Q&A로 알아보는 오픈소스

소프트웨어학부 blue-07 단팥빵 팀

2017011849 김재훈, 2017011821 김유신, 2017012042 백수민,
2017012460 이평원, 2017012615 황인재

목차

현재까지 이용되고 있는 과거의 오픈소스 소프트웨어는 무엇이 있나요?
앞으로 오픈소스 소프트웨어는 어떻게 이용될까요?

연표

- 1950년대

  컴퓨터 초기
  IBM에서 쉐어와 유즈가 제작됨.

- 1960년대

  소프트웨어 단체인 IBM의 SHARE와 DEC의 DECUS 설립
  PDP-1이 MIT에 도입되면서 해커문화가 발생
  MIT에서 다중사용자 지원 시스템 제안
  다중사용자 지원 시스템을 벨 연구소와 제너럴 일렉트로닉스가 연구

- 1970년대

  소프트웨어와 하드웨어가 분리됨.
  소프트웨어의 상용 판매가 급증됨.
  빌 게이츠가 '취미로 소프트웨어를 개발하는 개발자들에게 보내는 공개 서한'을 공개
  데니스 리치와 켄 톰슨이 유닉스 운영체제의 근간 발표
  AT&T가 사업 양보 명령을 받음.
  AT&T에서 유닉스가 무료로 발표됨.
  AT&T에서 유닉스 버전 7을 유료로 발표함.
  BSD가 최초로 배포되었다.
  TeX가 개발됨.

- 1980년대

  컴퓨터 네트워킹과 개인용 워크 스테이션이라는 주제로 해커문화가 발달함.
  BSD가 무료 배포 라이선스로 배포됨.
  유닉스가 유료가 됨에 따라 미닉스가 발표됨.
  GNU 운동이 시작되었으며, Linux가 발표되었다.

소개

Q. 오픈 소스란 무엇인가요?

A. 오픈 소스란 지적재산권 문제에 얽매이지 않고 공동으로 소프트웨어를 개발할 수 있는 협동방식이에요. 그리고 그 방식으로 개발된 소프트웨어는 오픈 소스 소프트웨어(OSS)로, 소스 코드에 대한 접근, 자유로운 재배포, 파생 저작물의 작성, 제한 없는 사용 등을 허용하는 라이선스와 함께 배포되는 소프트웨어이에요.

Q. 오픈 소스에 기미된 철학은 무엇인가요?

A. 오픈 소스의 시초라고 불리우는 리차드 스톨만은 GNU 프로젝트를 진행하면서 핵심을 '자유 소프트웨어'라는 것에 두었어요. 자유 소프트웨어의 자유(Free)는 가격으로서의 자유'무료'가 아닌 말그대로의 '자유'를 의미해요. 소프트웨어는 다른 어떠한 물체들과는 달리 훨씬더 쉽게 카피되고 변화될 수 있어요. 이러한 요소가 소프트웨어의 발전에 어떠한 방식이 유용한지를 보여주죠. 구체적으로 다른 사람들의 코드를 원하는대로 바꿀 수 있고, 사람들이 공동체를 구성할 수 있게 해주는 자유, 즉, 소프트웨어의 한 부분을 가져다 쓸 수 있고, 짧은 시간동안 변경하는 것만으로 필요하는 것을 만들어 낼 수 있으며, 많은 사람들이 그것을 사용할 수 있다는 것이 리차드 스톨만은 소프트웨어의 발전에 큰 기여를 할 것 이라고 주장했어요. 그리고 이것을 자유 소프트웨어 운동의 철학으로 세웠죠. 하지만 많은 사람들이 자유(Free)의 가격으로서의 자유로 해석해서 무조건 무료로 코드를 공개해야한다는 오해 때문에 자유 소프트운동의 초기에는 어려움을 겪었어요.

Q. 오픈 소스 프로젝트는 어떻게 진행되나요?

A. 과정을 나열해 보자면 먼저 프로젝트를 선정하고 프로젝트의 목표를 정의해요. 그런다음 프로젝트 명칭을 결정하고 초기 설계와 소스 코드를 작성하여 프로젝트 참여자들을 유인하는데 사용해요. 그리고 소프트웨어의 바탕이 될 라이선스를 결정하고 버전 번호부여 방법을 정해요. 소프트웨어가 완성되면 그것을 패키지 형식으로 전환하고 버전 관리 시스템도 선정해요. 마지막으로 문서를 작성하고 배포 수단을 정해요. 이때 문서란 개발자를 위한 참고 메뉴얼과 소프트웨어의 설계도 등을 말해요. 나중에 버그가 발견되었을 때 개발자들이 코드의 구조를 빠르게 다시 기억하는데 중요한 역할을 하죠. 그리고 완성된 프로젝트는 전자 우편이나 웹사이트를 통해 공표하여 배포해요. 여기서 웹사이트는 주로 GitHub 나 Pastebin 와 같은 모든 사람이 이용가능한 공유 사이트를 말하죠. 여기까지가 프로젝트를 완성하고 배포하기 까지의 과정인데요. 배포를 하면 사용자들은 그것을 보완 및 수정을 하고 활용하여 또 새로운 프로젝트를 진행할 수 있어요. 즉, 위의 과정이 반복되는거죠. 반복되면서 오픈 소스 소프트웨어들은 발전되고 기존의 것들을 이용한 새로운 소프트웨어들도 무수히 개발될 수 있어 오픈 소스의 전망은 무궁무진 하다고 말하고 싶어요.

Q. 오픈 소스 소프트웨어 기업은 어떻게 수입을 창출 하고 있는 거죠?

A. 아마도 여러분이 오픈 소스에 관심을 갖게 되면서 이 질문이 가장 궁금해 하는 점들 중 하나라고 생각해요. 소스 코드를 공개하고 저작권없이 오픈 소스를 전문하는 기업들은 수익을 어떻게 창출하고 유지될 수 있는 걸까요? 우선 오픈 소스 소프트웨어를 개발하는 회사들은 꼭 소스 코드를 무료로 배포할 필요가 없어요. 일부 라이선스에 따라 소스 코드의 일부분만 공개하고 나머지 부분도 다운받기 위해선 일정한 비용을 대가로 받을 수 있어 그걸로 수익을 창출 할 수 있죠. 그리고 오픈 소스 소프트웨어를 유통해주고 관리해주는 서비스를 제공하는 회사들은 서비스를 지원해주는 대가의 비용으로 수익을 창출 할 수 있어요.

