


THE CATHEDRAL & THE BAZAAR
MUSINGS ON LINUX AND OPEN SOURCE BY AN ACCIDENTAL REVOLUTIONARY
ERIC S. RAYMOND
WITH A FOREWORD BY BOB YOUNG, CHAIRMAN & CEO OF RED HAT, INC.

Version 2: MIT, 1983

- Richard Stallman was respected MIT “hacker”
  - Author of EMACS editor
- Since 1984 Stallman Coordinates GNU project
  - GNU is Not Unix (recursive name)
  - Intended to produce open, free version of Unix
- “Free as in speech... not beer”

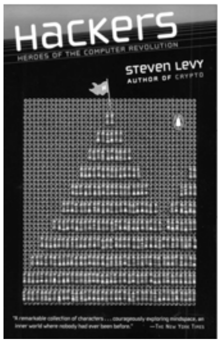


GNU’s Free Software Definition

- The freedom to run the program, for any purpose (freedom 0).
- The freedom to study how the program works, and adapt it to your needs (freedom 1). Access to the source code is a precondition for this.
- The freedom to redistribute copies so you can help your neighbor (freedom 2).
- The freedom to improve the program, and release your improvements to the public, so that the whole community benefits (freedom 3). Access to the source code is a precondition for this.

Version 3: Hacker Culture

- Stallman was propagating and defending a tradition going back to the late 1950s at MIT
- Propagated and revitalized by
  - Personal computes
  - Widespread internet access



The Hacker Ethic

- Access to computers… unlimited and total
- All information should be free
- Mistrust authority – promote decentralization
- Hackers should be judged by their hacking…
- You can create beauty and art on a computer
- Computers can change your life for the better

From ch. 2 of Hackers, by Steven Levy,

Summary of 3 Conventional Views

- Stress
  - Hacker culture and ideological commitments
  - Unpaid enthusiast virtuosos
  - Charismatic individuals
  - Novel licensing arrangements
- All about operating systems

A New Origin Story

- Scientific software libraries
- 1950s
- No concern with licensing arrangements
- Motivated by pragmatic commercial interests
  - Avoidance of duplicated efforts on generic programs
  - To free up resources for areas of proprietary interests

2: Mathematical Software and Open Source

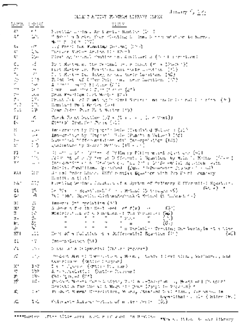
Scientific Computing

- Original function of early machines
  - Harvard Mark I, ENIAC
  - Source of the term “computer”
- Many applications are concerned with modeling natural or man made systems
  - Hydrogen bomb physics
  - Fluid Dynamics of air for aerospace
  - Celestial mechanics for space navigation

Mathematical Libraries

- Produced internally within computer centers
  - First example for EDSAC circa 1950
    - Invented along with subroutine
    - Discussed in 1951 programming text
    - Included Runge-Kutta differential equation routine
  - First US grant to support development may be for ILLIAC
    - Numerical Analysis funding from ONR 1950-1958

Subroutine library 1955 →



Early Needs

- Initially: very basic assembly language subroutines
  - Multiplication, square root, binary to decimal, floating point simulation, etc.
- FORTRAN (1956) covers basics, but plenty of challenges left
  - Each computer center is likely to need routines for
    - Linear algebra and matrix manipulation
    - Ordinary and Partial Differential Equation solvers
    - Special and Elementary functions
    - Curve fitting and least squares
    - Fast Fourier Transformation

3: SHARE and Mathematical Software

IBM 701/704/709

- Large, “first generation” machines of 1950s
  - Worth approximately $2 million
- Designed for technical computation
  - Early users dominated by Southern California aerospace firms
  - Cold war context
- Many employees for each computer installation

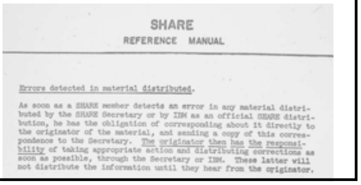
704 at LLNL, 1956

SHARE IBM User Group

- SHARE founded 1956
  - Cooperative group for users of large IBM computers
    - Discussions begin among IBM 701 users
    - SHARE represents “large” IBM scientific machine users
    - Representatives from each installation (52 by end of 1956)
  - Intended to “share” programs, expertise, experiences and best practices
    - Lobbying of IBM to alter machines or policies

SHARE Software Library

- Routines contributed by user sites
  - Reproduction and catalog handled by IBM
  - Classification scheme developed to organize
  - Contributors responsible for maintenance
- List posted of routines devised & desired



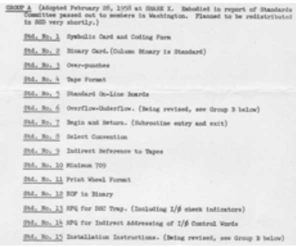
SHARE
REFERENCE MANUAL

Errors detected in material distributed.

As soon as a SHARE member detects an error in any material distributed by the SHARE Secretary or by IBM as an official SHARE distribution, he has the obligation of corresponding about it directly to the originator of the material, and sending a copy of this correspondence to the Secretary. The originator then has the responsibility of taking appropriate action and distributing corrections as soon as possible, through the Secretary or IBM. These latter will not distribute the information until they hear from the originator.

SHARE Practices

- Standardization needed to share code and practices
- Standardize machine configuration
  - Setting of switches, control panels, etc
- Standardize system software
  - Assembler and utility programs (not supplied by IBM)
  - Leads to big project to create “Share Operating System”



GROUP A (Adopted February 26, 1958 at SHARE X. Embodied in report of Standards Committee passed out to members in Washington. Planned to be redistributed in SSD very shortly.)

