Table of Contents

목차

1. 오픈소스의 정의

Open Source란?

우리는 오픈 소스에 대해 얼마나 알고있을까? "오픈 소스"라는 단어를 들었을 때 드는 생각은 '소스 코드가 오픈되어 있으니 무료로 사용할 수 있겠구나!'와 같을 것이다. 실제로 네이버 지식백과에 오픈 소스를 검색해보면 다음과 같이 설명한다.
'오픈소스란 누구나 이용 가능한 퍼블릭 도메인이다. 오픈 소스 운동은 독점적이고 상용화한 소프트웨어에 반하는 것이다. 즉, 오픈소스를 이용하면 그 이용으로 파생된 2차 저작물의 소스코드도 공개해야 한다.'
하지만 이 내용만으로는 오픈 소스에 대한 설명을 마무리지을 수 없다.
보다 더 정확하고 자세한 내용을 알아보기 위해 오픈 소스의 정의를 정리하여 발표한 오픈 소스 이니셔티브(Open Source Initiative)에 게재된 내용을 살펴보자.

1. 소개

   > 오픈 소스는 단지 소스 코드에 대한 접근만을 의미하는 것이 아니다. 오픈 소스 소프트웨어를 배포할 때에는 아래의 기준을 준수해야한다.

2. 무료 재배포

   > 라이센스는 여러 다른 출처의 프로그램이 포함된 총괄적인 소프트웨어 배포의 구성 요소로 소프트웨어를 판매하거나 양도하지 못하도록 제한하지 않는다. 라이센스는 그러한 판매에 대해 로열티 또는 기타 수수료를 요구하지 않는다.

3. 소스 코드

   > 프로그램은 소스 코드를 포함해야하며 컴파일 된 양식뿐만 아니라 소스 코드로도 반드시 배포할 수 있어야한다. 어떤 형태의 제품이 소스 코드와 함께 배포되지 않는 경우, 합리적인 복제 비용 이상으로 소스코드를 얻는 방법이 널리 알려져 있으며, 인터넷을 통해 무료로 다운로드하는 것이 바람직하다. 소스 코드는 프로그래머가 프로그램을 수정하는 기본 형식이어야한다. 의도적으로 읽기 어렵게 만든 소스 코드는 허용되지 않는다. 전처리기 또는 변환기의 출력과 같은 중간 양식은 허용되지 않는다.

4. 파생 작업

   > 라이센스는 수정 및 파생된 저작물을 허용해야하며 원본 소프트웨어의 라이센스와 동일한 조건으로 배포할 수 있어야한다.

5. 저자 소스 코드의 무결성

> 라이센스는 빌드 타임에 프로그램을 수정하기위한 목적으로 소스 코드와 함께 "패치 파일"의 배포가 허용되는 경우에만 소스 코드가 수정된 형태로 배포되는 것을 제한할 수 있다. 라이센스는 수정된 소스 코드로 빌드된 소프트웨어의 배포를 명시적으로 허용해야한다. 라이센스는 파생된 저작물이 원래 소프트웨어와 다른 이름 또는 버전 번호를 포함하도록 요구할 수 있다.

6. 개인이나 단체에 대한 차별 금지

> 라이센스는 어떤 개인이나 집단을 차별해서는 안된다.

7. 노력의 분야에 대한 차별 금지

> 라이센스는 특정 분야에서 프로그램을 사용하지 못하도록 제한해서는 안된다. 예를 들어, 프로그램이 비스니스에서 사용되거나 유전 연구에 사용되는 것을 제한하지 않아야한다.

8. 라이센스 배포

> 프로그램에 첨부된 권리는 프로그램이 재배포되는 모든 사람에게 해당 당사자가 추가 라이센스를 발급할 필요 없이 적용되어야 한다.

9. 제품 고유 라이센스 금지

> 프로그램에 첨부된 권리는 특정 소프트웨어 배포의 부분이 되는 프로그램에 의존해서는 안된다. 프로그램이 해당 배포에서 추출되어 프로그램 라이센스 조건에 따라 사용되거나 배포되는 경우, 프로그램을 재배포하는 모든 당사자는 원래 소프트웨어 배포와 함께 부여된 원본과 동일한 권한을 가져야한다.

10. 라이센스로 다른 소프트웨어 제한 금지

> 라이센스는 라이센스가 부여된 소프트웨어와 함께 배포되는 다른 소프트웨어에 제한을 두어서는 안된다. 예를 들어, 라이센스는 동일한 매체에 배포된 다른 모든 프로그램이 '오픈 소스 소프트웨어'여야 한다고 주장해서는 안된다.

11. 라이센스의 기술 중립성

> 라이센스의 조항은 개별 기술 또는 인터페이스 스타일에 근거할 수 없다.

Closed Source란?

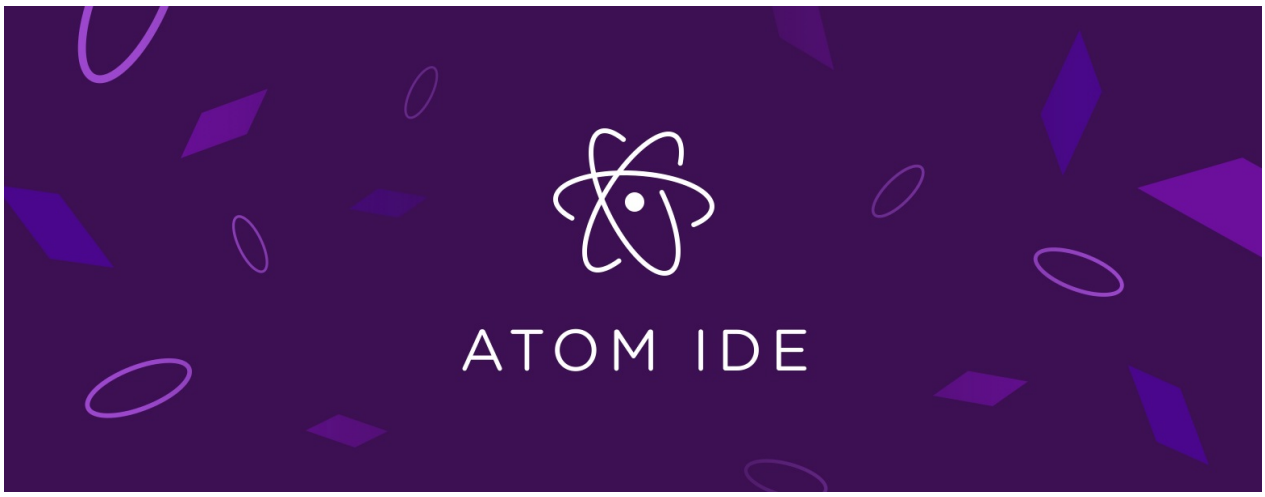
오픈 소스의 개념에 대해 알았다면 지금까지 우리가 흔히 사용해온 방식(예를 들어, 마이크로소프트)은 무엇일지 의문이 생길 것이다. 이를 바로 클로즈드 소스(Closed Source) 즉, 폐쇄형 소스라고 한다. 클로즈드 소스에 대한 설명은 아래와 같다.
폐쇄형 소스(또는 독점 소프트웨어)는 소스코드가 공개되지 않는 컴퓨터 프로그램을 의미한다. 누구나 볼 수 있고 수정할 수 있는 대중에게는 소스 코드가 공개되지 않는다. 이익 창출을 위해 소프트웨어를 판매하는 대부분의 회사는 클로즈드 소스를 사용하므로 사람들이 쉽게 변경하거나 무료로 복사할 수 없다. 클로즈드 소스라는 용어는 프로그램의 소스 코드가 공개되지 않는 "비공개 소스"와 혼동될 수 있다. 마이크로소프트의 공유 소스는 소스코드가 공개되긴 하지만 오픈 소스는 아닌 예이다. 다시 말해, "클로즈드 소스"라는 용어를 "오픈 소스"의 반대 개념으로 사용한다면 "공유 소스"는 "클로즈드 소스"의 한 형태에 해당된다.

오픈소스 프로젝트 사례

그렇다면 오픈소스 프로젝트의 사례에는 무엇이 있을까? 아래의 프로젝트들은 2016년 Top 10위 내에 든 프로젝트 중 일부이다.

- Atom

  > **Atom**은 자유-오픈 소스 형태의 OS X, 리눅스, Window용 문서 및 소스 코드 편집기이다. 대부분의 확장 패키지는 자유 소프트웨어 라이선스를 취하며 커뮤니티가 만들고 관리하고 있다. 아톰은 크로미엄에 기반을 두며 카피스크립트로 작성되어 있다. 초기에 아톰을 위한 확장 패키지와 아톰의 코어에 속하지 않는 사항들이 오픈 소스 라이선스로 공개되었다. 2014년 5월 6일, 코어 응용 프로그램을 포함한 아톰의 나머지 부분인 아톰의 패키지 관리자, 아톰의 크로미엄 기반 데스크톱 응용 프로그램 프레임워크, 일렉트론(Electron, 과거의 아톰 셸)이 MIT 라이선스 하의 자유 소프트웨어로 공개되었다.



- FreeCAD

  > **FreeCAD**는 파이썬으로 작성되었으며 실제 객체의 설계 사양을 작성하는 데 사용할 수 있는 많은 컴퓨터 지원 설계 또는 컴퓨터 지원 제도 도구 중 하나이다. FreeCAD는 3D 개체에 대한 다양한 일반 형식에서 가져오고 내보낼 수 있으며 모듈식 아키텍처를 사용하면 다양한 플러그인으로 기본 기능을 쉽게 확장할 수 있다.

- Kodi

> **Kodi**는 비영리 기술 컨소시엄인 XBMC 재단이 개발한 free-open source media player 응용 소프트웨어이다. 여러 운영 체제와 하드웨어 플랫폼을 지원하며 텔레비전과 리모컨 이용을 위한 인터페이스를 갖추고 있다. 사용자가 비디오, 음악, 팟캐스트, 인터넷의 비디오, 로컬 및 네트워크 스토리지 미디어의 모든 디지털 미디어 파일을 포함, 대부분의 스트리밍 미디어를 재생하고 볼 수 있다.

- Eclipse

> **Eclipse**는 상업적으로 친숙한 오픈 소스 소프트웨어에 대한 공동 작업을 원하는 개인 및 조직을 위한 커뮤니티이다. 이 프로젝트는 수명주기 전반에 걸쳐 소프트웨어를 구축, 배포 및 관리 할 수 있는 확장 가능한 프레임 워크, 도구 및 런타임으로 구성된 개방형 개발 플랫폼을 구축하는 데 중점을 둔다. 이클립스 재단은 이클립스 프로젝트를 주최하고 오픈 소스 커뮤니티와 보완 제품 및 서비스의 생태계를 육성하는 데 도움이 되는 비영리 단체 회원 지원 법인이다. 이클립스 프로젝트는 원래 2001 년 11 월 IBM에서 개발했으며 소프트웨어 벤더 컨소시엄의 지원을 받았다. 이클립스 재단은 2004 년 1 월 이클립스 커뮤니티의 청지기 역할을 하는 독립 비영리 단체로 설립되었다. 독립적인 비영리 단체는 Eclipse 주변에서 벤더 중립적이고 개방적이며 투명한 커뮤니티를 구축 할 수 있도록 만들어졌다. 오늘날 이클립스 커뮤니티는 소프트웨어 업계의 여러 분야에서 개인과 조직으로 구성된다.

오픈소스에 대해 궁금한 점

앞선 내용들 만으로는 오픈소스에 대해 가지고 있는 궁금한 점들이 해결되지 않았을 것이다. 아래의 대표적인 질문들로 조금이나마 궁금한 점을 해소할 수 있길 바란다.

- 자료를 오픈하는 것만이 오픈소스에 해당될까?

> 소스 코드를 공개한다고 해서 모두 오픈 소스인 것은 아니다. 예를 들어, 마이크로소프트는 전체 비율로 보았을 때 극소수의 고객(주로 정부나 거대 다국적 기업 또는 대학교 및 연구소)들에게 마이크로소프트 윈도우의 소스 코드를 공개했다. 오로지 보안 유지를 위해서만 소스 코드를 직접 수정할 수 있으며, 그 수정본을 재배포하는 것은 금지되어 있다. 이것은 오픈소스의 의의에 어긋나므로 이러한 경우는 오픈소스라고 부르지 않는다.

- 왜 오픈소스를 사용할까?

> 오픈소스의 미래 설문조사(Future of Open Source Survey)의 응답자 중 72%가 유료 소프트웨어들보다 보안이 더 강력하기 때문에 오픈소스를 사용한다고 답했다. 응답자의 80%는 품질이 좋기 때문이라고 답했다. 응답자의 68%는 오픈소스가 효율성을 증대하고 비용을 낮추는데 도움이 된다고 말했으며, 5%는 이 소프트웨어가 새로운 제품과 서비스를 만드는데 도움이 된다고 답했다. 더불어 오픈소스에 공개적으로 기부를 하고 받아들이고 있다고 답한 응답자도 50%였다.

- 오픈소스로 어떻게 수익을 낼 수 있을까?

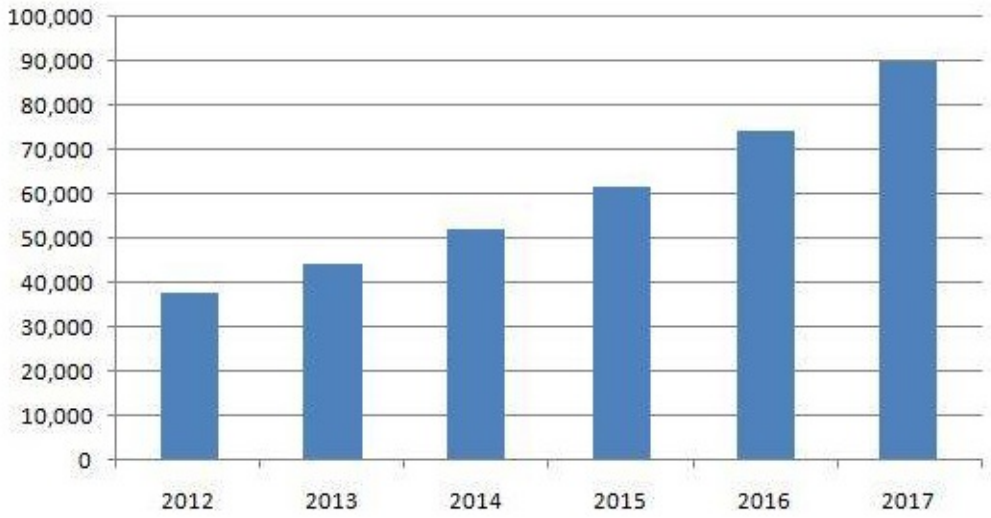
> 오픈소스 그 자체를 수익을 낼 수 있는 하나의 제품으로 보는 것이 아니라, 수익 흐름을 형성하는 촉매제로 보아야 한다. 수익을 내는 일반적인 방법은 오픈소스 프로그램과 유료 서비스를 나란히 제공하는 것이다. 그 예로는 대신 컴파일해서 실행 파일을 제공하거나, 유지 및 관리를 해주는 것, 그리고 강연이나 교육 등이 있다.