Q. 오픈 소스와 클로즈드 소스는 어떤 차이가 있나요?

A. 말 그대로의 의미를 해석해보면 구분하기 쉬워요. 오픈 소스는 'Open Source'로 소스 코드를 공개하여 공개된 코드를 누구나 쉽게 공급받을 수 있고 수정 및 활용이 자유로운 방식이에요. 반면 클로즈드 소스는 'Closed Source'로 소스 코드가 저작권 소유자에게만 공개되어있고 코드를 보기 위해선 소유자의 예외적 법적 권한 하에 허가되어야 하는 폐쇄적인 방식이죠.

Q. 오픈 소스의 저작권은 어떻게 설정되어있나요?

공통적 준수사항	선택적 준수사항
- 저작권 관련 문구 삽입, 수정 및 삭제 불가	- 사용 여부 명시
- 제품명 중복 불가	- 소스 코드의 공개할 범위 표기
- 서로 호환되는 라이선스들만 조합가능	- 특허 관련 문구 기입

Figure 1: 공통적 준수사항&선택적 준수사항

A. 오픈 소스 소프트웨어는 겉보기엔 저작권이 소멸된 저작물로 보일 수 있지만 절대 아니에요. 만약 저작권이 없었다면 누군가가 그것을 약간 수정한 뒤에 독점할게 뻔하죠.

이를 막기 위해 '카피레프트'라는 개념이 생겼어요. 이 소프트웨어의 저작권이 보호받고 저작자들은 복사본의 재배포를 허용하고, 수정하거나 무언가를 추가할 수 있는 권한을 주지만, 라이선스의 조항을 따라야 한다는 것 말이죠. 여기서 따라야 하는 라이선스의 조항에 대해 얘기해 보자면, 공통적 준수사항과 선택적 준수사항으로 나뉘어요.

먼저 공통적 준수 사항은 모든 사람이 필수적으로 지켜야할 사항인데요. 첫째로 개발자 정보와 연락처 등 저작권의 보호를 받기 위한 관련 문구를 유지 하고 이를 수정하거나 삭제해서는 안돼요. 그리고 둘째로는 제품명이 중복되어서는 안된다는 것인데요. 이는 오픈 소스 소프트웨어들이 주로 그것의 이름들이 상표가 되는 경우가 많기 때문이애요. 마지막으로꼭 서로 다른 라이선스가 조합되어 있어야 한다는 것이에요. 만약 서로 호환이 안되는 라이선스들이 조합되면 한 라이선스에선 가능한데 다른 라이선스에선 불가능한 경우가 발생되기 때문에 모든 측면에서 호환이 가능한 라이선스들을 조합해야해요.

선택적 준수사항은 라이선스에 따라 앞으로 말할 세가지 모두에 관련된 조항이 있을 수 있고 일부만을 요구하는 조항이 있을 수 있어요. 첫 번째로 사용 여부 명시로, 어떤 소프트웨어를 사용할 때 해당 오픈 소스 소프트웨어가 사용되었음을 표기하는 것이에요. 두 번째로 소스 코드의 공개인데요. 사실 이건 오픈 소스로써 반드시 이루어져야 할 것이지만, 여기선 소스 코드의 정확한 공개 범위를 조항에 표기하여 정할 수 있어요. 그리고 꼭 그 범위에 해당하는 코드는 공개해야 하는 거죠. 마지막 세 번때는 특허에 대한 기본적인 내용을 포함해야 한다는 것이에요. 만약 어떤 기술이 특허로 보호될 경우 해당 기술을 구현 할 때 반드시 특허권자의 허락을 받아야 하는데 이것은 사실 오픈 소스 소프트웨어 뿐만아니라 모든 소프트웨어에 공통적으로 해당돼요. 최근엔 SW특허가 급격히 증가하면서 이와 관련된 문제가 심각해지고 있어서 요즘 만들어지는 오픈 소스 소프트웨어 라이선스들에는 특허관련 조항을 포함하는 경우가 많아지고 있어요.

Q. 오픈 소스의 장단점은 무엇인가요?

장점	단점
- 공급자들에 대해 독립적	- OSS를 아직 찾기 어려움
- 개발 비용이 적게듦	- 부서에 필요한 OSS의 존재 여부를 알기 어려움
- 취약점의 빠른 피드백 가능	- 공급자의 문서화가 빈약하면 사용자가 코드를 이해하기 어려움
- 한 소프트웨어으로의 고착성을 피할 수 있음	- 현실적인 경험과 지원이 부족함