Std. No. 1 Symbolic Card and Coding Form

Std. No. 2 Binary Card. (Column Binary is Standard)

Std. No. 3 Over-punches

Std. No. 4 Tape Format

Std. No. 5 Standard On-Line Boards

Std. No. 6 Overflow-Underflow. (Being revised, see Group B below)

Std. No. 7 Begin and Return. (Subroutine entry and exit)

Std. No. 8 Select Convention

Std. No. 9 Indirect Reference to Tapes

Std. No. 10 Minimum 709

Std. No. 11 Print Wheel Format

Std. No. 12 EOF in Binary

Std. No. 13 RPQ for DSC Trap. (Including I/Ø check indicators)

Std. No. 14 RPQ for Indirect Addressing of I/Ø Control Words

Std. No. 15 Installation Instructions. (Being revised, see Group B below)

SSD

- Mechanism for communication between meetings
  - Mailing of large bundles of assorted materials
    - Committee reports
    - Drafts for comments
    - Letters, inquiries and responses
      - Including bug reports
- Also microfilms of source code for programs

Packaging of Mathematics

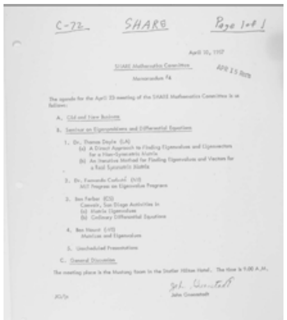
- Many routines are for mathematical functions
  - Substantial duplication and overlap in contributed routines
  - Quality issues
- Importance of tacit knowledge
  - Limits use, causes support issues
  - “Black boxing” of mathematical procedures

SHARE Labor

- Installation reps are senior figures
  - Responsible for design and specification
  - Commit employees of their firms to develop code
- Economy of effort in developing generic routines
  - Driven by economics – save time and money
  - No proprietary advantage in cosine routine

SHARE Structure

- Committees to manage particular projects
  - Mathematical software is one important area
  - Subcommittees for particular projects



C-72 SHARE Page 1 of 1

April 10, 1957

SHARE Mathematics Committee

APR 15 1957

Memorandum #4

The agenda for the April 23 meeting of the SHARE Mathematics Committee is as follows:

A. Old and New Business

B. Seminar on Eigenproblems and Differential Equations

1. Dr. Thomas Doyle (LA)
   (a) A Direct Approach to Finding Eigenvalues and Eigenvectors for a Non-Symmetric Matrix
   (b) An Iterative Method for Finding Eigenvalues and Vectors for a Real Symmetric Matrix
2. Dr. Fernando Corbató (MI)
   MIT Program on Eigenvalue Program
3. Ben Farber (CS)
   Convair, San Diego Activities in
   (a) Matrix Eigenvalues
   (b) Ordinary Differential Equations
4. Ben Mount (NY)
   Matrices and Eigenvalues
5. Unscheduled Presentations

C. General Discussion

The meeting place is the Mustang Room in the Statler Hilton Hotel. The time is 9:00 A.M.

John Greenstadt

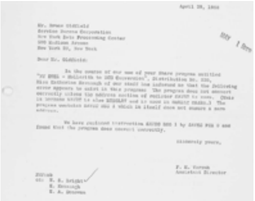
JG/js

SHARE and the Four Freedoms

- Freedom to run – YES
- Freedom to study and adapt source code - YES
- Freedom to redistribute – YES
  - Pretty much all 704/9/90 were members
- Freedom to improve and release to the public – YES

Similarities in Practices

- Ad-hoc collaboration groups for specific projects
  - Some effort at modular code architecture
- Mechanisms to share and respond to bug reports
- Standards for coding and configuration to facilitate collaboration
- Open circulation of proposals and design documents
  - “Indoctrination” into culture



April 28, [illegible]

Mr. Bruce Oldfield
Service Bureau Corporation
New York Data Processing Center
[illegible] Madison Avenue
New York 22, New York

[illegible]

Dear Mr. Oldfield:

In the course of our use of your Share program entitled "[illegible] - Hollerith to BCD Conversion", Distribution No. [illegible], Miss Katherine [illegible] of our staff has informed me that the following error appears to exist in this program. The program does not execute correctly unless the address section of [illegible] [illegible] is zero. [illegible] The program [illegible] which in itself does not ensure a zero address.

We have replaced instruction [illegible] by [illegible] and found that the program does execute correctly.

Sincerely yours,

F. H. [illegible]
Assistant Director

[illegible]
cc: [illegible]

Challenges to SHARE

- Problems develop in open source model
- See Akera – “The Limits of Voluntarism”, T&C, 2001
  - Following problems with the “SHARE Operating System” project the writing of system software migrates to IBM
- But mathematical software largely doesn’t
  - SHARE is main distribution mechanism until early 1970s
    - Large labs rely on own code libraries

4: Concluding Ponderings

Commercial Origins of Open Source Practices in 1950s

- To recap, by 1956 we already have
  - All formal characteristics of “free” software
  - Many practices of modern open source development
- But not the ideology of free software
  - Seen as pragmatic actions, economically driven sharing

Hidden Commonality

- Shared engineering culture?
  - 1950s MIT Hackers
  - 1950s Aerospace engineering computing groups
- Seek to solve tasks in technically efficient manner
  - Avoid needless duplication of work
  - Provide tools to people who need them

Shows need for Separation of Ideology and Practice

- Open source practices are older, more widespread than open source movement, so…
  - How important is the ideology?
  - Is selective use open source by big firms (IBM etc) the exception or the rule?
- How important are scientific norms to open source practices?
  - Publication and sharing of data
  - Goes back to 17th century gentlemen