2. 오픈소스의 사용현황

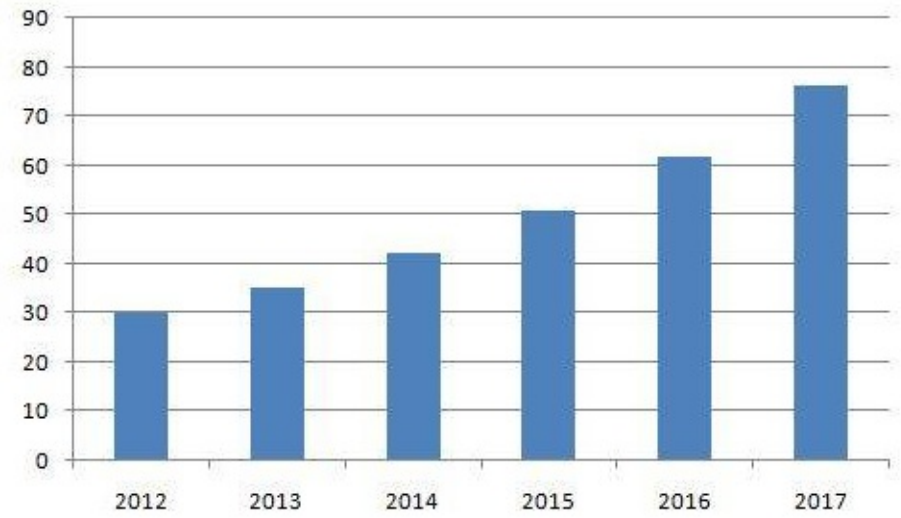
이제 오픈소스가 무엇인지는 알았다. 그렇다면 오픈소스를 도대체 왜 사용해야하는 것이며, 누가 얼마나 사용하는지, 과연 오픈소스를 사용하는 것이 좋은 것일지 궁금할 것이다. 지금부터 그에 대해 알아보자.

사용현황과 급증 이유

IDC에 따르면, 전세계 OSS 시장은 지난 2013년 445억 달러에서 오는 2017년 899억 달러의 시장을 형성할 것으로 전망되고 있다. 또한 2010년 7월부터 8월까지 2개월에 걸쳐 실시된 시장 조사 업체인 가트너의 연구 결과에 따르면 전 세계 11개 국가 547개의 IT 기업 중 22%에 해당하는 기업이 업무환경 전체에 오픈소스 소프트웨어를 사용하는 것으로 나타났다. 또 부서와 프로젝트별로 오픈소스 소프트웨어를 사용하는 기업은 전체의 46%, 오픈소스 소프트웨어 도입에 앞서 장단점을 고려하고 있는 기업은 21%에 달했다. 이는 연평균 18.8%씩 성장하는 추세이다.

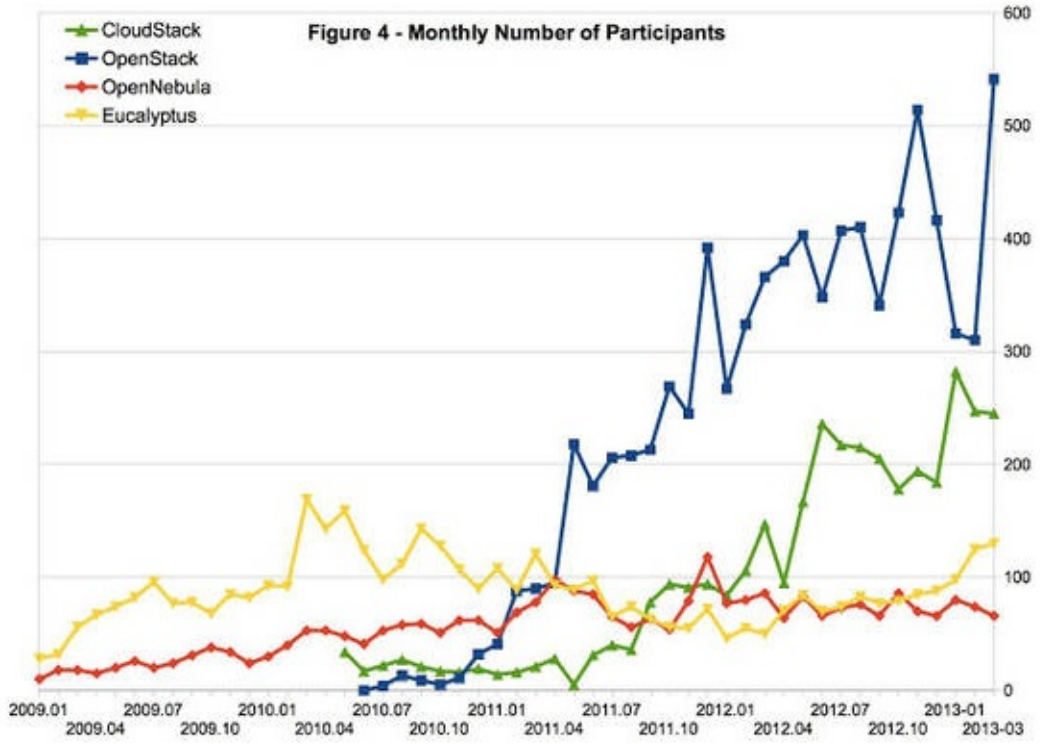


국내의 경우는 지난 2013년 352억원 규모를 형성했으며, 오는 2017년에 이르면 763억원 규모로 연평균 20.4%의 성장률을 기록할 것으로 전망되고 있다. 특히 국내의 OSS 활용은 대부분 비용 절감, 효율적인 비즈니스 환경 구현, 특정 벤더에 대한 종속성 탈피 등이 가장 큰 이유로 분석되고 있다. 특히 OSS가 기업 비즈니스에 특화돼 한층 더 경쟁력 있는 이익을 창출할 수 있다는 것이 근본적인 요인으로 작용하기 때문이다.



▲국내 OSS 시장 변화(그림=IDC)

그렇다면 왜 오픈소스를 사용할까? 우선 오픈소스 소프트웨어는 가격이 저렴하다. 소프트웨어에 대한 유지, 보수를 위해 일정한 금액을 지불해야하지만 상용 소프트웨어를 사용하는 것 보다는 훨씬 저렴하다. 자금력이 약한 기업들이 오픈소스를 선호하는 이유다. 또 데이터의 관리와 통합이 유리하고, 업무에 필요한 방향으로 신속, 정확하게 애플리케이션을 개발해 적용할 수 있고 비즈니스 프로세스 개선과 리엔지니어링이 용이하다. 보안 문제와 데이터 센터의 통합과 현대화 그리고 가상화에서도 우월하다. OSS의 활용 분야도 넓어지고 있다. 가장 활발한 곳이 OS 시장이다. 모바일 애플리케이션과 서비스의 확대로 인한 클라우드 서비스 활성화, 가상화 솔루션 도입으로 인한 리눅스 계열 OS 사용이 크게 증가하고 있다. 특히 안드로이드로 대표되는 모바일OS 시장에서의 활용도 크게 증가하고 있는 추세다. DB시장 역시 OSS가 활용되고 있는 대표적 사례다. 온라인 트래픽이 폭발적으로 늘어나고 있어 상용DB는 기업들에게 큰 부담으로 다가올 수밖에 없다. 때문에 큐브리드DB, SKY SQL 등의 OSS 기반의 DB 활용도도 커지고 있다. 클라우드 컴퓨팅과 빅데이터 역시 예외가 아니다. 아마존을 비롯해 구글 등 대표적인 클라우드 기업들은 OS를 활용해 클라우드를 구축했다. 또 HP와 델, IBM, 시스코 등의 글로벌 IT 기업들 역시 하둡, R 등과 같은 OSS와 오픈스택이라는 OSS를 기반으로 경쟁력을 더하고 있다. 특히 최근 ICT 분야에서 가장 각광을 받고 있는 사물인터넷(IoT) 분야에서도 OSS는 큰 역할을 담당하고 있다.



가트너 리서치 디렉터는 “오픈소스 소프트웨어를 통한 경쟁우위의 확보가 오픈소스 소프트웨어를 도입하는 중요한 이유로 주목받고 있기 때문에 단순히 가격 경쟁력에서의 우위를 점하는 것이 중요한 것이 아니라 기업의 요구조건에 부합하는 기능적인 측면을 더 강화해야 할 필요가 있다”고 말했다.

3. 오픈소스 라이선스

오픈소스 SW 개요

오픈소스SW는 소스코드가 공개되어 있는 SW를 말하며, 일반적으로 자유롭게 복제/배포/수정할 수 있다. 오픈소스SW의 대표적인 예로는 Linux 커널 및 아파치 웹서버, FireFox 웹브라우저, MySQL 등이 있다. 전 세계적으로 오픈소스SW는 FSF(Free Software Foundation)의 자유SW(Free Software)를 포함한 넓은 의미로 사용되고 있다. 하지만 자유SW와 오픈소스SW는 역사 및 추구하는 이념 등에서 미묘한 차이가 있다. 1980년대부터 소프트웨어가 거대 부가가치 산업으로 발전하자, 지식재산권 및 라이선스 계약을 통하여 소프트웨어의 복제, 배포, 수정에 제한을 가하려는 움직임이 나타났다. 이런 움직임에 반대하여 리처드 스톨만은 FSF를 설립하고 자유SW(Free Software) 운동을 전개하였다. 그러나 자유SW의 '자유(Free)'라는 단어가 일반인들에게 '무료'로 인식되고, 엄격한 GPL조항 때문에 상용SW개발에 이용할 수 없어 대다수 기업들이 자유SW운동에 참여하기를 꺼려하자 소스코드 공개에 보다 많은 참여를 이끌어내기 위하여 에릭 레이먼드, 브루스 페런스 등은 '오픈소스(Open Source)'라는 새로운 용어를 제안했다. 그리고 이러한 '오픈소스'는 1998년 오픈소스SW 활성화 및 오픈소스SW에 대한 인증을 담당하는 OSI(Open Source Initiative)가 결성되면서 널리 사용되기 시작했다. OSI는 오픈소스에 해당하는 라이선스의 최소한의 기준을 정의(Open Source Definition, OSD)해놓고 이 정의에 따라 인증, 관리 및 촉진시키는 일을 한다.

오픈소스 SW의 지식재산권과 라이선스

- SW 지식재산권 현재 SW는 다음과 같이 저작권, 특허권, 상표권, 영업비밀 등의 지식재산권에 의해 보호받고 있다.

저작권

> 저작권(copyright)은 창작물에 대하여 창작자(저작자)가 취득하는 권리로서 창작의 결과물을 보호 하며, 창작과 동시에 권리가 발생한다. 따라서 어떤 프로그래머가 특정 SW를 개발하면 컴퓨터 프로그램 저작권이 자동 발생하며, 그 권리는 프로그래머 또는 그가 속한 회사에 부여된다. 저작권이 있는 저작물의 경우 누구도 저작권자의 허락 없이는 해당 저작물을 쓸 수 없다.

특허권

> 특허권(patent)은 발명에 관하여 발생하는 독점적/배타적 지배권으로 법에 정해진 절차에 의해 출원을 하여야 하며, 심사를 통해 부여되는 권리이다. 특허기술을 사용하기 위해서는 반드시 특허권자의 허락을 얻어야만 한다. 특허 받은 방식을 구현하는 SW라면 프로그래밍 언어나 소스 코드와 상관없이 특허권자의 명시적인 허락을 받아야 한다.

상표권

> 상표권(trademark right)이란 상표권자가 지정상품에 관하여 그 등록상표를 사용할 독점적인 권리로서 일정한 절차에 따라 등록하여야 효력이 발생한다. 이러한 상표를 사용하기 위해서는 반드시 상표권자의 허락을 얻어야 하며 허락받지 않고 상표를 사용할 경우 처벌을 받게 된다. 상표권을 취득한 SW의 경우 상표를 사용하려면 상표권자의 명시적인 허락을 받아야 한다.

영업비밀

> 공개되지 않은 SW의 경우 영업비밀로서 보호를 받을 수 있으며, 공개된 SW라 하더라도 아이디어에 대한 부분은 영업비밀로 보호를 받을 수 있는 가능성이 있다. 단, 영업비밀로서의 SW 보호는 널리 공개되어 유통되는 경우에는 보호받기 어렵고, 이를 알지 못하고 사용한 제3자에게 법적으로 문제를 삼을 수 없다.

- 라이선스와 오픈소스 SW

라이선스의 의의

> 앞서 언급한 3가지에 의해 보호받으며 저작권자만이 쓸 수 있지만, 권리자가 다른 사람에게 일정한 조건으로 특정 행위를 할 수 있는 권한을 부여할 수 있다. 이와 같은 권한을 보통 '라이선스(license, 이용허락)' 라고 한다. 예를 들면, 우리가 윈도우즈를 구입하면, SW권리자인 마이크로소프트로부터 윈도우즈XP를 한 대의 컴퓨터에 설치하여 이용할 수 있는 라이선스 (권리)를 받은 것에 불과하다. 그러므로 윈도우즈 정품을 구입했다고 해서 다른 사람에게 빌려주거나 복제하여 팔 수 없다.

오픈소스 SW 라이선스

> 오픈소스SW 라이선스란 오픈소스SW 개발자와 이용자 간에 이용 방법 및 조건의 범위를 명시한 계약이다. 따라서, 오픈소스SW를 이용하기 위해서는 개발자가 규정한 라이선스를 지켜야 하며, 이를 위반할 경우에는 라이선스 위반 및 저작권 침해가 발생하고, 이에 대한 책임을 지게 된다. 이런 오픈소스SW 라이선스는 기본적으로 이용자의 자유로운 사용을 보장하고 있다. 오픈소스SW가 이와 같은 라이선스를 만들어서 운영하는 이유는 오픈소스SW를 이용하여 개발한 SW에 대해서도 법의 테두리 안에서 소스코드를 공개하도록 하기 위한 것이다. 2017년 05월 현재 오픈소스SW 라이선스의 인증을 관장하고 있는 OSI에 따르면 78개가 있다. 하지만 실제로 많이 사용되는 라이선스의 개수는 한정되어 있다. 오픈소스 프로젝트 개발 포털사이트인 Freshmeat(http://freshmeat.net)에 등록된 프로젝트 약 43,722개 중 약 72%가 GPL과 LGPL 라이선스이다.