Figure 2: 오픈 소스의 장단점

A. 먼저 오픈 소스의 장점은 첫째로 공급자들에 대해 독립적이라는 것이에요. 오픈 소스의 방식으로 개발된 소프트웨어는 이미 사용자들에게 소스 코드가 공개되어 있기 때문에 만약 공급자가 사라지거나 지원을 중단한다해도 소스 코드는 유지될 수 있어요. 그리고 두번째 장점은 개발 비용이 적게든다는 것인데요. 일반 상용 소프트웨어(상업적인 목적이나 판매를 목적으로 생산되는 소프트웨어)는 상업적 이용과 배포를 엄격하게 제한하는 라이선스 때문에 상업적 활용을 허가받기 위해서는 대가를 지불해야는 반면, 오픈 소스 소프트웨어는 소프트웨어의 사용, 수정, 복제, 배포의 자유를 허가하는 GPL, BSD, MIT 등과 같은 공공 라이선스를 바탕으로 개발되기때문에 라이선스 비용이 적게들어요. 그리고 사용자들 모두가 개발에 참여하기 때문에 따로 사람들을 교육시킬 필요가 거의 없어 교육비용이 적게 들고 오픈 소스 소프트웨어의 업그레이드 및 유지 보수를 하는 데의 비용도 적게 들죠. 세번째 장점은 오픈 소스 소프트웨어에서 발견된 보안의 취약성에 빠른 피드백이 가능하다는 것이에요. 정말 많은 사람들이 개발에 참여하고 있기 때문에 취약성이 발견되면 때로는 하루 또는 단 몇 시간 만에 문제를 언급할 수 있고 해결할 수 있어요. 반면 상용 소프트웨어의 사용자들은 개발자들이 새로운 버전을 출시하는 데에만 의존하고 있어 위의 과정이 진행되는데 정말 오랜 시간을 필요로 하고, 그 동안 사용자들은 노출된 위협에 취약할 수 있죠. 마지막으로 언급할 장점은 바로 한 소프트웨어에만 고착화 될 수 있는 문제를 피할 수 있다는 것인데요. 구매자들은 종종 소프트웨어 제품이 같은 제조업자에 의해 만들어진 자매품과 가장 잘 작동할 것이라고 생각해요. 하지만 오픈 소스 소프트웨어는 다른 관련 제품과는 독립적으로 쓰이는 경향이 있기 때문에 구매자들에게 다른 제품을 구입하려는 더 큰 자유를 제공할 수 있어요.

다음은 오픈 소스의 단점인데요. 먼저 첫째로 오픈 소스 소프트웨어를 아직은 찾기 힘들다는

것이이요. 정확하게 오픈 소스 소프트웨어를 구성하고 있는 것이 무엇인지, 오픈 소스 소프트웨어의 상대적인 장점과 약점이 무엇인지가 불분명하여 특정 제품에 있어서, 지원받는 것이 제한되거나 획득하기 힘들다는 염려가 있어요. 하지만 현제 IBM, Sun, HP와 같은 큰 공급자들이 GNU/Linux 운영체제를 지원하는데 상당한 노력을 하고 있고 기존 상용 소프트웨어의 공급자들 또한 이 운영체제를 사용하기 위해서 노력하고 있으니 앞으로 이 문제는 점차 해결될 것이라고 기대하고 있어요. 두번째 단점은 비즈니스 문제에 있어서 오픈 소스 소프트웨어는 전적으로 상용 소프트웨어처럼 동일한 소프트웨어를 광고하지 않기 때문에 부서가 자신들의 요구를 충족시켜줄 기존 오픈 소스 소프트웨어 제품이 있는지, 혹은 아주 작은 부분의 수정을 거친 후에 그들의 요구를 충족할 수 있는 제품이 있는지에 대해서 알 수 없는 경우가 있어요. 하지만 이 문제는 각 부서들이 자신들의 경험들을 공유할 수 있는 그룹(OGC)에 가입하게 됨으로써 점차 해결되고 있어요. 그리고 세번재 단점은 오픈 소스 소프트웨어의 공급자의 빈약한 문서화로 인해서 배포과정에 문제가 발생하는 것이에요. 시스템을 구입한 오픈 소스 소프트웨어의 부서는 계약상의 일부로써 기술된 문서에 적절하게 응답해야 하기 때문에 기존의 문서가 빈약해서는 안돼요. 이를 위해선 공급자들이 기술된 문서들을 정확하게 전달하는 것이 중요해요. 마지막 단점은 상용 소프트웨어에서 오픈 소스 소프트웨어로 전환하기 위한 현실적인 경험과 지원이 부족하다는 것이에요. 대부분의 하드웨어 공급업자들은 때에 맞추어 오픈 소스 공동체에게 새로운 운영요소를 발표하지 않아서 오픈 소스 소프트웨어는 하드웨어 지원이라는 측면에선 상용 소프트웨어에 뒤처질 수 밖에 없어요. 하지만 최근 하드웨어 공급업자 스스로가 그들의 기반에 오픈 소스 소프트웨어를 사용하고 시장화 하기 시작하면서 이 문제는 감소하고 있어요.

Q. 오픈 소스를 추천하는 이유는 무엇인가요?

A. 오픈 소스의 방식은 전 세계의 수많은 개발자가 디버깅에 참여하고 개발하기 때문에 외부 환경변화 및 기술발전 속도가 매우 빨라요. 그리고 폐쇄적인 개발과 디버깅을 수행함으로써 개발된 프로그램들은 발전 속도가 빠르지 않아 그동안 그들에 대한 안정성과 신뢰성의 문제가 발생할 수 있지만 오픈 소스의 방식은 이를 보장할 수 있죠. 무엇보다 위에서 말했듯이 비용이 적게 들어 경제적 부담이 크지 않아요. 또 오픈 소스에 흥미는 있지만 아직 실력이 부족한 사람들에겐 누구나 코드를 볼 수 있기 때문에 그걸 보고 연구하고 공부해서 배울 수 있는 좋은 기회가 될 수 있어요.

