



2. 오픈 소스 개관

목차

- 해커
- 오픈 소스
 - 정의와 역사
 - 라이선스
 - 소프트웨어 개발
- 세계의 오픈 소스 소프트웨어
- 국내의 오픈 소스 소프트웨어
- 바람직한 개발 모델
- 우리의 할 일
- 결론

Hackers

- 초기 컴퓨터 프로그래머 이후, 컴퓨터의 다양한 기능을 활용하여 자신의 것으로 하고자 함
- 역사적으로는 1960년대 MIT와 PDP-1에서 시작
- 인터넷의 전신인 ARPAnet의 발달과 더불어 “네트워크 국가”로 성장
- 현재에도 컴퓨팅 문화의 발전을 주도

Hackers (2)

- **오픈 소스의 중심 세력**
- **나쁜 의미로서의 사용 – Crackers**
 - 반드시 구분하여 사용하여야 함
- **크게 세가지의 구분**
 - MIT/ITS/LISP 해커들 – 1960-1980년대
 - BSD/UNIX/C 해커들 – 1969년 이후
 - PC/DOS/Mac 해커들 – 1970년 후반부터
- **네트워크의 발달과 더불어 결집이 가능**

Hackers (3)

■ MIT/ITS/LISP 해커들

- AI연구소 중심. ITS운영체제와 LISP언어
- 초기 전통에 큰 영향
- AI연구소의 상업화와 ITS운영체제의 소멸
- Richard Stallman – The last hacker

■ BSD/UNIX/C 해커들

- 현재의 중심 세력
- 유닉스 워크스테이션과 C 언어 사용자
- 유닉스와 인터넷의 광범위한 확산에 노력
- 1990년대 이후 “Free Unix”에 의해 새로운 발전

Hackers (4)

■ PC/DOS/Mac 해커

- Apple II의 등장 이후, 개인용 컴퓨터 중심으로 발전
- BASIC/C/PASCAL등의 언어 사용
- 전체적인 네트워킹의 부재
- 윈도우의 발전에 큰 도움
- 386이후 PC의 유닉스가 발전하자 유닉스 진영과 혼합되기 시작
- 일반 개인을 위한 컴퓨터 사용 보급에 일조

■ 해커들의 통합

- PC유닉스의 보급과 인터넷의 발달
- 서로 다른 그룹과 교류

Hackers (5)

■ 정보의 공유

- 자신들의 테크닉을 널리 사용하게 하는 것에 의해 즐거움을 느낌
- 기술와 소프트웨어의 공유
- 네트워크와 인터넷이 이를 진일보하게 함
- Warez... ??

■ 상업 소프트웨어에 의한 위기

- 1990년대 이후 MS의 성장과 유닉스의 담보
- PC용 유닉스의 성장 이후 새로운 국면

Hackers (6)

- PC 유닉스의 발전으로 인해 해커들은 자신의 PC에서 워크스테이션급의 프로그램 실행 가능
 - 오픈 소스 소프트웨어와 운영체제의 발전에 큰 원동력이 됨
 - 1990년 초 Linux와 386BSD가 386에서 실행
- 현재
 - PC용 유닉스는 현재의 주도적인 해커의 운영체제
 - 오픈 소스 소프트웨어의 발전을 주도
 - 기술적 성공에 이은 사회적 성공을 달성
 - 소프트웨어 개발 이외에도 사회적 운동을 주도

소프트웨어의 구분

- 상업적 소프트웨어
- 데모
- 트라이얼
- 셰어웨어
- 프리웨어
- 바이너리만
- 프리웨어
- BSD스타일 오픈소스
- GNU스타일 오픈소스
- 소스코드 제공

Royalty-free libraries	X	X	X	X			
Open Source (BSD-Style)	X	X	X	X	X		
Open Source (Apache Style)	X	X	X	X	X	X	
Open Source (Linux/GNU style)	X	X	X	X	X	X	X
License Feature	Zero Price Avenue	Re distrib utable	Unli mite d Usa ge	Sour ce Cod e Avai lable	Sou rce Cod e Mo difi able	Public "Check -ins" to core codeba se	All deri vati ves mu st be free