- 오픈소스 라이선스의 이해와 활용

오픈소스SW는 독점SW(proprietary software)와 동일하게 저작권 등에 의한 법적 보호를 받고 있으며, 이와 같은 권리에 기반하여 오픈소스SW 저작권자는 오픈소스SW 이용자에게 라이선스를 부여한다. 그러나 오픈소스SW 라이선스는 일반적인 독점SW 라이선스와는 많은 점에서 차이가 있다. 이를 살펴보면 아래와 같다. 여기서 라이선시(Licensee)는 라이선스를 받는 자이고, 라이선서(Licenser)는 라이선스를 부여하는 자이다.

- 라이선시는 해당 오픈소스SW를 자유롭게 이용할 수 있다.
- 라이선시는 해당 오픈소스SW를 자유롭게 복제할 수 있으며, 일정한 조건하에 재배포할 수 있다.
- 라이선시는 해당 오픈소스SW를 자유롭게 수정하여 이용할 수 있으며, 일정한 조건하에 수정된 내용을 재배포할 수 있다.
- 라이선시는 해당 오픈소스SW의 소스코드를 자유롭게 획득하고 접근할 수 있다.

오픈소스SW 라이선스는 또한 SW 이용자에게 일정한 의무를 부과한다. 자세한 내용은 오픈소스SW와 함께 배포되는 라이선스를 통해 알 수 있다. 해당 오픈소스SW에 대한 라이선스는 주로 소스코드 내부나 홈페이지 등에 명시되어 있다. 이용자가 오픈소스SW 라이선스에서 요구하는 준수사항을 이행하지 않으면 권리자로부터 저작권법 위반(또는 계약 위반)으로 소송을 제기당할 수 있다. 패소할 경우, SW 배포는 불가능하며 손해 배상을 포함한 책임을 부담할 수 있다. 따라서 라이선스의 의무사항을 명확히 이해하여 이런 상황을 예방해야 한다. 그러나 독점SW 라이선스에서 규정된 의무사항과 비교하면 오픈소스SW 라이선스가 요구하는 내용은 결코 어렵지 않으며, 이를 잘 이해하고 준수하면 독점SW보다 훨씬 비용을 절감할 수 있다. 또한 몇몇 라이선스만이 독자 개발한 소스코드의 공개를 요구하고 있기 때문에 이를 잘 분석한 후 이용한다면 문제의 발생 소지는 거의 없다고 봐야할 것이다. 따라서 오픈소스SW를 다운로드받아 개발에 적용할 때는 반드시 라이선스의 요구 사항을 확인하여야 한다. 자체 판단이 불가능할 경우에는 외부 전문가에게 조언을 의뢰하여 개발 시작 전 해당 라이 선스의 요구 사항과 오픈소스SW의 이용목적을 확실히 분석하여야 한다. 이렇게 하는 것만으로도 충분히 올바르게 오픈소스SW를 최대한 활용할 수 있으며, 나중에 발생할 수 있는 문제들을 사전에 차단할 수 있다.

오픈소스 라이선스의 구체적 내용

오픈소스SW 라이선스의 의무사항은 각각의 라이선스마다 조금씩 차이가 있지만 크게 나누어 보면 공통적으로 '저작권 관련 문구 유지', '제품명 중복 방지', '서로 다른 라이선스의 SW 조합 시 조합 가능 여부 확인' 등이 있고, 선택적으로는 '소스코드 공개', '특허관련 사항 준수' 등이 있다.

1. 공통적 준수사항

2. 저작권 관련 문구 유지

> 저작권이란 표현된 결과물에 대해 발생하는 권리이며 저작물의 창작과 함께 자동적으로 부여된다. SW의 경우는 소스코드에 프로그램의 이름과 개발자, 버전, 연락처 등을 포함하고 있는 경우가 많으며 이러한 것들은 저작인격권으로 보호받는다. 오픈소스SW는 거의 대부분 소스코드 상단에 개발자 정보와 연락처 등이 기록되어 있으며 개발자 정보를 임의로 수정하거나 삭제하여서는 안된다. 특히 GPL 등 수정된 결과물을 다시 공개하도록 규정하고 있는 '상호주의(reciprocal)' 라이선스들의 경우 저작인격권으로 보호받기 때문에 소스코드상의 개발자 정보가 수정, 삭제된 채로 외부에 공개되면 저작권 침해문제가 발생할 수 있으므로 주의해야 한다. 저작권 관련 문구 유지는 쉽게 판단이 가능한 사항이므로 항상 준수하여야 한다.

- 제품명 중복 방지

> SW의 제품명은 상표권으로 보호받는다. 따라서 오픈소스SW의 경우에 이와 동일한 이름을 제품명이나 서비스명으로 사용하면 상표권 침해의 문제가 생기게 된다. 특히 유명한 오픈소스SW일수록 해당 오픈 소스SW의 이름이 상표로 등록되어 있는 경우가 많기 때문에(예: 리눅스) 더욱 조심해야 한다.

- 서로 다른 라이선스의 조합

> SW를 작성하고자 할 경우 기존에 만들어진 코드를 재사용하거나 결합하는 경우가 많은데, 이러한 때 결합되는 각 코드의 라이선스가 서로 상충되는 경우가 있다. 이러한 문제를 라이선스의 양립성(Compatibility)문제라고 한다. 서로 다른 라이선스로 배포된 오픈소스SW를 결합하는 경우 반드시 두 개의 라이선스가 서로 호환되는지를 확인하여야 한다.

1. 선택적 준수사항

선택적 준수사항은 라이선스에 따라 다르다. 자세한 사항은 오픈소스 라이선스 가이드의 라이선스별 준수사항 부분(3.2)를 참고하기 바란다.

라이선스 종류

라이선스의 종류는 매우 다양하다. 모든 라이선스에 대해 소개를 하기에는 너무 양이 방대하므로 널리 사용되는 라이선스를 추려 소개한다.

> GNU General Public License(GPL ; version 2.0과 3.0존재)
>
> - version 2.0
>   자유 소프트웨어 재단(OSF)에서 만든 자유 소프트웨어 라이선스다. 미국의 리처드 스톨만(Richard Stallman)이 GNU-프로젝트로 배포된 프로그램의 라이선스로 사용하기 위해 작성했다.
> - 컴퓨터 프로그램을 어떤 목적으로든지 사용할 수 있다.
> - 컴퓨터 프로그램의 복사를 언제나 프로그램의 코드와 함께 판매 또는 무료로 배포할 수 있다.
> - 컴퓨터 프로그램의 코드를 용도에 따라 결정할 수 있다.
> - 변경된 컴퓨터 프로그램 역시 프로그램의 코드와 함께 자유로이 배포할 수 있다.
>   이 라이선스는 위처럼 네 가지 조항을 명시하고 있다.
>   대부분의 소프트웨어에 대한 라이선스는 소프트웨어를 공유하거나 수정할 수 있는 자유를 금지하기 위해 고안되었다. 반면에 GNU 일반 공중 라이선스는 자유 소프트웨어를 공유하고 수정할 수 있는 자유를 보장하기 위해 의도되었다. 즉, 소프트웨어가 사용자 모두에게 자유롭게 이용될 수 있도록 하는 것이다. 이 일반 공중 라이선스는 자유 소프트웨어 재단의 소프트웨어 대부분을 비롯하여, 저작자가 이 라이선스의 사용을 지정한 기타 모든 프로그램에 적용된다. (자유 소프트웨어 재단의 소프트웨어 중 일부는 이 라이선스 대신 GNU 라이브러리 일반 공중 라이선스가 적용된다.) 누구나 자신의 프로그램에 이 라이선스를 적용시킬 수 있다.
> - version 3.0
>   자유 소프트웨어 재단(FSF)과 이 재단의 GNU 프로젝트에 의해 배포되며 GNU 소프트웨어에 적용되는 공개 소프트웨어의 대표적인 라이선스 체계이다. GNU GPL이라고도 하며, 저작권(copyright)의 반대라는 의미로 카피레프트(copyleft)라고도 한다. 라이선스 사용료나 사용상의 제약 조건을 자유롭게 하여 소프트웨어 유통을 활성화하기 위한 의도에서 출발한 것으로 GNU 소프트웨어로 공개되는 원시 부호는 누구나 변경 또는 일반 공중 라이선스(GPL)로 재배포하고, 이를 이용하여 상업적 웹 사이트를 구축할 수도 있다. 그렇다고 저작권의 완전한 포기를 의미하는 것은 아니어서 GPL의 기본 원칙과 공개하는 측이 정의한 바를 충실하게 따르도록 되어 있다. 1990년대에 마련된 GPL V2.0에 이어 2005년에 V3.0이 발표되었다. GPL 버전 3은 2007년 6월 29일에 발표되었다. 2005년 후반에 자유 소프트웨어 재단에서 GPL의 세번째 판을 개발할 것이라고 발표했다. 바뀐 점 중에서 가장 중요한 4가지를 말하자면, 소프트웨어 특허에 대처하는 것, 다른 라이선스와의 호환성, 어떤 부분의 원시 코드와 무엇이 GPL이 포함되어야 하는 원시 코드를 구성하는지와 디지털 제한 관리(Digital Restrictions Management)에 신경을 썼다.

GNU Library or Lesser General Public License (LGPL ; version 2.0과 2.1, 3.0 존재)

- version 2.1
  라이브러리는 공유하되 개발된 제품에 대해서는 소스를 공개하지 않고 상용SW 판매가 가능한 GPL보다 완화된 라이선스를 말한다. GNU 약소 일반 공중 라이선스의 이름으로 공표된 최초의 버전이다. 본 라이선스는 GNU 라이브러리 일반 공중 라이선스 버전2의 후속판으로 간주되기 때문에 버전 번호를 2.1로 붙인 것이다.
- version 3.0
  라이브러리는 공유하되 개발된 제품에 대해서는 소스를 공개하지 않고 상용SW 판매가 가능한 GPL 보다 완화된 라이선스를 말한다. 이 라이선스는 GNU 일반 공중 라이선스 버전 3에 추가된 추가 허용 사항들로 구성된다. GNU 약소 일반 공중 라이선스의 버전 3를 의미하며 GNU GPL은 GNU 일반 공중 라이선스의 버전 3을 의미한다.

BSD licenses

- New and Simplified BSD licenses (2013년 이후 OSI 라이선스 제외)
  캘리포니아 대학이 관장하고 있는 공개 라이선스 및 라이선스 문장. 유닉스(Unix) 의 양대 뿌리 중 하나인 버클리의 캘리포니아 대학에서 배포하는 공개 소프트웨어의 라이선스이다. GNU 자유 소프트웨어 재단의 일반 공중 라이선스(GPL)보다 훨씬 개방적인 4개항의 간단한 문구로 되어 있다. 그동안 sendmail을 비롯하여 수 많은 인터넷 관련 소프트웨어의 소스나 바이너리가 BSD 라이선스로 공개되어 소프트웨어 및 인터넷 발전에 기여한 바 크다. 이러한 정신은 FreeBSD, NetBSD, OpenBSD, BSDi 등 파생된 라이선스에서도 그대로 이어지고 있다.
- BSD 3-Clause
  원본 라이선스가 종종 "BSD-old"로 참고되어지고 있는만큼, 여기서 파생된3-조항 버전은 "BSD-new"로 불리기도 한다. 다른 이름으로는 "New BSD", "revised BSD", "BSD-3" 혹은 "3-조항BSD"가 있다. 참고로 "New BSD"라는 말을 쓰고는 있지만, BSD 라이선스의 가장 최신 버전은 아니다. 이 버전 이후 "Simplified BSD 라이선스"라고 알려진 BSD-2-Clause 버전이 나왔다. 2008년 1월 9일, OSI 이사회는 FreeBSD 등에 사용되며, 마지막의 "홍보 불가(no-endorsement)"조항을 삭제하여 결과적으로 MIT 라이선스와 거의 동등해진 "Simplified BSD 라이선스"를 승인했다.
- BSD 2-Clause
  좀 더 단순화된 버전의 BSD 라이선스가 사용되게 되었는데, 주로 알려진 이름은 "FreeBSD"이다. New BSD(3-조항) 라이선스와의 주된 차이점은 비보증 구문을 삭제하였다는 것이다.

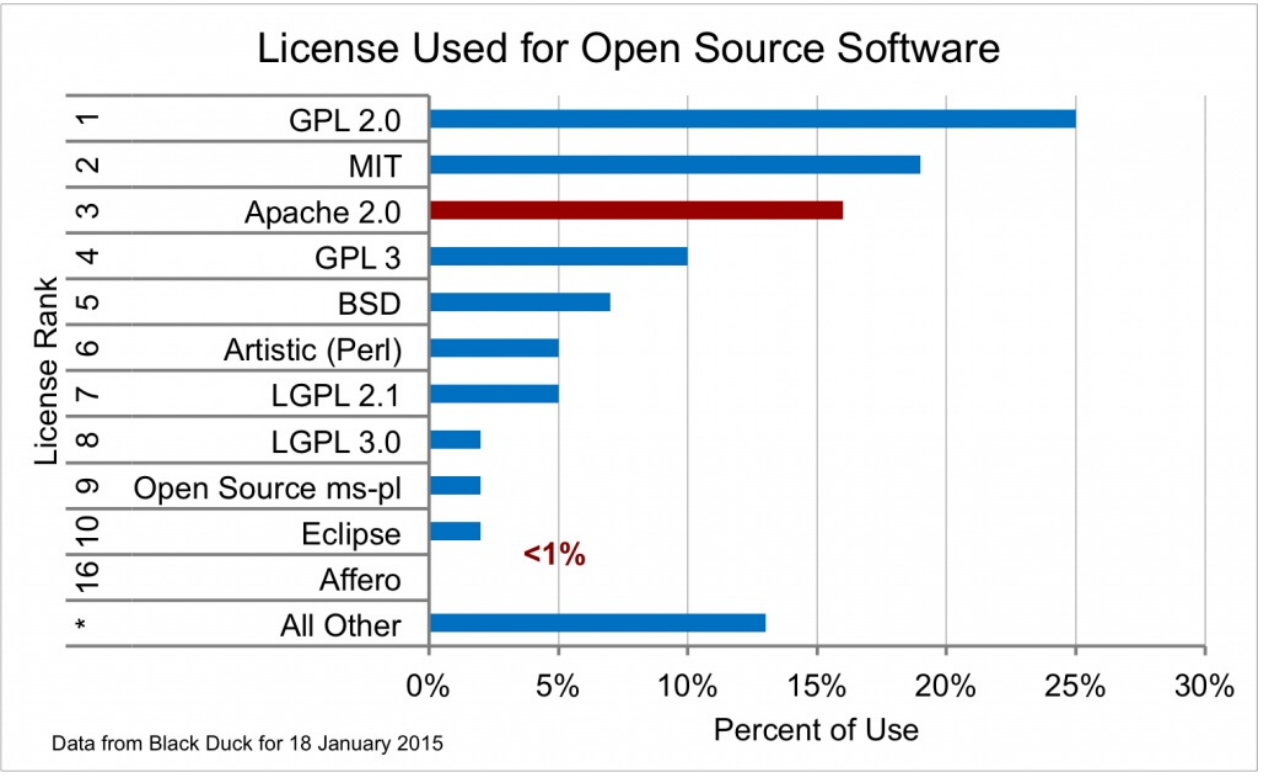
Apache License

- Apache Software License 1.1 (2013년 이후 OSI 라이선스 제외)
  2000년에 제정된 ASF 라이선스 1.1 버전. 2004년 2.0 버전이 나옴으로 2.0 버전으로 대체되는 라이선스이다. 최종사용자 문서에 다음과 같은 문구를 포함해야한다. “이 제품은 아파치 소프트웨어재단(http://www.apache.org/)에 의해 개발된 소프트웨어를 포함한다”

- Apache License 2.0
  고성능의 하이퍼텍스트 전송 규약(HTTP) 서버. 미국 일리노이 대학의 전미 슈퍼컴퓨터 응용 연구소(NCSA)에서 만든 'NCSA-httpd 1.3'이라는 전시용 프로그램을 근거로 기능 추가와 개량을 거듭해서 개발된 프로그램등을 대표하는 라이선스이다.