개괄

Q. 오픈 소스는 어떻게 도입되었나요?

A. 처음에 모든 소프트웨어는 오픈 소스였어요. 주로 프로그래머들이 시스템을 더 발전시키려는 목적으로 기술, 응용 프로그램 등을 개발하고 이를 공유했죠. 초기에는 자원이 부족했기 때문에 사람들은 소프트웨어를 공유하는 것이 당연하다고 여겼어요. 그리고 이들은 스스로 조직화하기 위해서 커뮤니티를 만들었는데 대표적으로 IBM의 쉐어(1967)와 DEC의 DECUS(1961)가 있어요. 시간이 흘러 1970년대 사유 소프트웨어가 등장하기 시작했는데, 그 이유에는 크게 2가지 사건이 있어요. 우선 1970년대까지는 소프트웨어는 하드웨어 제조업자들에 의해 무료로 배포되고 있었어요. 초기에는 소프트웨어의 개발비용이 그렇게 비싸지 않았기 때문에 하드웨어에 소프트웨어를 끼워서 팔 수 있었기 때문이에요. 하지만 소프트웨어의 개발비용이 상승함에 따라서 소프트웨어를 독립적으로 판매하려는 기업이 많아지고, 1969년 미국 법무부에서 하드웨어에 소프트웨어를 끼워팔던 IBM을 상대로 반독점 소송을 제기했어요. 두 번째로는 마이크로소프트의 빌 게이츠가 개발자 커뮤니티에서 한 통의 편지를 공개했어요. 이 편지에서 오픈 소스를 비난하며 독점 소프트웨어라는 새로운 개념을 제시했어요. 이런 사건을 계기로 상용 소프트웨어는 급격하게 늘어났어요. 그 후 AT&T라는 회사의 두 직원은 C언어를 개발하고, UNIX라는 운영체제를 만들었어요 이는 기존의 운영체제에 비해 좋은 점이 많았고, 무상이었기 때문에 많은 사람이 이용했죠. 하지만 시간이 흘러 AT&T에서 UNIX를 상용화했고, 대학이나 연구소에서도 사용할 수 없게 됐어요. 그 외에도 1980년대 상업화하기 시작한 UNIX는 점차 폐쇄적으로 변해갔죠. 그래서 같은 뿌리를 가진 UNIX라도 호환되지 않기 시작했어요. 또한, 기업들은 점차 오픈 소스에 대해 반감을 품게 됐죠. 그러던 중 MIT 연구소의 리처드 스톨만이 폐쇄적인 소프트웨어에 대해 불만을 품기 시작하면서 오픈 소스 운동을 시작했죠.

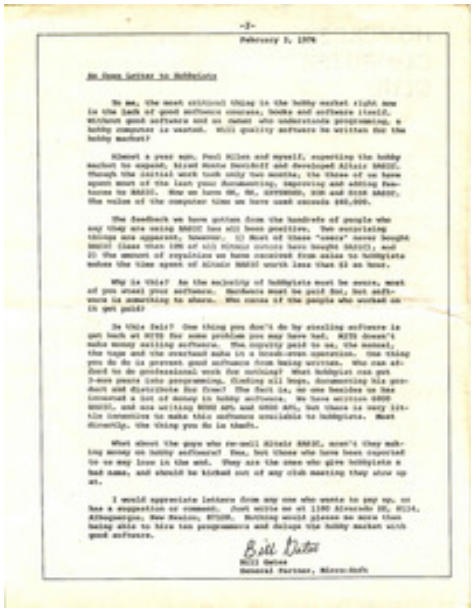
초기

Q. 해커 문화와 오픈 소스는 어떤 관계인가요?

A. 초기의 해커 문화는 MIT에서 시작했어요. 1961년 DEC에서 PDP-1을 MIT대학에 기부하면서 처음 컴퓨터를 접한 MIT의 취미 동아리였던 테크 모델 철도 클럽(TMRC)의 학생들은 그걸 새로운 장난감으로 인식하고, 오늘날 우리가 인식하는 프로그래밍 도구와 관련된 은어, 그리고 주변 환경을 만들어 냈어요. 그리고 '해커'라는 단어가 이들의 해커 문화에서 부터 시작했죠. 그리고 이들은 MIT 인공 지능연구소 해커 문화의 중심이었어요. 다른 연구소와 함께 멀틱스라는 운영체제를 개발하는 중 개발 방향이 맞지 않아 1960년대 후반부터 자신들이 새롭게 ITS(Incompatible Time Sharing System)라는 운영 체제를 개발했어요. 당시 ITS 운영체제에서는 기존과는 다른 사용자 환경을 제공했어요. 그것은 누구나 시스템에 암호 없이 로그인하고, 문서나 소스코드 같은 모든 파일을 누구나 편집할 수 있게 만들었던 것에요. 또한 이것은 MIT 내부뿐만이 아니라 다른 기관에서도 아파넷을 통해 ITS에 접속할 수 있었어요. 이러한 ITS의 개방적인 부분은 해커 문화에 큰 영향을

줬고, 오픈 소스에도 영향을 줬어요. 대표적으로 당시 MIT 소속이던 리처드 스톨만은 지금 오픈 소스의 선두주자로 있어요.

Q. 마이크로소프트는 오픈 소스에 대해 어떻게 생각했나요?



-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 useless. 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" have 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.

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 us, 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

Figure 3: 취미로 소프트웨어를 개발하는 개발자들에게 보내는 공개 서한(1976)

Figure 4: 마이크로소프트 헬러윈 문서(1998)

A. 마이크로소프트는 오픈 소스에 대해 무척 부정적으로 생각했어요. 왜냐하면 마이크로소프트는 독점 소프트웨어를 기반으로 만들어진 회사였기 때문이에요. 그리고 이전에는 마이크로소프트가 컴퓨터 시장을 독점하고 있었는데, 어느 날 오픈 소스로 만들어진 리눅스의 등장으로 밥그릇을 나눠 먹어야 하기 때문이에요. 그래서 마이크로소프트를 창업한 지 얼마 안 되었던 빌 게이츠는 '홈브루 컴퓨터 클럽'에서 '취미로 소프트웨어를 개발하는 개발자들에게 보내는 공개 서한'(An Open Letter to Hobbyists)이란 편지를 썼어요. 이 편지는 커뮤니티를 통해 일어나는 소프트웨어의 만연한 저작권 침해에 대해 좌절감을 표현하는 편지였어요. 글의 내용을 보면 "당신들 대부분은 소프트웨어를 훔치고 있다.", "어떤 취미가들이 3명의 프로그래머가 1년 동안 작업하고, 디버깅하고, 문서화한 제품을 무료로 배포할 수 있겠는가?" 등 오픈 소스에 대한 원색적인 비난을 했어요. 또한, 마이크로소프트의 CEO였던 스티브 발머는 리눅스는 지적 재산권에 있어서 암같은 존재라고 언급했어요. 이는 오픈 소스를 이용하면 이후 개발하는 소프트웨어도 모두 공개소스로 내놓아야 하는 게 업체들의 지적 재산권을 침해한다는 주장이었어요. 마지막으로 1998년 11월 1일 에릭 S. 레이먼드에 의해 공개된 핼러윈 문서에는 리눅스나 오픈소스 운동이 마이크로소프트에 독점적인 시장에 위협을 가하고 있다는 점을 인정하면서 견제하고 있다는 게 서술되어 있어요.