오픈 소스

- Open Source Software (OSS)
 - 소스가 공개된 소프트웨어의 통칭
- 상업적 소프트웨어 개발과 반대되는 개념
 - PC용 유닉스와 함께 확대되는 경향
- 여러가지 형태와 다양한 라이선스를 가짐
- 다양한 개발 방식
- 최근의 컴퓨터 비즈니스에 큰 영향

오픈 소스 - 역사

- “해커”의 역사와 같이 함
- 프리(free) 소프트웨어
 - Free는 자유 - 프로그래밍과 배포의 - 를 의미
 - Richard Stallman이 1980년 초 Free Software Foundation 를 구성하고 GNU시스템의 작성에 힘씀
 - Free는 “무료”의 의미가 아니지만 많은 사람들이 혼동
- Debian Project의 Bruce Evans
 - Debian에 포함될 오픈 소스 소프트웨어의 가이드라인
 - 여러 사람의 도움으로 1997년 Open Source라는 말이 정립 됨
 - <http://www.opensource.org>



Free Software VS Open Source

■ Open Source

- 무난하지만, 소스만 공개한다는 의미에 치중하는 인상
- 일반적으로 받아들이기는 쉬움

■ Free Software

- ‘자유’(개발과 사용, 재배포)의 의미에 비중
- ‘무료’라는 의미로 오해하기 쉬움

오픈 소스 - 정의

Opensource.org

- **Open Source Definition(v1.0)**
 - 어떤 소프트웨어가 오픈 소스 소프트웨어인가를 규정
 - 법적인 라이선스 조항이 아니라 일반적인 사회 통념을 반영
- **Open Source Initiative**
 - Open Source 마크와 캠페인 운영
 - 알려진 오픈 소스 제작자들에 의한 운영

오픈 소스 – 정의 (2)

- Free Redistribution
- Source Code
- Derived Works
- Integrity of the Author's Source Code
- No Discrimination Against Persons or Groups

오픈 소스 – 정의 (3)

- No Discrimination Against Fields of Endeavor
- Distribution of License
- License Must Not Be Specific to a Product
- License Must Not Contaminate Other Software
- Example Licenses
 - GNU GPL, BSD, X Consortium, Artistic, MPL

오픈 소스 - 라이선스

- 크게 GNU GPL와 BSD 라이선스
- 오픈 소스의 경우, 해당 소프트웨어는 저작권 문제를 해결하기 위해 특별한 라이선스를 적용
 - GNU: General Public License (GPL)
 - BSD License
 - Artistic License, Mozilla Public License(MPL), FreeBSD License, XFree86 License 등
- 어느 정도로 자유로운가?

A Holy War: GPL vs BSD

■ GPL

- 파생물도 GPL을 따라야 함
- 소스를 반드시 구할 수 있어야 함(변경 포함)
- Richard M. Stallman이 주창함
- LGPL과 LGPL2

■ BSD

- 파생물은 별도의 라이선스 부여 가능
- 그럴 필요 없음. 상업적 사용 가능
- BSD UNIX의 라이선스 조항
- 여러가지 변종 존재 (FreeBSD, X, Artistic etc..)

오픈 소스 - 개발

- 이전의 연구소나 지역 단위의 개발에서 네트워크를 통한 전세계적인 개발 방식으로 전환
- 현재는 인터넷의 여러 통신 수단을 통한 집단적 개발 방식이 주류
- 이러한 개발 방식은 Eric Raymond의 “성당과 시장”에서 잘 정의하였음
- 크게 성당식과 시장식으로 구분할 수 있음

오픈 소스 - 개발(2)

■ 성당식 개발

- 소수의 뛰어난 프로그래머들에 의해 제작
- 장기적인 릴리즈 중심, 중간판이 없음
- 개발 과정이나 기능 개선은 주로 개발자들의 판단에 의해
- 깔끔하고 정리된 코드
- 상업적 소프트웨어의 기본적인 개발 방식
- 오픈 소스에서도 널리 사용되고 있음
 - emacs, GNU toolkit, X Window..