MIT License MIT 라이선스(MIT License)는 미국 매사추세츠 공과대학교(MIT)에서 해당 대학의 소프트웨어 공학도들을 돕기 위해 개발한 라이선스다. MIT 라이선스를 따르는 소프트웨어를 개조한 제품을 반드시 오픈 소스로 배포해야 한다는 규정이 없으며 GNU 일반 공중 라이선스의 엄격함을 피하려는 사용자들에게 인기가 있다. 이 라이선스를 따르는 대표적 소프트웨어로 X 윈도우 시스템이 있다.

Eclipse Public License Eclipse Public License (EPL)는 오픈소스라이선스로 Eclipse 재단에서 자사의 소프트웨어를 위해 사용된다. 이 라이선스는 Common Public License (CPL) 을 대체하며 특정용어등과 관련된 특허소송침해건을 제거했다. EPL은 GPL보다 약한 상호주의(copyleft)조항을 가지고 있어 기업친화적인 오픈소스 라이선스로 설계가 되었다.



" 오픈소스 라이선스 사용률 (2015) "

위에 소개한 라이선스는 오픈소스 라이선스 사용률 그래프에 명시된 상위 10개 중 일부이다.

Chapter 02. 오픈소스 역사

Section1 : 1980년대 이전의 역사

- 진정한 프로그래머 및 초기 소프트웨어
- 진정한 프로그래머
- 컴퓨팅 기술의 발달
- 기업에서의 초기 소프트웨어
- 유닉스 및 C언어의 탄생
- 소프트웨어 공유 문화의 위기
- 상업적 소프트웨어의 등장
- 소프트웨어의 사적 소유화
- 빌게이츠의 에세이 전문

1. 1980년대 이전의 역사

진정한 프로그래머 및 초기 소프트웨어

진정한 프로그래머

“태초에 진정한 프로그래머들이 있었다.”

오픈 소스 운동의 대표적인 인물인 에릭 레이먼드가 말했다.

하지만 초기 소프트웨어 포로그래머들은 스스로 자신을 그렇게 부르지 않았고, 그렇다고 그들에게 어떤 명칭이 있는 것도 아니었다.

‘진정한 프로그래머’ 라는 별명은 1980년 이후까지도 없었다.

그렇다면, 도대체 이 '진정한 프로그래머'란 누구를 칭하는 것일까?

에릭 레이먼드의 입장에서, 누구를 칭하는 것인지, 오픈소스의 역사를 알아보며 생각해보길 바란다.

컴퓨팅 기술의 발달

1945년 이후부터 *computing technology* 는 전 세계의 수많은 사람들을 유혹하였다.

소수 열정적인 프로그래머들에 의해 기술이 발전하였으며, 사람들은 소프트웨어를 재미로 만들고 즐기기 시작했다.

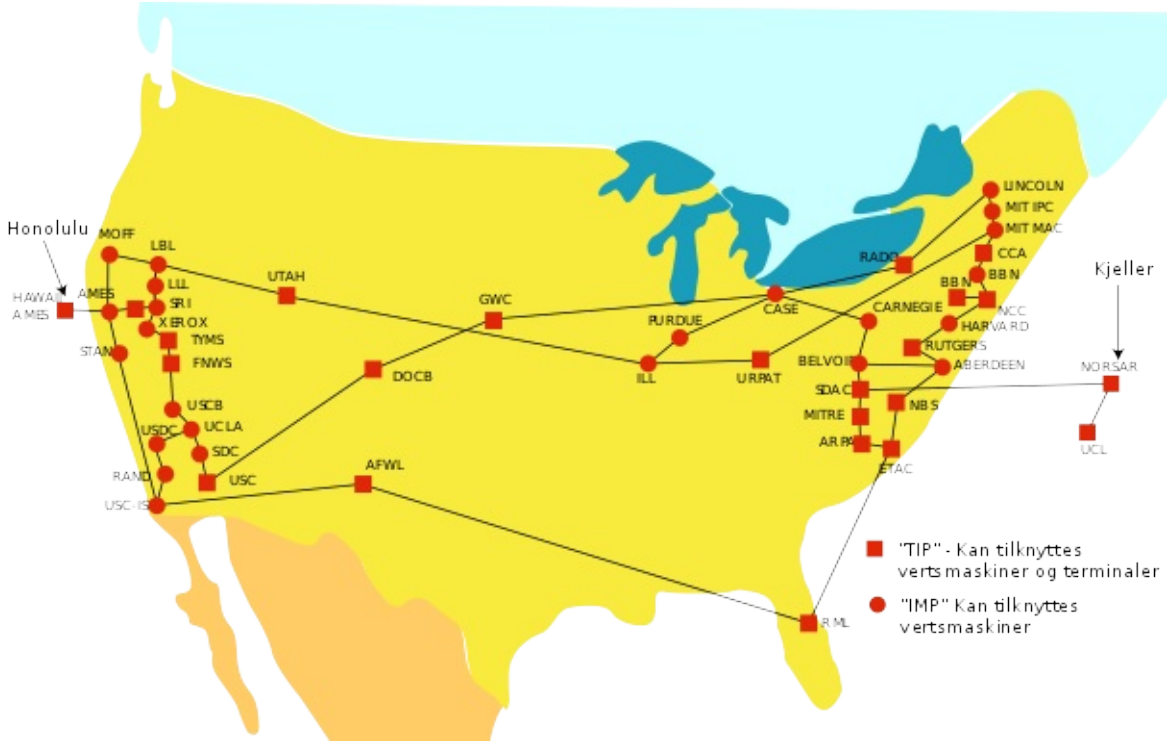
‘진정한 프로그래머’ 들은 공학이나 물리학을 배운 사람들이었다.

기계어, 어셈블리어등으로 코딩을 하였고 1970년대 초반까지 ‘진정한 프로그래머’ 들은 컴퓨팅에 있어 기술 문화를 주도하였다.

'진정한 프로그래머’ 문화가 해낸 것은 대화형 컴퓨팅, 대학, 네트워크의 발달이었다. 그들은 결국 오늘날의 오픈 소스 해커 문화로 진화할 공학의 전통을 탄생시켰다.

한편, 대학에서는 ‘지식 공유’라는 학문적 원칙을 따라 많은 컴퓨팅 기술을 생산하였고, 또 이에 대한 업그레이드들을 공개하고 공유했다.

네트워크 또한 MIT의 학생들의 지식 공유로부터 시작했으며, ARPAnet시스템을 사용하여 대륙 간 지식 공유를 활발하게 하였다.



미국에서 쓰인 "ARPAnet"

기업에서의 초기 소프트웨어

이는 기업에서도 다르지 않았다.

1950~1960년대 '진정한 프로그래머' 들의 공동 작업으로 탄생한 거의 모든 소프트웨어들은 public domain software로서 공유되었다.

오랫동안 학계에서 확립된 '개방과 협력의 원칙'에 따라 일반적으로 배포된 것이다. 소프트웨어의 소스코드 또한 배포되었는데, 이유에는 2가지가 있다.

- 첫번째 이유는 OS 혹은 하드웨어에서 실행되지 않아 소프트웨어를 사용자 스스로가 수정해야 할 일이 많았기 때문인 것.
- 두번재 이유는 사용자가 버그를 고치거나 새로운 기능을 추가할 수 있게 하기 위함이다.

무료 소프트웨어의 대표적인 예

이러한 무료 소프트웨어의 대표적인 예로는 1953년 개발된 A-2 System이 있다.

A-2 System은 컴파일러 프로그램인데, 소스코드와 함께 배포되었다. A-2 System은 최초의 자유 소프트웨어 및 오픈소스 소프트웨어이다.

IBM의 메인프레임 컴퓨터(대형컴퓨터)들도 소스코드가 포함된 채로 출시되었다.

또 다른 예로는 1975년 출시된 Tiny basic이 있는데, 일반적으로 사용되는 BASIC의 기능을 축소하여 메모리가 작아도 사용 가능하도록한 마이크로컴퓨터용 BASIC이다.

Tiny basic은 현재에도 사용되는 기기가 존재한다. (참고로, BASIC은 프로그래밍 언어이다.)

Tiny basic

유닉스 및 C언어의 탄생

1969년, 켄 톰슨 *Ken Thompson* 이라는 벨 연구소의 해커가 유닉스를 발명하였다.

1970년대 초에 AT & T는 정부 및 학술 연구자에게 유닉스 초기 버전을 무료로 배포했지만 이 버전은 배포하거나 수정 된 버전을 재배포 할 수 있는 권한이 없었기 때문에 현대의 의미에서 자유 소프트웨어가 아니었다.

데니스 리치 *Dennis Ritchie* 라는 해커는 톰슨의 유아기적 유닉스에서 사용하기 위해 'C'라고 불리는 새로운 언어를 발명하였다.

유닉스처럼 C도 즐겁고, 제한이 없고, 유연하게 설계되었다.

이 도구에 대한 관심은 벨 연구소 내에 퍼졌다. 톰슨과 리치는 연구소 내부에서 사용하기 위해 우리가 지금 사무 자동화 시스템이라고 부르는 것을 만들었고, 당시인 1971년에 강력한 추진력을 얻을 수 있었다.

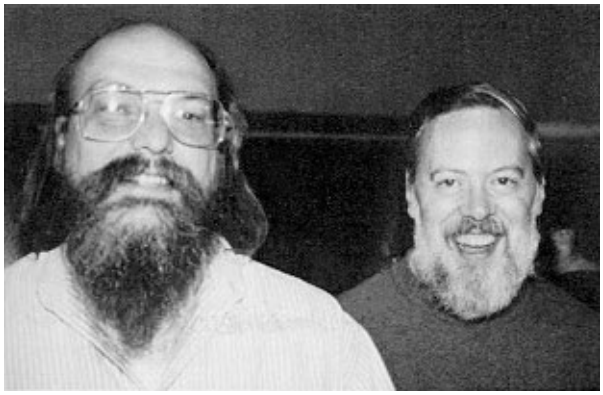
그러나 그들은 더 큰 목적에 관심이 있었다.

전통적으로 운영체제는 해당 호스트에서 최대한의 효율을 얻어내기 위해 어셈블리어로 작성했다.

하지만 톰슨과 리치는 하드웨어와 컴파일러 테크놀러지가 전체 운영체제를 C언어로 작성할 수 있을 만큼 좋아졌다는 사실을 인식한 것이었다.

이런 일은 일찍이 없었으며 이 사실이 암시하는 것은 대단한 사항이었다.

만약 유닉스가 서로 다른 형태의 기계에서 같은 모습, 같은 기능을 제공할 수 있다면 모두를 위한 공통 소프트웨어 환경으로 동작할 수 있을 것이고, 사용자들은 더 이상 기계가 쓸모없어질 때마다 소프트웨어를 새로 설계하기 위해 비용을 지불할 필요가 없을 것이기 때문이다.



"켄 톰슨과 데니스 리치"

소프트웨어 공유 문화의 위기

소프트웨어 공유 문화를 통해 점점 컴퓨팅 기술이 발전해감과 동시에, 이러한 공유 문화를 위협하는 사건들이 점점 발생하기 시작했다. 시작은 유닉스가 만들어진 연도와 같은 1969년이었다.

상업적 소프트웨어의 등장

1960년대 후반부터 운영 체제 및 프로그래밍 언어, 컴파일러가 발전하면서 소프트웨어 생산 비용이 하드웨어에 비해 크게 증가하기 시작했다.

그 당시 소프트웨어 산업은 IBM같은 하드웨어 제조 업체의 번들 소프트웨어 제품과 경쟁을 하고 있었고

번들 소프트웨어는 하드웨어의 가격에 포함되어 같이 팔리고 있었다.

소프트웨어 산업이 수익을 내지 못하고 있는 동안(초기 소프트웨어 들은 자유로운 수정과 배포가 가능했다)

컴퓨터를 임대해서 쓰는 사람들은 소프트웨어의 지원을 원했다.

그 중 소프트웨어의 발전과 지원이 더 많이 필요한 사람들은 하드웨어 제조업체가 하드웨어의 가격 안에

소프트웨어 비용을 포함하여서 판매하는 것을 원하지 않았다.

이렇게 진행이 되던 중에 1969년 1월 17일에 열린 미국과 IBM 반독점 소송에서 미국 정부는 번들 소프트웨어가 반경쟁적이라고 명시해버렸다.

이 소송 이후로 일부 소프트웨어는 무료로 계속 제공되었지만, 한편으론 제한적인 라이센스를 가진 소프트웨어가 점차 증가하였다.