Q. AT&T에서 만든 UNIX는 무엇인가요?

A. 캔 톰슨이란 사람 AT&T 벨 연구소의 직원이었어요. 1960년대 중반 벨 연구소와 제너럴 일렉트로닉, GE, MIT에서는 ITS와 같은 시분할 운영체제인 멀틱스(Multics)의 개발을 진행하고 있었어요. 하지만 벨 연구소에서는 멀틱스의 거대한 크기와 복잡성에 좌절해 점차 프로젝트에서 손을 떼 버렸어요. 하지만 톰슨은 멀틱스의 환경을 그리워해서 쓰레기가 되어버린 PDP-7에 자기 생각을 구현해보기로 했죠. 멀틱스의 말장난을 이용해 UNICS라는 이름으로 1970년 프로젝트를 제안했고, 브라이언 커니핸이란 사람에 의해 UNIX라고 철자가 바뀌게 됐어요. UNIX 개발 당시에 PDP-11으로 컴퓨터가 업그레이드되었기 때문에 어셈블리로 작성했던 UNIX의 코드를 다시 작성해야 했어요. 그래서 데니스 리치와 톰슨은 함께 개발한 B언어로 UNIX를 다시 개발하려고 했었죠. 그러던 중 B언어를 수정하여 C언어로 재탄생 시키고 이로 1972년 UNIX를 다시 개발했어요. 이것은 기계 종속적이지 않기 때문에 어느 장치에나 사용할 수 있었고, 이 도구에 대한 관심은 벨 연구소 내에 퍼졌어요. 그렇게 탄생한 UNIX의 공식적인 법인체인 AT&T에게 전화 사업 이외의 사업에 대해 양보 명령이 내려졌고, UNIX는 어떠한 지원 없이 무상으로 제공됐어요. 그리하여 수많은 대학과 연구소에서 사랑을 받게 됐죠. UNIX는 AT&T의 공식적인 지원이 없었기 때문에 학생이나 연구자 같은 사용자들은 발생한 문제를 직접 해결해야 했어요. 이 과정에서 서로 간의 아이디어, 정보, 프로그램 및 버그 수정본을 공유하였죠. 또한, 사용자들이 더 편하게 개발하도록 커니핸과 플라우거는 몇몇 핵심 유틸리티의 구현을 쉽게 이해할 수 있는 책을 소스코드와 함께 출판하였고, 사용자들은 각 유틸리티를 강화하고 다양한 아키텍쳐에 이식하기 위해 공동으로 협력하였어요. 1978년에 그들은 공식적으로 STUG라는 사용자 그룹을 구성하였어요. 사용자 그룹도 형성되고, 책도 발간되는 등의 발전을 이루지만, AT&T가 UNIX를 상용화함에 따라 1979년에 시스템 소스 코드에 접근 제한을 두는 라이선스를 가진 버전 7을 배포하였어요. 이로 인해 UNIX는 대학 교육과정에서 사용할 수 없었고, 오픈 소스 소프트웨어 운동에 자극을 주게 되었죠.

중반

Q. GNU 프로젝트란 무엇인가요?

Figure 5: GNU Logo

A. 마이크로 소프트의 공개서한과 IBM를 상대로 한 미국 법무부의 반독점 소송에 의해 사유소프트웨어가 급증하기 시작했어요. 한편, 70년대 후반 리처드 스톨만이 일하고 있던 MIT연구소에서, 개발자들끼리 코드를 공유 못하게 되어 연구소 전체 효율이 떨어졌어요. 또한, 이 일를 통해 MIT 주변의 소스 공유 커뮤니티가 붕괴되었어요. 이에 분개하며 리처드 스톨만은 사유 소프트웨어에 대한 적의를 키워나갔어요. 그는 자신이 커뮤니티를 다시 만들자는 신념아래에 나중에 저작권 문제를 깨끗하게 정리하고자 1984년에 MIT에서 퇴사하였어요. 그리고 그는 GNU(GNU is Not Unix) 라는 유닉스를 대체할 유닉스 호환 운영체제를 만드는 프로젝트를 시작하게 되었어요. 운영체제를 만드는 과정에서 자유 소프트웨어의 기술, 철학과 법적 토대를 만들게 되어요.
1985년에 GNU 선언문(GNU Manifesto)를 통해 유닉스와 호환되는 자유 운영체제인 GNU를 만드는 이유와 철학, 예상되는 질문의 대답을 일반에 알렸으며, 해커들에게 참여를 유도했어요. GNU의 철학은 오픈소스 다큐멘터리 Revolution OS에서 밝힌 것에 의하면 GNU 프로젝트에서 리처드 스톨만은 자유 소프트웨어를 말할 때 “내가 자유 소프트웨어를 말할 때, 나는 가격이 아닌 자유를 언급하는 것이다. 공짜 맥주가 아닌 말할 자유를 생각하

라"와 같이 설명하였고, 자세히는 밑의 4가지를 가진 소프트웨어를 자유 소프트웨어라고 규정하였어요.