오픈 소스 - 개발(3)

■ 시장식 개발의 특징

- 해커 자신의 필요에 의해 개발이 시작
- 인터넷에 릴리즈 후 관심 있는 다른 사람들에 의해 개발이 공동으로 진행
 - 병렬적인 테스트와 디버깅 가능
- 반복되는 릴리즈와 피드백을 통한 버그 수정
 - 빠른 수렴성
- 사용자들 간의 의견 수렴에 의한 방향 결정
 - 사람들이 원하는 방향으로의 진행

시장식 개발의 성공

- 세상에는 충분한 수의 해커들이 있음
 - 네트워크의 발달로 결집 가능
- 기존의 소프트웨어 공학의 이론들이 무시

The ability of the OSS process to collect and harness the **collective IQ of thousands of individuals** across the Internet is simply amazing. More importantly, OSS evangelization **scales with the size of the Internet** much faster than our own evangelization efforts appear to scale.

- from Halloween Documents I

세계의 오픈 소스 소프트웨어

- **오픈 소스 소프트웨어는 이미 현재의 인터넷을 떠받치고 있음**
 - Bind와 sendmail, apache의 사용
 - Socket은 BSD의 구현 사항
- **현재에는 그 사용 범위를 더욱 넓히고 있음**
 - 해커들의 전문 분야인 운영체제와 네트워킹 시스템에서 출발하여, 일반 사용자를 위한 GUI시스템과 오피스 소프트웨어로 확대

중요한 OSS

■ Bind와 sendmail, apache

- 각각 DNS시스템과 전자우편, 웹 서버 시스템을 떠받치는 중요 소프트웨어
- 모두 오픈 소스

"open source software works." ... **BIND** has absolutely dominant market share as the single most mission-critical piece of software on the Internet. **Apache** is the dominant Web server. **SendMail** runs probably eighty percent of the mail servers and probably touches **every** single piece of e-mail on the Internet - *Tim O'Reilly, interview with Techweb*

중요한 OSS (2)

■ GNU 프로젝트

- Richard Stallman에 의해 1980년 초반에 시작
- 소스가 공개된 공개 유닉스 클론이 목적
- HURD커널은 상당히 늦게 발표되고 개발이 느림
- 나머지 시스템은 거의 완성
- Emacs, GCC, binutils, fileutils, textutils

■ X 윈도우 시스템

- 산업 표준의 유닉스 윈도우 시스템은 오픈 소스
- XFree86은 PC 플랫폼을 위한 X윈도우

중요한 OSS (3)

■ Perl

- 시스템 관리자를 위한 언어에서 범용 언어로
- CGI, 시스템 관리 등 만능의 스크립트 언어
- 저자인 Larry Wall은 patch의 제작자로도 유명

■ Mozilla

- 소스가 공개된 넷스케이프사의 웹브라우저
- 대기업이 소스를 공개한 첫번째 사례

■ GNOME과 KDE시스템

- 사용자 위주의 데스크탑 시스템을 목표

중요한 OSS (4)

■ Linux

- 1990년대 Linus Torvalds에 의해 커널 시작
- 나머지는 GNU시스템과 다른 오픈소스에 의존
- 이후 시장 개발 방식의 전형적인 예가 됨
- 멀티플랫폼 유닉스 운영체제
- 폭넓은 사용자층과 활발한 개발
- 여러가지 특징있는 배포본의 등장
- 여전히 커널은 Linus에 의해 주도되고 있음
- 일부 사람은 GNU/Linux라 부르고자 함

중요한 OSS (5)

■ Programming Tools

- Zope, and PHP, are popular engines behind the "live content" on the World Wide Web.
- Languages:
 - Perl
 - Python
 - Ruby
 - Tcl/Tk
- GNU compilers and tools
 - GCC
 - Make
 - Autoconf
 - Automake
 - etc.

오픈 소스 웹 사이트

- Free Software Foundation www.fsf.org
- Open Source Initiative www.opensource.org
- Freshmeat.net
- SourceForge.net
- OSDir.com
- developer.BerliOS.de
- Bioinformatics.org
- see also individual project sites; e.g.,
www.apache.org; www.cpan.org; etc.