소프트웨어의 사적 소유화

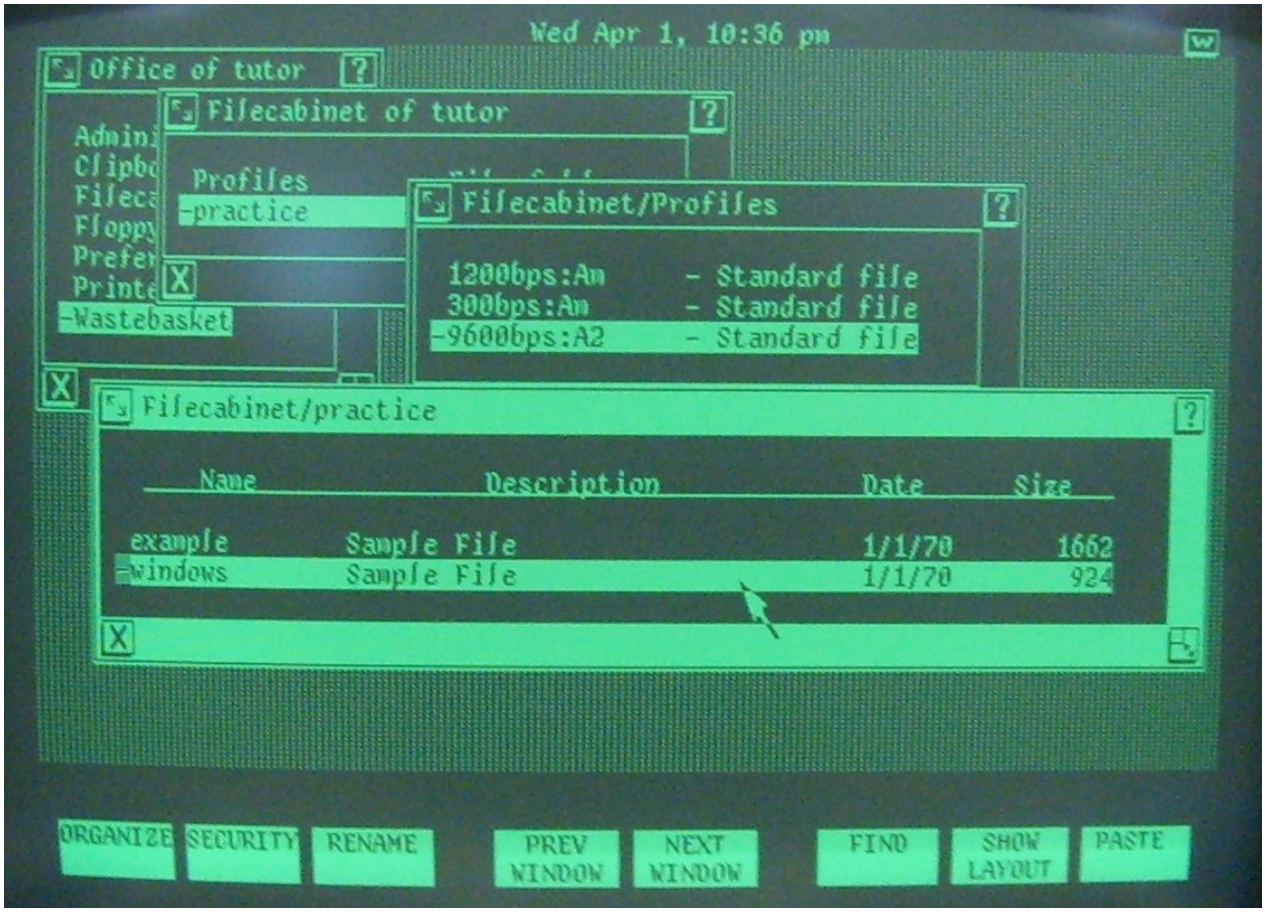
- **AT & T**

1970년대 초, AT & T는 정부 및 학술 연구자에게 UNIX 초기 버전을 무료로 배포했지만

이 배포 버전에서는 수정된 버전을 재배포하거나 배포할 수 있는 권한이 없었다.

UNIX가 널리 보급된 1980년대 초반, AT & T는 UNIX의 무료배포를 중단하고 시스템 패치를 부과했다.

다른 아키텍처로 전환하는 것이 매우 어렵기 때문에 이에 따라 무료로 배포를 받은 단체나 사람들은 상용 라이센스를 지불하며 UNIX를 사용하기 시작했다.



" The AT&T PC 7300 의 화면 "

" AT&T UNIX PC "

- **컴퓨터 벤더와 소프트웨어 전용 회사**

1970년대 후반에서 1980년대 초반, 컴퓨터 벤더와 소프트웨어 전용 회사는 소프트웨어 라이센스에 대한 비용을 부과하기 시작하였다.

바야흐로 소프트웨어를 프로그램 제품이라고 인식을 하기 시작한 마케팅이었다.

점차 컴퓨터 및 소프트웨어 전용 회사들은 소프트웨어에 대한 저작권, 상표 및 임대를 통해 소프트웨어를 자신으로 간주하기 시작하였고,

이러한 소프트웨어들에 대해 법적인 제한을 부과하기 시작했다.

- **IBM**

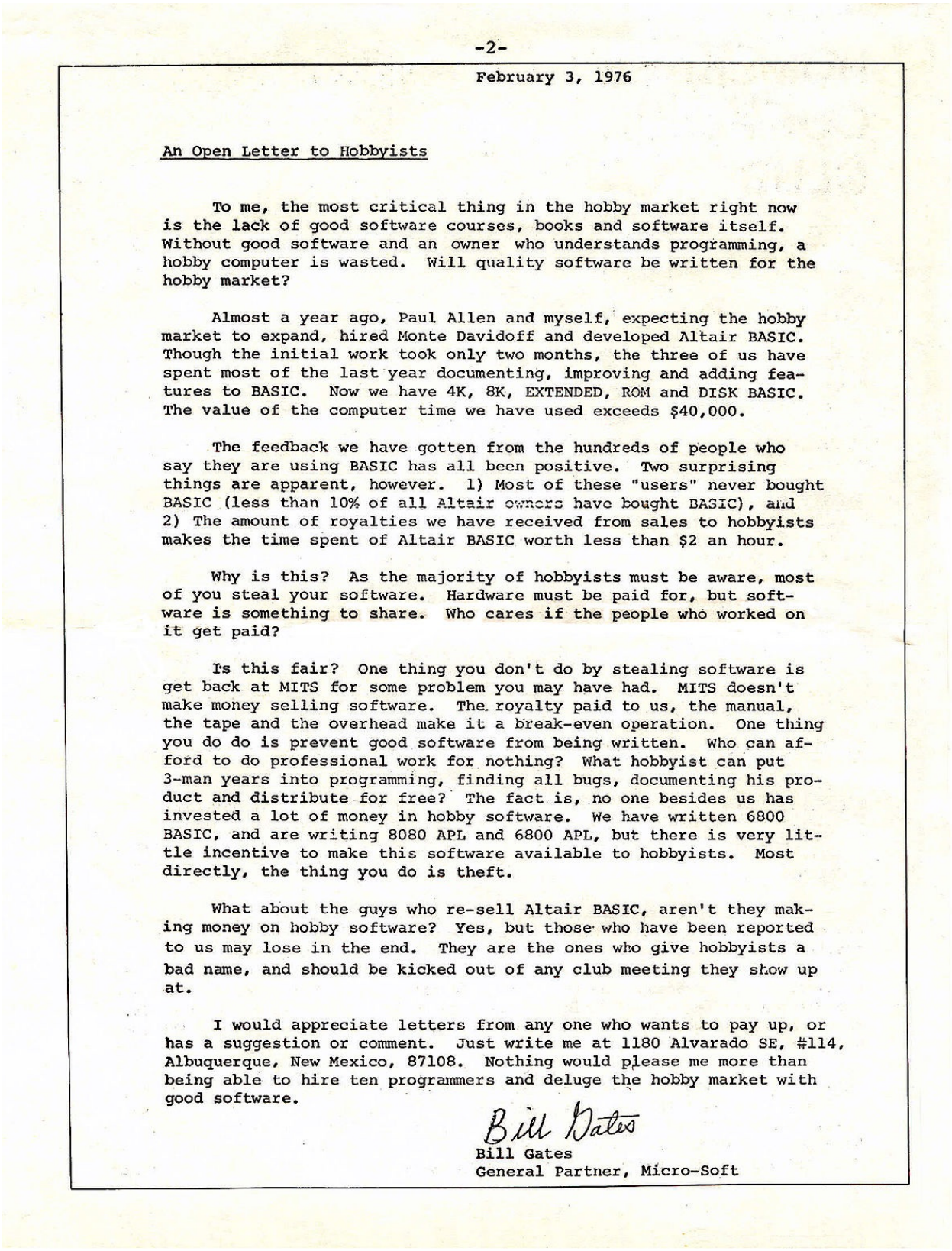
1983년 2월 8일에 IBM은 생산품 및 소프트웨어에 대해 새로운 정책을 발표하였다.

당시 발표는 새로운 생산품들과 현재 팔리고 있는 여러 생산품들의 업그레이드 버전의 소스코드의 접근을 제한한다는 내용이 포함되어 있었는데

이러한 IBM의 발표는 지금까지 IBM을 사용한 사용자들의 많은 불만을 야기하였다.

IBM policy draws fire 에서 보면, 1985년 기사이지만 1983년 시행한 IBM 정책에 대한 반응에 관한 기사를 볼 수 있다.

- **빌게이츠의 에세이**



-2-

February 3, 1976

An Open Letter to Hobbyists

To me, the most critical thing in the hobby market right now is the lack of good software courses, books and software itself. Without good software and an owner who understands programming, a hobby computer is wasted. Will quality software be written for the hobby market?

Almost a year ago, Paul Allen and myself, expecting the hobby market to expand, hired Monte Davidoff and developed Altair BASIC. Though the initial work took only two months, the three of us have spent most of the last year documenting, improving and adding features to BASIC. Now we have 4K, 8K, EXTENDED, ROM and DISK BASIC. The value of the computer time we have used exceeds $40,000.

The feedback we have gotten from the hundreds of people who say they are using BASIC has all been positive. Two surprising things are apparent, however. 1) Most of these "users" never bought BASIC (less than 10% of all Altair owners have bought BASIC), and 2) The amount of royalties we have received from sales to hobbyists makes the time spent of Altair BASIC worth less than $2 an hour.

Why is this? As the majority of hobbyists must be aware, most of you steal your software. Hardware must be paid for, but software is something to share. Who cares if the people who worked on it get paid?

Is this fair? One thing you don't do by stealing software is get back at MITS for some problem you may have had. MITS doesn't make money selling software. The royalty paid to us, the manual, the tape and the overhead make it a break-even operation. One thing you do do is prevent good software from being written. Who can afford to do professional work for nothing? What hobbyist can put 3-man years into programming, finding all bugs, documenting his product and distribute for free? The fact is, no one besides us has invested a lot of money in hobby software. We have written 6800 BASIC, and are writing 8080 APL and 6800 APL, but there is very little incentive to make this software available to hobbyists. Most directly, the thing you do is theft.

What about the guys who re-sell Altair BASIC, aren't they making money on hobby software? Yes, but those who have been reported to us may lose in the end. They are the ones who give hobbyists a bad name, and should be kicked out of any club meeting they show up at.

I would appreciate letters from any one who wants to pay up, or has a suggestion or comment. Just write me at 1180 Alvarado SE, #114, Albuquerque, New Mexico, 87108. Nothing would please me more than being able to hire ten programmers and deluge the hobby market with good software.

Bill Gates

Bill Gates
General Partner, Micro-Soft

" 빌게이츠의 Open Letter to Hobbyists "

1976년 빌게이츠는 *"Open Letter to Hobbyists"* 라는 제목으로 에세이를 썼다.

이 때 당시 애용자들(Hobbyists)은 자신의 소프트웨어나 제품들을 공유하고 있었는데 Microsoft의 Altair BASIC도 그 중 하나였다.

그는 이 에세이에서 애용자들(Hobbyists)이 라이센스를 비용하지 않고

Microsoft의 제품인 Altair BASIC 을 광범위하게 공유하는 모습 때문에 낙담했다고 썼었다.

빌게이츠가 보낸 에세이 전문

> AN OPEN LETTER TO HOBBYISTS

컴퓨터 애호가들에게 보내는 공개 편지

> By William Henry Gates III

윌리엄 헨리 게이츠 3세(빌 게이츠)

번역: Viz

> February 3, 1976

1976년 2월 3일

> An Open Letter to Hobbyists

컴퓨터 애호가들에게 보내는 공개 편지

> To me, the most critical thing in the hobby market right now is the lack of good software courses, books and software itself. Without good software and an owner who understands programming, a hobby computer is wasted. Will quality software be written for the hobby market?

제가 보기에, 지금 취미(컴퓨터) 시장에 있어서 가장 중대한 문제는 좋은 소프트웨어 교육과정과 서적, 그리고 좋은 소프트웨어 자체가 부족하다는 것입니다. 좋은 소프트웨어와 프로그래밍을 이해할 수 있는 사용자가 없는 한 (취미용)컴퓨터는 무용지물이 됩니다. 그럼 앞으로 좋은 프로그램이 취미(컴퓨터) 시장을 위해서 쓰여지게 될까요?

> Almost a year ago, Paul Allen and myself, expecting the hobby market to expand, hired Monte Davidoff and developed Altair BASIC. Though the initial work took only two months, the three of us have spent most of the last year documenting, improving and adding features to BASIC. Now we have 4K, 8K, EXTENDED, ROM and DISK BASIC. The value of the computer time we have used exceeds $40,000.

거의 일년 전, 폴 엘런과 저는 취미(컴퓨터) 시장이 성장할 것이라는 기대하에 몬테 다비도프를 고용하고 알테어 베이직을 개발했습니다. 초기 작업은 단지 두 달 밖에 걸리지 않았지만 저희 세명은 남은 일년 동안 베이직의 문서 작업과 성능 개선, 기능 추가를 해 왔습니다. 현재 저희는 4K, 8K의 확장된 ROM과 디스크 베이직을 완성했습니다. 그 동안 저희가 사용한 컴퓨터 시간의 가치는 4만 달러나 됩니다.

> The feedback we have gotten from the hundreds of people who say they are using BASIC has all been positive. Two surprising things are apparent, however, 1) Most of these "users" never bought BASIC (less than 10% of all Altair owners have bought BASIC), and 2) The amount of royalties we have received from sales to hobbyists makes the time spent on Altair BASIC worth less than $2 an hour.

저희들이 저희 베이직을 사용하고 있다는 수백명의 사용자들에게 받은 반응은 모두 다 긍정적인 것이였습니다. 그러나 두 가지 놀라운 점은 명백히, 1) 거의 모든 사용자들이 베이직을 구입한 적이 없다는 것이고(10% 미만의 알테어 사용자들이 베이직을 구입하였습니다) 2) 저희가 판매상으로부터 받은 저작권료는 그동안 저희의 알테어 베이직 개발 시간을 시간당 $2도 보상하지 못한다는 것입니다.

> Why is this? As the majority of hobbyists must be aware, most of you steal your software. Hardware must be paid for, but software is something to share. Who cares if the people who worked on it get paid?

왜 그렇습니까? 애호가 분들의 대부분에 해당하긴 하지만, 당신들은 당신들의 소프트웨어를 훔친 것입니다. 하드웨어는 꼭 구입해야 하지만 소프트웨어는 그냥 공유하는 것입니까? 아무도 소프트웨어를 만들 사람이 보수를 받는지는 신경 안쓰는 것입니까?

> Is this fair? One thing you don't do by stealing software is get back at MITS for some problem you may have had. MITS doesn't make money selling software. The royalty paid to us, the manual, the tape and the overhead make it a break-even operation. One thing you do do is prevent good software from being written. Who can afford to do professional work for nothing? What hobbyist can put 3-man years into programming, finding all bugs, documenting his product and distribute for free? The fact is, no one besides us has invested a lot of money in hobby software. We have written 6800 BASIC, and are writing 8080 APL and 6800 APL, but there is very little incentive to make this software available to hobbyists. Most directly, the thing you do is theft.