- 목적에 상관없이 프로그램을 실행시킬 수 있는 자유
- 필요에 따라서 프로그램을 개작할 수 있는 자유(이러한 자유가 실제로 보장되기 위해서는 소스 코드를 이용할 수 있어야만 한다. 왜냐하면 소스 코드 없이 프로그램을 개작한다는 것은 매우 어려운 일이기 때문이다.)
- 무료 또는 유료로 프로그램을 재배포할 수 있는 자유
- 개작된 프로그램의 이익을 공동체 전체가 얻을 수 있도록 이를 배포할 수 있는 자유

이것이 GNU의 철학이 되었고, 이 철학 아래에서 그는 운영체제를 만들면서 1985년에 텍스트 에디터인 이맥스(Emacs)와 통합 컴파일러인 GCC(GNU Compiler Collection)를 시작으로 대체 소프트웨어를 개발하였어요. 이맥스를 리처드 스톨만이 배포할 때 자유 소프트웨어의 특징이 나타났어요. 바로 상업적 사용이 허용되어있는 것이에요. 처음에 그는 이맥스를 MIT 컴퓨터의 익명 FTP 사이트에 올려둔 상태였어요. 하지만 그 당시 이맥스에 관심있는 사람들 중에서 인터넷을 통해 FTP를 이용할 수 있는 사람은 거의 없었어요. 따라서 리처드 스톨만은 어떻게 그들에게 배포할지 고민중이던 상황에서 150달러의 비용을 지불하면 누구에게나 이맥스가 들어있는 테이프를 우편으로 보내는 방법을 생각해냈어요. 이런 방식으로 자유 소프트웨어를 이용한 사업을 시작하게 되었고, 리눅스에 기반한 GNU 시스템 전체를 담고 있는 배포판을 판매하고 있는 현재의 리눅스 배포판 업체들의 시초가 되었어요.
같은 해인 1985년에 리처드 스톨만은 자유 소프트웨어를 규정함에 있어서 자유의 범위에 대해 고민하였어요. 자유의 범위를 너무나도 넓히면 공공 도메인(public domain)으로 공개된 소프트웨어를 손쉽게 독점 소프트웨어로 변형시킬 수 있었어요. 돕 홉킨스라는 사람이 보낸 편지에서 나오는 "카피레프트"라는 이름을 사용하여 카피레프트라는 방식의 조항을 자유 소프트웨어에 적용하기로 하였어요. 이는 카피레프트 라이선스로 배포된 소프트웨어를 수정해서 배포할 때에도 똑같은 라이선스가 적용된다는 것이에요. 대부분의 GNU 소프트웨어에는 카피레프트를 실제로 구현한 라이선스 기준인 GPL(GNU General Public Lisence)이 적용되어 배포되었어요.
GNU을 지원하기 위해 '소프트웨어를 공유하고 변경할 자유'를 촉진하기 위해 자유 소프트웨어 재단(Free Software Foundation, FSF)를 설립했어요. 자유 소프트웨어 재단은 리처드 스톨만이 관리하던 자유 소프트웨어들을 대신 관리하였어요. 즉, 이맥스의 테이프 배포 사업을 맡게 되었고 점차적으로 GNU와 그 이외의 다른 종류의 자유 소프트웨어들, 매뉴얼의 판매도 관리하였어요. 자유 소프트웨어 재단은 기부와 판매, 부수적 서비스를 통해 자금을 충당하고 있었으며, 모은 자금을 이용해서 GNU 소프트웨어 패키지를 만들거나 관리할 사람들을 고용했어요. 그들을 통해 GNU 프로젝트에 박차를 가했죠.
1990년 무렵에는 GNU 시스템이 거의 완성되었지만 운영체제를 구성하는 핵심 부분 중의 하나인 커널이 누락된 상태였어요. GNU 프로젝트내에서 Mach를 기반으로 한 서버 프로세스들을 연결해서 커널을 구현하기로 했어요. 만들려고 했던 GNU HURD(허드)는 Mach와 자율적인 프로세스 서버들의 집합이라고 할 수 있으며 유닉스 커널이 갖고 있는 다양한 기능들을 그대로 수행할 수 있는 것이었어요. 그러나 GNU HURD의 개발은 Mach를 자유 소프트웨어로 배포한다고 예고했던 시기까지 지연되었어요. 또한, Mach를 선택한

것은 커널 프로그램의 디버깅을 피하기 위함이었지만 Mach를 사용하기 위한 준비에 엄청난 애를 먹게 되었고 GNU HURD를 만드는 작업에 많은 시간이 소요되었어요.
GNU HURD가 개발된 이후에도 GNU HURD는 상당히 부족한 점이 많았어요. 그러나 1991년에 리누스 토발즈가 만든 Linux가 배포되었고, 그가 GNU 프로젝트에 Linux를 포함하자고 하므로써 GNU 시스템은 결실을 맺어요. 그리고 이 시스템을 GNU/Linux 라고 부르게 되어요.
이렇게 긴 과정을 통해 여러 사유 소프트웨어 기업에 의해 사라질 뻔했던 자유 소프트웨어 문화가 사라지지 않고 여태까지 남아있어요.