중요한 OSS (5)

■ FreeBSD, OpenBSD, NetBSD

- 4.3BSD Networking Release 2와 BSD의 마지막 버전인 4.4BSD-Lite, 386BSD기반
- 버클리 유닉스의 충실한 계승자
- 90년대 초 라이선스 문제로 법적인 분쟁
- 안정된 인터넷 서버나 보안 서버로 많이 사용
- 원래 BSD의 개발 방식은 성당식이나 이들 운영체제는 대부분 시장 방식의 개발방식을 많이 채용하고 있음

국내의 오픈 소스 소프트웨어

- 1980년 후반 이후 유닉스의 보급과 함께 발전
- 상업용 UNIX사용자들에게서 출발
 - hanterm, helvis 등
- 주로 한국어 사용에 관련된 프로그램 위주
 - Localization과 Internationalization
- UNIX와 PC(8-32비트) - BASIC/PASCAL/C
해커들

국내의 오픈 소스 소프트웨어(2)

- **Linux의 발달과 더불어 발전**
 - 수많은 한글관련 프로그램의 등장
- **지역화에 치중하는 경향 있음**
 - 독자적 국제화된 소프트웨어 개발 부족
- **잘 운영되는 소프트웨어가 있는가?**
 - 해커 문화의 형성이 부족
 - 뒷심, 테크닉 부족!

국내의 OSS 프로젝트

■ Hanterm

- 가장 오래된 한국의 오픈 소스의 대표 소프트웨어
- Xterm 기반으로 한글 처리의 표본 - 조합 인코딩 글꼴
- <http://elf.kaist.ac.kr/hanterm>

■ ami

- X윈도우용 국제화된 한글 입력기
- <http://www.kr.freebsd.org/~hwang>

■ OpenHWP

- 아래한글의 위기에서 출발하여 공개 HWP 클론을 만드는 것이 목표
- 현재 프로젝트의 진행이 느림

국내의 OSS 프로젝트 (2)

■ Debian-KR Project

- 공개 리눅스 배포본인 데비안의 한국어 지원
- <http://www.debian-kr.org>

■ Korea FreeBSD Users Group

- 공개 유닉스인 FreeBSD의 사용자 모임
- 문서 번역과 패키지 제작 등
- <http://www.kr.FreeBSD.ORG>

■ MiziOS

- 최초의 독립적인 국내 리눅스 배포본
- <http://linux.mizi.co.kr>

국내의 OSS 프로젝트 (3)

- 기타 메일링 리스트나 프로젝트들
 - KLDP: Korean Linux Documentation Project
 - 리눅스 관련 문서의 한국어 번역
 - 가장 성공한 프로젝트 중 하나, <http://kldp.org>
 - GNU Free Translation Project, 한국어 팀
 - GNU프로젝트 산물의 메시지 번역(gettext)
 - <http://translation.gnu.or.kr>
 - 이외 소규모 모임으로 emacs-kr, hangul-patch등이 있음

바람직한 개발 모델

- 소프트웨어의 최초 모델 작성 -> 공개 -> 사용자의 증가 -> 피드백을 받아 개선 -> 새로운 개발자의 참여 -> 계속 발전
- 특히 피드백이 중요
 - 주는 사람과 받는 사람의 자세
- Software Practice HOWTO
 - 이러한 소프트웨어의 개발에 대한 지침서

바람직한 개발 모델 (2)

- 소스트리의 제공(CVS 등)
 - 특정 OS나 파일 포맷을 사용하지 않음
- 피드백을 받을 수 있는 공간 제공
 - 홈페이지, 메일링 리스트, 게시판 등
- 개발자간 의사 소통의 중요성
 - 원격지에서도 협동 개발이 가능
- 그리고 사용자/개발자들의 관심
 - 그러나 전부는 아님: openhwp Project

바람직한 개발 모델 (3)

- 사용자/준개발자 측면
 - 올바른 피드백 주기
 - 반복 가능한 오류, 시스템 환경, 패치 제공
 - Code forking은 자신있을 때에만!
 - 숨기지 말 것
 - 배포본이나 개인 홈페이지 등.
 - 자신만 아는 것은 아무에게도 도움되지 않음
 - 가능하다면, 되돌려 줄 것!
- 중요한 것은 모든 사람이 쉽게 볼 수 있게 하는 것
 - 나머지는 사용자들이 알아서 해 준다!