그게 공평합니까? 당신이 소프트웨어를 훔치지 않는다면 당신은 오직 당신이 겪는 문제에 대해서 MITS에 화풀이 할 수 밖에 없을 것입니다. MIT의 문제 많은 공개 프로그램만을 쓰면서 불평할 수 밖에 없을 것입니다. MITS는 소프트웨어를 돈을 받고 팔지 않기 때문입니다. 저희에게 지급되는 저작권료는 매뉴얼, (프로그램이 저장된)테이프, 그리고 경비를 겨우 매꿀 수 있을 뿐입니다. 당신이 하고 있는 일(무단복제)는 좋은 프로그램이 만들어지지 못하게 하고 있습니다. 누가 아무 보답 없이 전문적인 일을 할 수 있겠습니까? 어떤 애호가가 세 명이 일년동안 프로그래밍하고 버그를 고치고 문서화 작업을 한것을 공짜로 배포하겠습니까? 저희 말고는 누구도 막대한 양의 돈을 (취미용)소프트웨어에 투자한 적이 없는게 사실입니다. 저희는 6800 베이직을 만들었고 8080 APL과 6800 APL을 만들고 있지만 이걸 애호가 분들에게 제공해 드릴 만한 의욕이 별로 없습니다. 정말 솔직하게 말해서, 당신들이 하고 있는건 절도 행위입니다.

> What about the guys who re-sell Altair BASIC, aren't they making money on hobby software? Yes, but those who have been reported to us may lose in the end. They are the ones who give hobbyists a bad name, and should be kicked out of any club meeting they show up at.

알테어 베이직을 (불법적으로 복재해서) 재판매하고 있는 자들은 어떻습니까? 그들은 (취미용)소프트웨어을 통해서 돈을 벌고 있지 않은가요? 그렇지만 저희들에게 신고된 자들은 마지막엔 패배(재판에서 패소)할 것입니다. 그들은 컴퓨터 애호가들의 이름에 먹칠을 하고 있으며 그들이 참가하는 모든 클럽 모임에서 추방되어야 할 것입니다.

> I would appreciate letters from any one who wants to pay up, or has a suggestion or comment. Just write to me at 1180 Alvarado SE, 114, Albuquerque, New Mexico, 87108. Nothing would please me more than being able to hire ten programmers and deluge the hobby market with good software.

저희에게 소프트웨어 값을 지불하실 의향이 있거나, 건의사항, 의견이 있으신 분은 편지를 보내주시면 감사하겠습니다. 1180 Alvarado SE, 114, Albuquerque, New Mexico, 87108 주소의 제 이름으로 보내주시면 됩니다. 저에게 열 명의 프로그래머를 고용해서 취미(컴퓨터) 시장에 좋은 소프트웨어를 쏟아낼 수 있게 해주신다면 그 것보다 행복한 일이 없을 것입니다.

> Bill Gates

빌 게이츠

> General Partner, Micro-Soft

무한 책임 사원(사장), 마이크로-소프트

Chapter 02. 오픈소스 역사

Section2 : 1980년대 역사

- 위기를 넘어 계속적으로 이어지는 소프트웨어의 공유
- 비공식적인 소프트웨어의 공유
- 자유 소프트웨어 운동의 시작
- GNU(GNU is Not UNIX)프로젝트
- 자유 소프트웨어 운동
- 1980년대 온라인 소프트웨어 공유 커뮤니티
- 설명과 예시

2. 1980년대 역사

소프트웨어 공유에 위기가 닥치던 시기

이렇게 중대한 시기에 어떻게 소프트웨어가 공유되었는지와 자유소프트웨어운동이 어떻게 전개되었는지에 대해 살펴볼 것이다.

위기를 넘어 계속적으로 이어지는 소프트웨어의 공유

비공식적인 소프트웨어의 공유

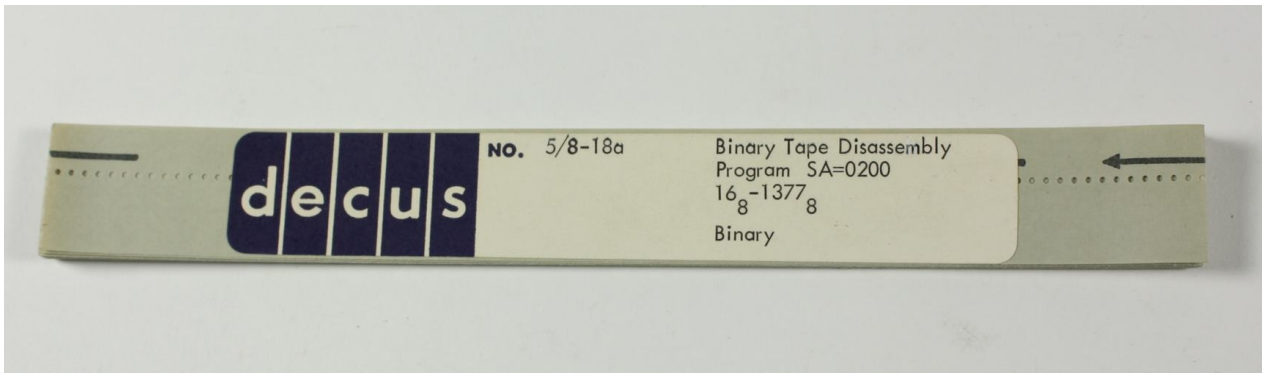
이렇게 기업들이 라이센스와 비용을 부과해도 계속적으로 소프트웨어를 공유하는 사람들이 있었다.

- **DECUS tapes**

1980년대 초, DEC 사용자들을 위해 만들어진 자유 소프트웨어 전송 시스템이다.

당시 *the TECO editor*, *Runoff text formatter*, *List file listing utility* 같은 프로그램들이 이 DECUS tapes에서 배포되고 전송되었다.

이렇게 DECUS tapes에서 배포되고 전송되는 유틸리티 패키지는 DEC의 작동 시스템에 많은 도움을 주었다.



" DECUS tapes "

- **hobbyists, hackers**

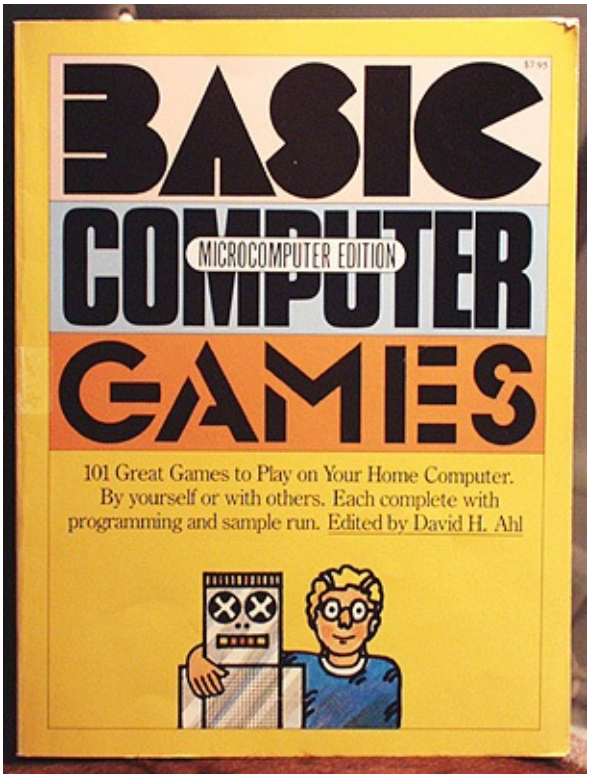
소스코드를 다른 프로그래머들이나 사용자들에게 무료로 공유하고 싶어하는 사람들이 존재했다.

이러한 사람들은 "hobbyists" 또는 "hackers" 라고 불렸다.

아직 인터넷이 잘 소개되지도 않았고, 널리 사용되지도 않았기 때문에 "hobbyists" 들은 여러가지 방법으로 소스코드들을 공유하였는데,

컴퓨터 잡지나 컴퓨터 프로그램 책들을 이용하기도 하였다.

당시에 쓰던 컴퓨터 잡지인 " Compute1 "



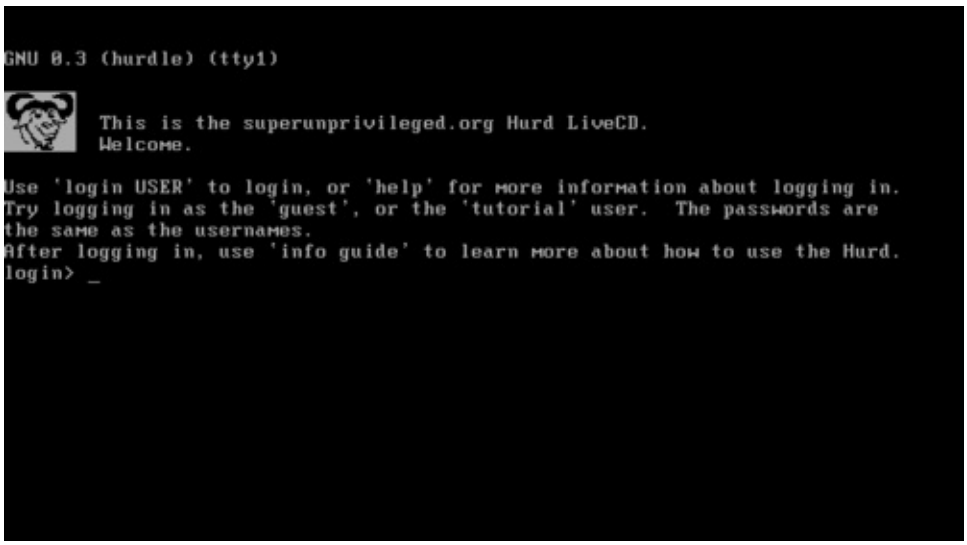
컴퓨터 책 " BASIC computer games "

자유 소프트웨어 운동의 시작

소프트웨어 기업의 소프트웨어에 대한 이용·배포·복제·수정 등에 일정한 제한을 가하려고 하는 추세와

독점 체제에 반발하여 **‘공유’** 를 주장하는 운동이 일어났다.

GNU(GNU is Not UNIX)프로젝트



" GNU 허드로 가동된 모습 "

리차드 스톨만은 소프트웨어 상업화에 반대하고 소프트웨어 개발 초기의 상호협력적인 문화로 돌아갈 것을 주장하며 1983년 GNU (GNU is Not UNIX) 프로젝트를 주도했다.

- **배경**

이 프로젝트가 시작된 이유는 여러가지가 있지만 다소 엉뚱한 사건도 조금 포함이 되는데 다음과 같다.

> 리처드 스톨만은 소스코드가 사용자(소비자)들에게 제한이 되어있는 프린터를 사용했었다. 이 프린터를 사용하다 보니까 조금씩 프린터가 말썽을 부리는데 이것을 수정을 할 수가 없어서 짜증이 났던 사건이다.

- **내용**

이 프로젝트에서 리처드 스톨만과 스톨만에 동조하는 프로그래머들은 GNU 프로젝트 하에서

유닉스와 호환되는 시스템 소프트웨어를 만들기 시작했고 개발된 GNU 소프트웨어는 모두 무료로 개방했다.

그리고 프로그래머라면 누구나 GNU의 소프트웨어를 가져다 수정하고 덧붙이고 개량해 다시 사람들에게 개방할 수 있었다.

- **GNU 선언문**

리처드 스톨만이 1985년 3월, GNU 프로젝트의 목표를 설명하고 다른 사람들의 참여와 지원을 요청하기 위해

닥터 돕스 저널(Dr. Dobb's Journal of Software Tools 10)에 실은 글이다.

이 선언문이 발표될 당시 GNU 선언문의 목표에 대해 여러가지 반대 의견도 존재했었다.

GNU 선언문의 내용은 처음에는 프로젝트의 진행에 따라 조금씩 바뀌었지만 현재는 있는 그대로를 보존하는 것이 낫다는 판단 때문에 그대로 유지되고 있다.

그 중에 일부를 소개하자면 다음과 같다.

> '소프트웨어는 공유돼야 하며 프로그래머는 소프트웨어로 돈을 벌어서는 안 된다'

자세한 내용은 GNU 선언문 한국어판에서 확인할 수 있다.

자유 소프트웨어 운동

리처드 스톨만이 시작한 운동이며 소프트웨어 사용자들의 자유를 보장하는 것을 목적으로 둔 사회 운동이다.

- **내용**

다음과 같은 내용을 포함하며 주장한다.

1. 사용자들은 소프트웨어 실행에 자유가 있다
2. 소프트웨어의 연구 및 수정이 자유롭다.
3. 변경사항의 유무의 관계없이 소프트웨어를 재배포 시킬 수 있다.
4. **FSF(Free Software Foundation)**

1985년, 리처드 스톨만이 설립한 자유 소프트웨어 재단이다.

자유 소프트웨어 운동을 지원하기 위해 설립을 했으며, 저작권(copyright)에 대응하는 카피레프트(copyleft) 운동도 같이 주창했다.

사이트 : FSF (Free Software Foundation)

- **철학**

사회 운동의 일환으로써 철학을 포함하고 있다.

여기에 모두 담을 수는 없지만 일부의 내용을 말하자면 이렇다.

1. 컴퓨터의 사용이 사람들이 서로 협력하는 것을 방해하면 안된다.
2. 독점 소프트웨어와 같은 제한을 거부한다.
3. 모든 소프트웨어의 사용자들은 자유 소프트웨어 정의(The Free Software Definition)에 열거된 자유를 가져야한다.

1980년대 온라인 소프트웨어 공유 커뮤니티

자유 소프트웨어 운동이 진행되는 한편, 온라인에서 소프트웨어를 공유하는 커뮤니티가 생겨나기 시작했다.

주로 이러한 커뮤니티들은 **BBS networks** 를 통해서 소프트웨어를 공유하였는데,

이렇게 공유가 되던 소프트웨어들은 BASIC 및 다른 인터프리터 언어로 작성 되어있어서 소스 코드도 함께 배포할 수가 있었으며

프리웨어, 즉 공짜로 쓸 수 있는 소프트웨어들이었다.

커뮤니티에서 활동하던 사람들은 이러한 소프트웨어들의 소스 코드를 수집하고, 수정하고 논의하는 활동들을 많이 했다.