Q. Linux는 무엇인가요?

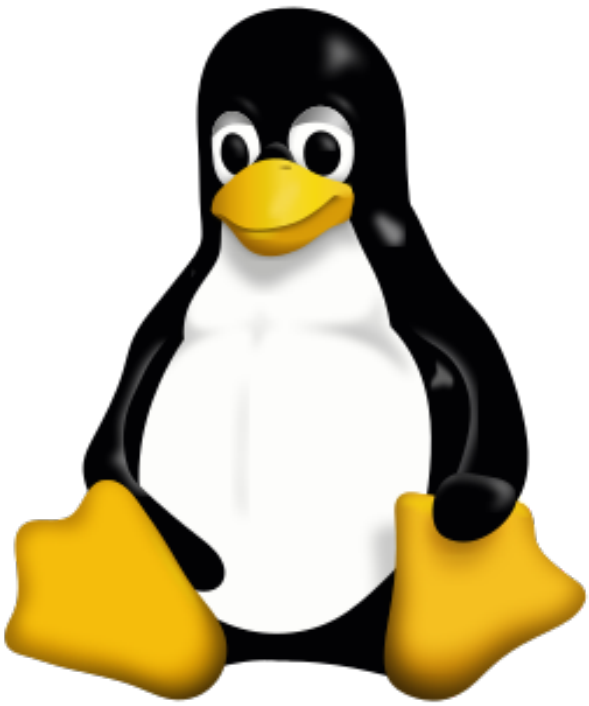


Figure 6: Linux Logo

A. AT&T가 유닉스를 유료화한 이후, 1987년 암스테르담 브리제학교의 앤드류 타넨바움이 자유 소프트웨어 라이선스를 가진 미닉스를 개발했어요. 미닉스는 BSD라이선스를

기반으로 배포되었고, 교육 목적으로 사용할 수 있는 마이크로커널 아키텍처 기반이었어요. 또한, 소스 코드를 학습과 연구를 위해 대학에서 사용할 수 있었어요.
이렇게 유료체제로 바뀐 유닉스에 반발하여 나온 미닉스를 기반으로 연구를 진행하고 있었어요. 하지만 1990년대 초에 대학원생이었던 리누스 토발즈가 미닉스를 사용하다가 한계를 느꼈어요. 또한, 그는 상용 운영체제였던 Sun OS를 사용하면서 "대학 컴퓨터에서 썼던 Sun OS와 비슷한 것을 제 PC에서 실행시킬 수만 있었으면 좋겠다 싶었는데, 저에게 딱 맞는 것을 찾을 수 없어서 직접 만들어 보기로 했습니다."라 말했어요.
그가 결국 개발한 리눅스라는 운영체제는 자유로 공개되었고, 각 연구기관, 대학, 개인용 컴퓨터에서 자유롭게 사용되었어요. 또, 그는 소스를 공개하고 수정, 재배포가 가능하게 하는 GPL 라이선스를 적용하여 리눅스 운영체제를 당시 운영체제를 만드는데에 골머리를 썩고 있던 GNU 운동에 기여하였어요. 이 덕분에 리눅스는 GNU 프로젝트에서 만들어진 소프트웨어를 모두 이식할 수 있었으며, 오픈된 소스를 사용하는 사용자들은 대부분 해커들이었기 때문에 리눅스는 눈 깜짝할 새에 발전되었어요.
향후 리눅스는 웹서버 프로그램인 아파치(Apache)와의 호환이 되도록 제작되어 엄청난 파급력과 함께 배포판이 개발되었어요. 그 결과 현재에는 페도라, 우분투, 안드로이드 등 현재 200개가 넘는 배포판이 존재해요.

Q. BSD란 무엇인가요?

Figure 7: FreeBSD Logo

A. BSD란 Berkeley Software Distribution의 약자로, 미국 캘리포니아 주립대학교 버클리 분교에서 배포한 소프트웨어예요.
유닉스를 만든 벨 연구소의 켄 톰슨은 1966년 버클리에서 전기공학 학위를 취득하였어요. 버클리는 켄 톰슨이 다니던 당시에 자유언론운동의 발상지로 유명했어요. 이 운동은 버클리에서 시작된 학생운동으로 학생들에게 자유로운 이야기할 권리와 학술적인 자유를 주장하였죠. 이 운동은 미국 전역으로 확산될 정도로 규모가 컸었고 이 분위기에서 다닌 켄 톰슨은 벨 연구소에서 동료들과 개발한 유닉스에 대해 거리낌없이 많은 사람들과 나누려는 사상적 철학이 있었어요. 1975년에 그는 안식년 휴가를 받아서 버클리 대학으로 돌아왔어요. 그 시기 버클리 대학은 과거와 달리 정치적으로 무관한 곳이 되어있었지만 컴퓨터 과학에 관한 연구를 하기에는 좋은 환경이었어요. 켄 톰슨은 소규모 교실에서 유닉스의 소스를 강독하는 등 공유 정신과 함께 연구 발표하였어요. 유닉스 소스 강독 때 감명받은 밥 파브리가 CSRG(Computer Systems Research Group, 컴퓨터시스템연구그룹)를 설립하고 이 그룹에 있던 21세에 대학원생이던 비범한 천재 빌 조이와 몇몇 대학원생, 벨 연구소의 동료들과 함께 유닉스를 개선하였어요. 이렇게 탄생한 것이 버클리판 유닉스(Berkeley Unix) 또는 버클리 소프트웨어 배포판(Berkeley Software Distribution)으로

불리는 BSD 유닉스였어요.
이는 벨 연구소의 오리지널 유닉스보다 좋은 성능이었어요. 또한, 이후 유닉스가 매우 비싸게 유료화되었고 소스를 공개안하게 되면서 각종 연구기관의 기준 운영체제가 BSD 유닉스가 되었어요. 이는 훗날 인터넷의 모태가 되는 미국 국방부 고등연구계획국(DARPA, Defense Advanced Research Projects Agency)의 아르파넷(ARPANET) 프로젝트의 기본 컴퓨터 환경으로도 선택되었어요.
BSD는 1970년대엔 주로 빌 조이에 의해 개발되었어요. BSD 유닉스의 개발이 순탄한 것은 아니었어요. 일부 소스가 AT&T의 소스였기 때문에 AT&T와의 마찰이 있었어요. 마찰이 날때마다 BSD 프로젝트의 지원자였던 밥 파브리는 AT&T에 가서 BSD유닉스가 행정적으로 아무런 곤란이 처하지 않도록 해결하였어요. 또, 미국 국방부 고등연국계획국(DARPA)로부터 지속적이고 엄청난 양의 지원금을 타게 만든 연구 지원 요청서를 작성했어요. 그리하여 행정적 위험과 금전적 위험을 모두 생각하지 않으면서 연구자들은 연구 및 개발을 할 수 있었어요. 이후 BSD 유닉스 개발에서 주도적으로 이끌던 빌 조이가 1982년에 썬 마이크로 시스템즈(Sun Microsystems)에 공동창업자로 IT산업계로 뛰어들었어요. 빌 조이가 떠나면서 BSD 유닉스는 소수의 핵심 개발자들이 네트워크상의 다수의 공헌자들의 성과를 관리하는 방식으로 개발이 되었는데, 이것이 "오픈소스 개발방법론"의 시초가 되었어요.
이후 BSD유닉스가 발전하는 것과 더불어 유닉스 또한 개별적으로 발전해요. 상업적 이용을 목표로 한 AT&T는 결국 BSD 유닉스에게 법적 소송을 걸어요. 그렇지만, AT&T 또한 저작권 문제로 맞소송을 받게 되었고, BSD 유닉스와 AT&T는 "BSD 유닉스에서 AT&T 사의 고유한 코드를 제거하면 유닉스라는 상호를 사용할 수 있도록 한다"라는 내용으로 합의해요. 1989년에 BSD 유닉스는 AT&T의 소스를 모두 제거하고 새로운 라이선스인 BSD 라이선스를 가진채로 AT&T로부터 완전히 벗어나요. 현재에도 4.4BSD와 더불어 NetBSD, FreeBSD, OpenBSD와 같은 운영체제를 오픈 소스로 배포하고 있어요.
비록 처음에는 소스 강독으로 시작되었지만 개발과정에서 오픈 소스 개발방법론을 처음으로 사용하고 끝끝내 오픈 소스로 함으로써 오픈 소스 소프트웨어 운동에 적잖은 영향을 주었어요.