바람직한 개발 모델 (4)

- 프로젝트를 이끄는 능력
 - 새로운 기능의 추가, 발전에 대한 전망
 - 능력있는 후계의 발굴과 양성
 - 프로젝트가 커질 경우 의사결정 기구의 마련
 - 최종 판단은 1인에게 - Linux
 - 집단 의사 결정 - Apache, Perl
- 도움에 감사하고, 칭찬에 인색하지 말 것
 - Credit의 유지는 반드시 필요!!



Open Source is Moving up the Stack



Applications?

Apache, JBoss, Eclipse

Linux

TCP/IP, SendMail, HTTP

OSS의 가능성

- **해커용 프로그램 이상의 질을 달성할 수 있는가?**
 - 사용자 위주 프로그램, 데스크탑, 그래픽...
 - 이미 일부 프로그램에서 달성 - GNOME, KDE, GIMP
 - 인터페이스에 충실 - 그래픽 인터페이스의 분리
- **현재까지는 “원본에 충실한” 소프트웨어..**
 - ‘진화’할 수 있는가?
 - Emacs, Linux, GNOME...
- **일반인에게 OSS는**
 - 쓰기 편리하고
 - 구하기 쉬운 소프트웨어

우리의 할 일

- **OSS에 대한 지속적인 관심**
 - 일회적인 것이 아니라 실제적인 참여를 장려
- **독자적인 소프트웨어의 제작**
 - 국제적인 조류와 필요에 부응
 - OSS의 개발 방식을 수용
- **국제화에 대한 심도있는 연구와 수용**
 - 각종 소프트웨어의 국제화는 세계적인 추세
 - 표준과 구현에 대한 관심
 - 부족한 점에 대한 연구와 도움

결론

- 오픈 소스 소프트웨어는 컴퓨터의 발생 이후부터 존재
- 21세기의 소프트웨어 발전의 커다란 축으로 부활
- 해커 중심에서 일반인과 비즈니스에 접근, 통합하는 경향
- 각자의 지지와 개발 참여가 모두를 위한 도움이 됨

관련 문서

- **The Cathedral and The Bazaar - Eric S. Raymond**
 - 오픈소스 개발 방식에 대한 고전(?)
- **Halloween Document - Microsoft**
 - 오픈 소스에 대한 잘 정리된 보고서
- <http://www.opensource.org>
- <http://www.gnu.org>



3. 오픈소스에 대한 오해

여기서 다루지 않는 것.

- Open Source로 돈을 벌 수 있나?

Mine for Open Source

■ 나의 경험

- 1998년 Mozilla 소스 코드 공개 시 시작
- Gettext(.po)로 번역하여 CVS에 제공, 1.0 부터 제대로 된 Localization 시스템 구축.
- 주요 작업 버전: Mozilla 1.0~1.7, Firefox 0.7~0.9, Thunderbird 0.5~0.7

■ 주요 개발 활동

- Bug Report : 버그질라를 통해 보고 및 Followup
- Code Tester: 일일 버전에 대한 테스트 (주로 i18n)
- Code Offering : 국내 CVS Committer에게 코드 제공 (신청식, jshin@i18n10n.com)

Contribution of Open Source

- Joining local open source community
- Bug Reporting for good products
- Beta Testing especially i18n and l10n
- Localization for ko-KR locale
- Translation site management
- Code review, writing and testing
- Submit your code
- CVS committer
- Be major role member (if be, you give up your general job!)

Myths of Open Source

- OSP is opened!
- OSP is closed (hard to catch high level)

- OSP is difficult!
- OSP is easy (from bug report)

- OSP is inexpensive
- OSP is expensive (time and money)

- OSP is sharing
- OSP is non-sharing (self-development)

Making mine project

- Start from simple application or library
- License : GPL or BSD?
- Homepage, CVS, Distribution
 - Sorceforge.net <http://sourceforge.net>
 - KLDP.net <http://kldp.net>
 - Google Code <http://code.google.com>
- Writing a Code
- Make a release cycle and points
- Marketing and make a friends