예시

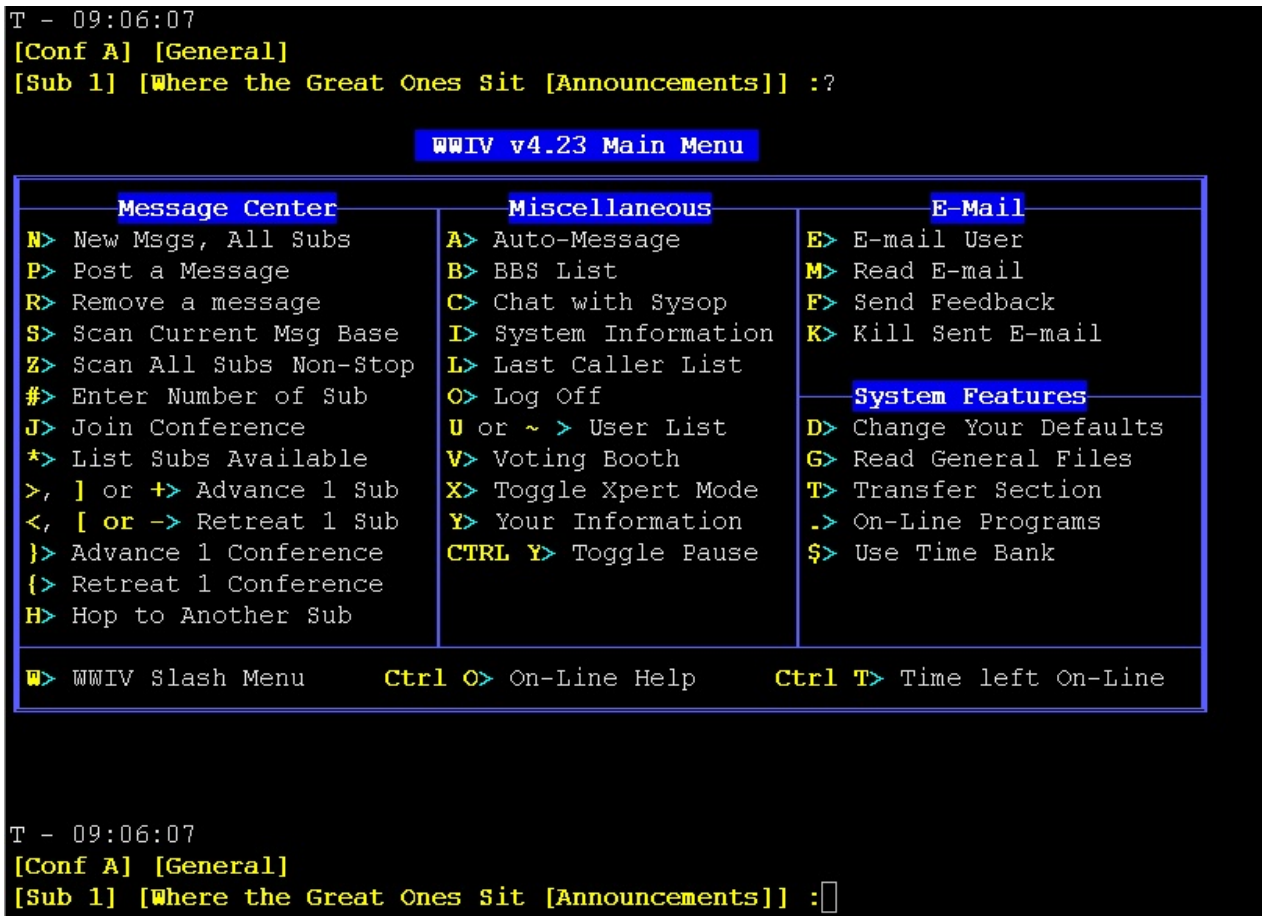
- **WWIV**

1980 년대 후반부터 1990 년대 중반까지 인기를 끌던 BBS 시스템 및 네트워크 중 하나이다.

초기에 Wayne Bell이 BASIC으로 개발을 하였다.

이 **WWIV** 에서 소프트웨어를 *"modding"* 하고 또 그 *"mods"* 를 배포하는 문화가 발달하여서 처음에는 *BASIC* 으로 포팅이 되었으나(ported), 후에는 *C++* 로 포팅이 되었다.

소스 코드는 등록된 사용자들에게 배포되어, 사용자들이 직접 그 소프트웨어를 공유하고 컴파일하여 새로운 버전의 소프트웨어를 다시 만들기도 하였다.


T - 09:06:07
[Conf A] [General]
[Sub 1] [Where the Great Ones Sit [Announcements]] :?
WWIV v4.23 Main Menu
Message Center
N> New Msgs, All Subs
P> Post a Message
R> Remove a message
S> Scan Current Msg Base
Z> Scan All Subs Non-Stop
#> Enter Number of Sub
J> Join Conference
*> List Subs Available
>,] or +> Advance 1 Sub
<, [or -> Retreat 1 Sub
}> Advance 1 Conference
{> Retreat 1 Conference
H> Hop to Another Sub
Miscellaneous
A> Auto-Message
B> BBS List
C> Chat with Sysop
I> System Information
L> Last Caller List
O> Log Off
U or ~ > User List
V> Voting Booth
X> Toggle Xpert Mode
Y> Your Information
CTRL Y> Toggle Pause
E-Mail
E> E-mail User
M> Read E-mail
F> Send Feedback
K> Kill Sent E-mail
System Features
D> Change Your Defaults
G> Read General Files
T> Transfer Section
.> On-Line Programs
$> Use Time Bank
W> WWIV Slash Menu
Ctrl O> On-Line Help
Ctrl T> Time left On-Line
T - 09:06:07
[Conf A] [General]
[Sub 1] [Where the Great Ones Sit [Announcements]] :

" WWIV 화면 "

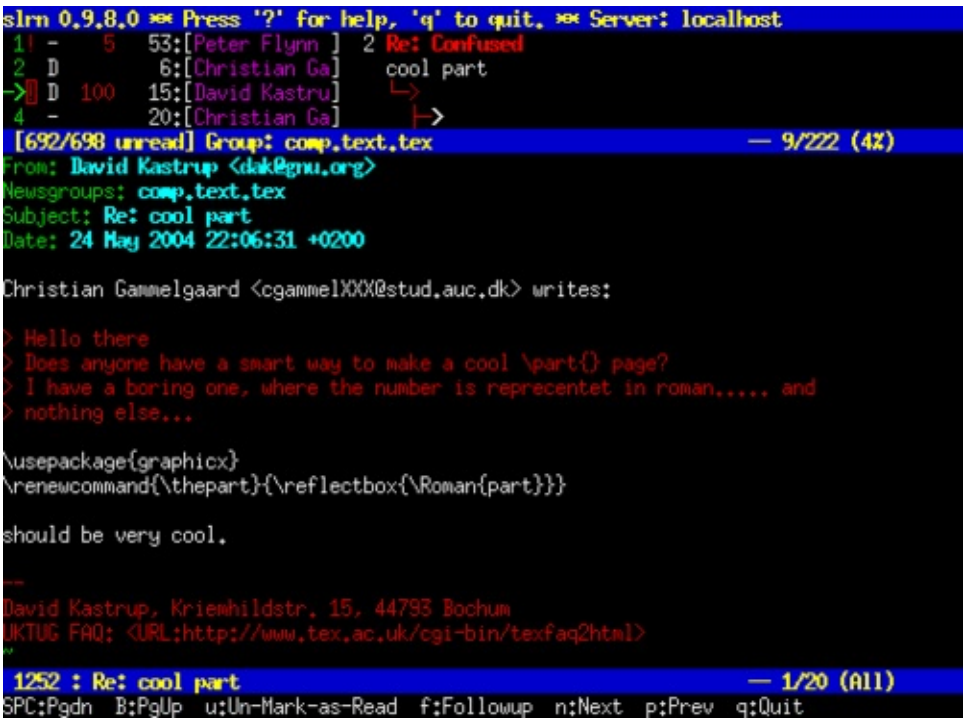
- **Usenet**

유즈넷(usenet)은 인터넷을 이루는 한 가지로, 유저 네트워크의 줄임말인데,

주로 텍스트 형태의 기사들을 전 세계의 사용자들이 공개된 공간에서 주고 받아 토론할 수 있게 고안된 분산 네트워크이다.

유즈넷의 출현으로 프로그래밍 단체가 더욱 더 활성화되었고,

프로그래머들이 소프트웨어를 공유하고 다른 사람이 작성한 소프트웨어에 더 쉽게 기여할 수있게 되었다.



" Usenet 화면 "

Chapter 02. 오픈소스 역사

Section3 : 1990년대 역사

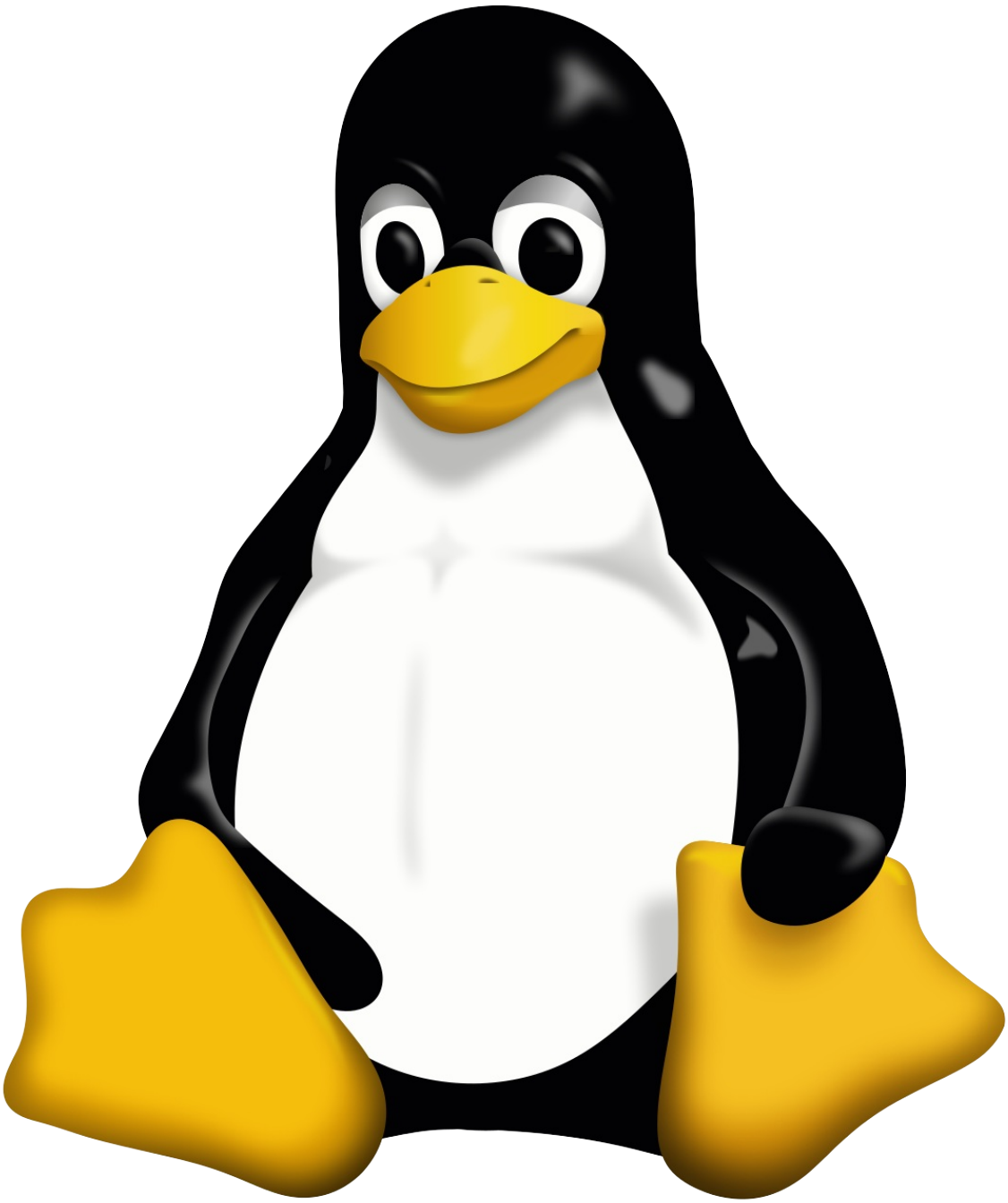
- 리눅스
- .com(닷컴)
- 오픈소스
- 오픈소스 이니셔티브
- 에릭 S 레이몬드

3. 1990년대

1980년대에는 사람들이 코드를 공유하고 리처드 스톨만이 자유 소프트웨어 운동과 FSF를 설립하며 오픈소스의 기반을 다지지만 정식으로 오픈소스라는 이름을 가진 활동은 존재하지 않았다.

이번 1990년대에 오픈소스라는 이름이 등장하게 되는데, 이번에 함께 소개되는 리눅스, .com, 에릭 S 레이몬드, 오픈소스이니셔티브와 함께 읽으며 그 의미를 알아갔으면 좋겠다.

리눅스



- 리누스 토발즈(Linus Torvalds)가 만든 컴퓨터 운영 체제, 혹은 그 커널을 의미하며, 리눅스는 다중 사용자, 다중 작업(멀티태스킹), 다중 스레드를 지원하는 네트워크 운영 체제(NOS)이다.

> 엄밀하게 따지면 이 '리눅스'라는 용어는 리눅스 커널만을 뜻하지만, 리눅스 커널과 GNU 프로젝트의 라이브러리와 도구들이 포함된, 전체 운영 체제(GNU/리눅스라고도 알려진)를 나타내는 말로 흔히 쓰인다.

- 만들어진 배경

엔드류 스튜어트 타넨바움 교수가 만든 교육용 유닉스인 미닉스를 사용하던 중, 교수가 미닉스를 다른 사람이 개조하지 못하게 제한을 두자 리누스 토발즈가 만들게 되었다.

- 내용

리눅스 커널은 자유 소프트웨어 프로젝트를 채택해 소스 코드가 사람들에게 공개되었고 많은 개발자들을 모았다.

기존에 있던 GNU 프로젝트의 GNU 허드보다 좋은 평가를 받아 개발자들은 리눅스 커널을 GNU 허드보다 더 선호하였다.

초기 리눅스는 개개인의 애호자들이 광범위하게 개발하였다.

이후 리눅스는 IBM, HP와 같은 거대 IT 기업의 후원을 받으며, 서버 분야에서 유닉스와 마이크로소프트 윈도 운영 체제의 대안으로 자리잡았다.

후에 GNU 운영 체제와 리눅스 커널의 결합이 되었는데, 1991년에 최초의 완전한 자유 소프트웨어 운영체제가 되었다.

지금 현재, 리눅스에서 리누스 토르발스는 리눅스 개발의 약 2%만큼 기여했다고 한다.

> 리눅스 커널
>
> 리눅스 커널은 1991년에 리누스 토르발스에 의해 생긴 말이다. 일찍이 미닉스 커뮤니티가 리눅스 커널에 코드와 개념을 제공하였다. 그 당시 GNU 프로젝트는 자유 소프트웨어 운영 체제에 필요한 요소를 많이 만들어 냈지만 자체 커널 GNU 허드는 완전하지 않았고 이용성이 없었다. BSD 운영 체제는 법적 문제로부터 헤어나오지 못했다. 이는 초기 버전의 제한된 기능에도 불구하고 리눅스가 새로운 운영 체제를 사용하기 위한 프로젝트로부터 코드를 채용한 개발자들과 사용자들을 빠른 속도로 모았다는 것을 말해 준다.

커널이란*?*

운영 체제의 핵심이 되는 부분

Unix 계열 운영 체제의 커널인 리눅스 커널은 *1994*년에 *GNU* 일반 공중 사용 허가서 버전*2* 아래에서 공개된다. 대표적인 오픈 소스의 본보기이다.

> 운영 체제로 리눅스 커널을 쓰는 기업, 제품
>
> - Google Android
> - Firefox OS
> - Nokia Maemo

.com (닷컴)

90년대 중반에는 자유 소프트웨어로 웹 서버를 만들었다. 자유 소프트웨어인 Apache HTTP Server는 가장 많이 사용되는 웹 서버 소프트웨어가 되었다.

오픈 소스

1997년, 에릭 S 레이몬드(Eric S. Raymond)(이하 레이몬드)가 좋은 오픈 소스 소프트웨어를 만들기 위한 교훈이 담긴 책인 ‘The Cathedral and the Bazaar’을 출판하고 그 책이 주목을 받으면서 Netskape Communications Corporation이 인터넷 제품군을 자유 소프트웨어로 공개하는 동기를 부여했다.

Netscape가 제품을 자유 소프트웨어로 공개하면서 사람들은 자유 소프트웨어를 산업에 가져오는 방법을 연구하기 시작했다.

1998년, 자유 소프트웨어 운동을 하는 사람들 중 일부가 ‘오픈 소스’ 라는 단어를 만들어냈고, 이후 레이몬드와 사람들은 그 단어를 알리는 일을 시작했다.

이 단어는 기술 출판사 Tim O’Reilly가 1998년에 연 행사에서 큰 호응을 얻고, ‘프리웨어 서미트(Freeware Summit)’로 불리다가 '오픈 소스 서밋(Open Source Summit)'으로 명명되었다.

그 행사에는 레이몬드를 포함한 여러 사람들이 모였는데, ‘자유 소프트웨어’라는 단어로 인해 혼란이 생겼다. ‘소스웨어’, ‘오픈 소스’가 주장되었고 투표를 해 ‘오픈 소스’가 채택되었다.

이후 오픈 소스 소프트웨어가 대중화되면서 Microsoft같이 오픈 소스 방식이 아닌 산업계는 위협을 느끼기도 했다.

오픈 소스 이니셔티브 (OSI, Open Source Initiative)

- 오픈 소스 정의(OSD)의 관리 및 촉진을 담당하는 비영리 조합

오픈소스 이니셔티브는 오픈 소스 소프트웨어 사용을 장려하기 위한 일종의 캠페인으로, 1998년 2월에 Netscape Communications Corporation에 대한 소스 코드를 공개한 것에 대해 영감을 받아 레이몬드를 포함한 여러 사람이 모여 만들었다.

- 하는 일

OSI가 인증하는 오픈 소스 소프트웨어(OSS) 인증 마크를 통해 소프트웨어가 실제로 오픈 소스라는 것을 증명하고, 오픈 소스의 복제도 가능하다.

소프트웨어의 소스 코드를 자유롭게 읽고, 재배포 및 개조를 가능하게 함으로써 소프트웨어가 향상되고, 한 사람이 느린 속도로 소프트웨어를 개발하는 것보다 여러 사람들이 고치고 쓰고 버그를 개선하는 것이 보다 빠를 수 있다는 오픈 소스의 기본 이념을 구현하는 단체이다.

- 자유 소프트웨어와 OSI의 관계

스톨만은 자신의 자유 소프트웨어 운동과 오픈 소스 이니셔티브를 동일한 자유 소프트웨어 공동체 내에서 별도의 단체로 묘사했으며 철학적인 차이에도 불구하고 오픈 소스와 자유 소프트웨어의 지지자는 "둘 다 실제 프로젝트에서 함께 작동한다"고 인정했다.

> 어떤 사람들은 자유 소프트웨어와 거의 같은 부류를 나타내고자 용어 “오픈 소스” 소프트웨어를 사용한다. 오픈 소스 소프트웨어는 자유 소프트웨어와 정확히 같은 종류의 소프트웨어는 아니다. 오픈 소스 소프트웨어는 우리가 너무 제한적이라고 여기는 라이선스를 받아들이기도 하며, 그들이 인정하지 않는 자유 소프트웨어 라이선스가 존재하기도 한다. 하지만 부류의 범위에 대한 차이는 작다. 거의 모든 자유 소프트웨어는 오픈 소스이고, 거의 모든 오픈 소스 소프트웨어는 자유이다. — 자유 소프트웨어 재단, https://www.gnu.org/philosophy/categories.htm

에릭 S 레이몬드(Eric S. Raymond) 1957.12.4

- 미국의 소프트웨어 개발자이자 컴퓨터 관련 책 저자이다.
- 지은 책

*The Cathedral and the Bazaar (성당과 시장)

*The New Hacker’s Dictionary (새로운 해커 사전)

1996년, 레이몬드는 오픈 소스 이메일 소프트웨어 “popclient”를 인수하여 Fetchmail로 이름을 바꾸었다.

이 경험이 있은 후 그는 The Cathedral and the Bazaar (성당과 시장)을 출판했다.

레이몬드는 리누스 토발즈에 의해 영감을 얻은 "리누스의 법칙 (Linus Law)" 이라고 불리는 경구를 썼다.

> Given enough eyeballs, all bugs are shallow. (충분한 눈알만 가지고 있으면 모든 버그는 얕다.)

이것은 레이몬드의 책 'The Cathedral and the Bazaar(성당과 시장)'에서 처음으로 나타났다.

Chapter 02. 오픈소스 역사

Section4 : 현대 : 인물, 재단, 기업에 대해 ...

- 리처드 스톨만
- 리누스 토르발스
- GIT
- GOOGLE
- CHROME OS
- ANDROID
- 레드햇

4. 현대 : 인물, 재단, 기업에 대해 ...

리처드 스톨만

- GNU 프로젝트와 자유소프트웨어 재단(FSF)의 설립자

1953년 맨해튼에서 리처드 스톨만은 태어났다.

학창시절인 1965년부터 1969년까지 컴퓨터를 접하며 프로그래밍을 배웠고,

1971년부터 매사추세츠 공과대학 인공지능 연구소에서 '해커1)'로 일하기 시작했다.

그는 이 시기를 회고하면서 “요리법을 공유하는 것이 요리의 역사만큼 오래된 것처럼, 소프트웨어를 공유하는 것은 컴퓨터의 역사만큼 오래된 것이었다. 우리(해커들)는 MIT에서 소프트웨어를 공유하는 이상적인 공동체를 만들었다"고 말했다.

1985, GNU 선언문을 발표하였다.(유닉스에 대항하여 자유로운 대안을 만들기 위함)

> GNU 선언문 GNU: GNU is Not Unix, 유닉스 계열 컴퓨터 운영 체제

GNU 선언문을 발표한 후에는, GNU 프로젝트를 지원하기 위해 자유 소프트웨어 재단(Free Software Foundation) 설립하였다.

현재 널리 쓰이고 있는 일반 공중 사용 허가서(GPL) 소프트웨어 라이선스의 개념을 도입하였다.

1989, GPL내에 카피레프트의 개념을 적용하였다.

> 카피레프트: copyright에 반대되는 개념으로, 저작권에 기반을 둔 사용제한이 아니라 정보의 공유를 목표하는 것, 지적재산권에 반대해 지적 창작물에 대한 권리를 모든 사람이 공유할 수 있도록 하는 것

1991, 리누스 토르발스의 리눅스 커널 발표로 1992년에 GNU/리눅스 운영 체제가 탄생하였다.

리누스 토르발스

- 리눅스 커널과 Git 개발자

리누스 토발즈는 1969년, 12월 28일 핀란드의 헬싱키에서 태어났다.

어린 시절, 리누스 토발즈의 집안은 넉넉하지 못했고 부모가 이혼을 하는 등, 가정도 다소 불안한 편이었다.

그러던 1981년, 통계학 교수인 외할아버지가 코모도어(Commodore) VIC-20 컴퓨터를 구매하면서 리누스 토발즈의 인생은 바뀌기 시작했다.

호기심 많은 10대 소년에게 컴퓨터는 정말로 무한한 재미를 주는 존재였으며, 곧 깊이 빠져들었다. 그리고 몇 년 후 외할아버지가 세상을 떠나면서 이 컴퓨터는 자연스럽게 그의 것이 되었다.

1988년 헬싱키 대학교에 입학한 후, 이듬해 입대해 11개월 간 병역의무를 수행할 때를 제외하곤 그는 줄곧 컴퓨터에 매달렸다고 한다.

1991, Linux의 프로토타입을 공개

GNU 프로젝트를 접한 후, 스톨만의 'GNU General Public License version 2(GPLv2)'를 사용

1994, 리눅스 Version 1.0을 발표

1996, 컴퓨터 과학 석사 졸업(논문 제목:'Linux: A Portable Operating System')

Git

- 컴퓨터 파일의 변경 사항을 추적하고 여러 사람 사이의 파일에 대한 작업을 조정하는 버전 제어 시스템이다.
- 주로 소프트웨어 개발의 소스 코드 관리 에 사용되지만, 모든 파일 세트의 변경 내용을 추적하는 데 사용할 수 있다.

*분산 개정 관리 시스템으로서 속도, 데이터 무결성 및 분산 된 비선형 워크 플로우에 대한 지원을 목표로 한다.

- 소스 코드 관리를 위한 분산 버전 관리 시스템

  > DVCS : 분산 버전 관리 시스템이란, CVCS(중앙집중식 버전 관리 시스템)과 다르게 각 개발자가 중앙 서버에 접속하지 않은 상태에서도 코드 작업을 할 수 있는 것이 특징이다.

대부분의 클라이언트 - 서버 시스템과 달리, 모든 컴퓨터의 모든 Git 디렉토리 는 네트워크 액세스나 중앙 서버와 관계없이 완전한 기록 추적 및 전체 버전 추적 기능을 갖춘 저장소이다.

리누스 토르발스가 리눅스 커널 개발에 이용하려고 개발하였다고 한다.

- 연혁

2005, 버전 0.99가 첫 출시되었다.

Google

- 구글은 '페이지 랭크'라는 독자적인 검색 알고리즘을 개발해 검색 시장을 장악하고 성장한 세계 최대 인터넷 검색 서비스 회사다.
- 구글은 전 세계 60개국 이상에 지사를 두고, 130개가 넘는 언어로 검색 인터페이스를 제공하고 있다.

- 연혁

1998, 'BackRub'이라는 이름으로 설립

미국 전체 인터넷 검색의 2/3, 전 세계의 70%를 장악

2005, 구글맵과 세계 최초의 위성영상지도 서비스인 구글어스(Google Earth)를 출시

2006, 유튜브(세계 최대의 동영상 공유 사이트) 인수

2007, 더블클릭(디지털 마케팅 회사) 인수

2012, 모토롤라 모빌리티를 인수했고, 구글의 무인자동차가 미국 네바다 주 교통부로부터 면허를 획득,구글 최초의 태블릿 PC ‘넥서스 7’ 출시

Chrome OS

- 구글이 설계한 오픈 소스 운영 체제로 Linux 커널을 기반으로한다.
- Google 크롬 웹 브라우저를 주요 사용자 인터페이스로 사용하므로 Chrome OS는 기본적으로 웹 응용 프로그램을 지원한다.

Chrome OS는 부분적으로 오픈 소스 Chromium OS(크로미엄 OS)프로젝트 하에 개발되었다.

다른 오픈 소스 프로젝트와 마찬가지로 개발자는 Chromium OS의 코드를 수정하고 자체 버전을 만들 수 있지만 Chrome OS 코드는 Google 및 파트너 만 지원하며 목적을 위해 설계된 하드웨어에서만 실행된다.

또한 Chrome OS는 Chromium OS와는 달리 자동으로 최신 버전으로 업데이트된다.

- 연혁

2009.07, Chrome OS 발표

2009.11, Chrome OS의 소스 코드를 Chromium OS 프로젝트로 출시

Android

- 휴대 전화를 비롯한 휴대용 장치를 위한 운영 체제와 미들웨어
- 사용자 인터페이스 그리고 표준 응용 프로그램(웹 브라우저, 이메일 클라이언트, 단문 메시지 서비스(SMS), 멀티미디어 메시지 서비스(MMS)등)을 포함하고 있는 소프트웨어 스택이자 모바일 운영 체제

Java 언어로 응용 프로그램을 작성할 수 있게 함

리눅스 커널 위에서 동작

2005, 구글에서 안드로이드 사를 인수

2008, 안드로이드를 오픈 소스로 선언

안드로이드의 모든 소스 코드를 오픈 소스 라이선스인 아파치 v2 라이선스로 배포하고 있어, 기업이나 사용자는 각자 안드로이드 프로그램을 독자적으로 개발을 해서 탑재할 수 있다.

2013, 현재 API만 완전한 공개소스인 상태이며 VM에대한 소스는 공개하지 않고 있다. 그런점에서 완전한 오픈소스 스마트폰 운영체제라고 할 수는 없다.

레드햇

- 레드햇은 1993년 미국에서 설립된 기업

기업용 오픈소스 기술을 주로 개발하고 있다.

> 여기서 말하는 기업용 오픈소스 기술이란 이미 공개된 오픈소스 기술에 추가 기능이나 유지보수 서비스를 더한 것을 말한다.

과거 기업에서 오픈소스 기술을 이용할 때는 몇 가지 제약이 있었다.

빠른 업데이트 주기로 새로운 기술을 검증하기 힘들고, 안정성과 보안도 취약했다. 또한 직접 만든 기술이 아니니 오픈소스 기술의 전문성을 기르기도 어려웠다.

레드햇은 이러한 문제점을 해결하는 데 주력했다.

예를 들어, 커뮤니티 오픈소스 기술이 6개월 단위로 업데이트 된다면, 레드햇은 과거 버전을 3년간 안정적으로 운영할 수 있도록 기술을 제공했다.

타사 기술과 함께 이용할 수 있도록 호환성을 높이기도 했다. 또한 별도의 교육 프로그램을 운영해 오픈소스 기술의 전문성을 높일 수 있도록 지원하고 있다.

- 연혁

1993, Bob Young이 리눅스와 유닉스 소프트웨어 악세사리를 판매하는 ACC Corporation을 설립

1994, Marc Ewing이 자신의 리눅스 배포판을 만들어 Red hat Linux라고 이름 붙였다.

1995, Young은 Ewing의 사업을 인수했고, 자신의 사업과 통합하여 Red Hat Software를 설립하여 CEO가 되었다.

1999, Cygnus Solutions 인수

> 시그너스 솔루션즈(Cygnus Solutions): 1989에 자유 소프트웨어의 상업적인 지원을 위해 John Gilmore, Michael Tiemann and David Henkel-Wallace가 설립한 회사이다.