후기

자유소프트웨어의 성장과 오픈소스의 탄생

Q. 오픈소스는 어떻게 저작권을 관리하나요?

A. 앞서 설명한 **카피레프트** 라는 개념은 추상적이라서 실제 소스코드에서 바로 사용하지는 못했어요. **오픈소스 라이선스** 는 카피레프트를 구체화시켜 실제로 소프트웨어의 저작권을 명시하기 위해 탄생한 라이선스에요. 주로 사용자에게 소스코드의 사적인 이용과 수정 및 배포, 상업적 이용을 허락해 주지만 라이선스나 저작권을 명시해야 한다는 제약조건을 가지고 있어요. 대표적인 오픈소스 라이선스로 다음과 같은 것들이 있어요.

GNU General Public License(GPL)

GNU 프로젝트로 배포하는 소프트웨어에 적용할 목적으로 리처드 스톨만이 만들었어요. 자유소프트웨어재단에서 만든 라이선스답게 가장 엄격한 제약조건이 있어요. 라이선스 및 저작권과 변경사항을 명시해야 하고, GPL 소스코드를 수정하거나 GPL 소스코드를 활용한 소프트웨어 모두 GPL로 공개해야 해요.

MIT License / Berkely Software Distribution(BSD) License

각각 미국 메사추세츠 공과대학교, 버클리의 캘리포니아 대학교에서 배포하는 라이선스에요. 매우 느슨한 조건을 가지고 있기 때문에 널리 쓰이고 있어요. 라이선스와 저작권을 명시해야 해요.

Beerware License

"뭐 딱히 조건은 없고, 제 코드 잘 썼으면 나중에 만날 때 맥주나 한잔 사줘요!" 라는 의미의 다소 장난기 섞인 라이선스로, 가장 낮은 제약조건을 가지고 있어요. 제약없이 소스코드를 마음껏 사용할 수 있어요.

Q. 자유소프트웨어를 이용한 사업도 있나요?

A. OSS는 소스를 공유하기 때문에 클로즈드 소프트웨어처럼 프로그램 자체를 판매하는 것은 의미가 없어요. 그러나 오픈소스만의 특징을 이용해 클로즈드 소프트웨어에서는 못하는 사업을 할 수 있어요. 바로 SW 지원을 해주는 사업이에요. 독점 소프트웨어는 대부분 소프트웨어를 소유한 회사 한 곳에서만 지원해줄 수 있어서 독점의 횡포의 위험이 있고, 서비스가 불만족스러워도 달리 선택권이 없지만 자유소프트웨어는 프로그램의 지원을 위한 시장도 자유롭게 존재해요. 양질의 프로그램을 사용하면서 서비스가 필요하면 기술

지원을 제공해주는 사업자를 고를 수가 있어요. 존 길모어와 마이클 티만이 창업한 시그너스 솔루션 *(Cygnus Solutions)* 이라는 회사는 이 이점을 노려 GNU 자유 소프트웨어와 관련한 컨설팅과 서비스를 판매했어요. 시그너스라는 이름은 창업자의 친구에게서 무작위로 추천받은 GNU와 관련된 이름들 중 하나인데(cyGNUs), 나중에 *Cygnus, Your GNU Support* 라고 재귀적으로 멋지게 정의되었어요. 시그너스는 소프트웨어 지원 외에도 GNU C Compiler 프로젝트에 기여하고 Cygwin이라는 윈도우용 유닉스 에뮬레이터를 개발하다가 1999년 레드햇에 합병되었어요.

레드햇 *(Red hat)* 역시 오픈소스로 사업을 하면서 소프트웨어를 무료로 제공하고 지원을 유료로 하는 기업이에요. 리눅스를 일반 사용자도 쓰기 쉽도록 배포해 초기 리눅스 확산에 큰 기여를 해주었어요. 레드햇은 오픈소스만으로 2016년 매출 20억 달러를 달성하며 오픈소스가 소프트웨어 산업을 망칠 것이라는 우려를 정면으로 반박했어요.

Q. 자유소프트웨어와 오픈소스 소프트웨어에는 무슨 차이점이 있나요?

A. 영어단어 *Free* 에는 무료라는 의미도 있어서 '자유소프트웨어(Free Software)' 라고 하면 그때의 사람들은 무료를 떠올렸어요. 돈을 받고 팔 수 없는 것이라 생각했죠. 물론 이건 잘못된 생각이에요. 이것과 비슷한 개념의 프로그램은 프리웨어(Freeware)라는 이름을 따로 가지고 있어요. 자유소프트웨어운동의 기본 철학인 **소프트웨어가 공개되어 있고, 사용할 수 있다** 는 의미 전달이 필요했어요. 이러한 연유로 '오픈소스 소프트웨어' 라는 개념이 만들어졌어요. 자유소프트웨어의 철학을 극도로 활용한 리눅스가 등장하면서 오픈소스의 의미도 확장되었어요. 커뮤니티를 형성하여 서로 협력해서 개발하는 방식인 '오픈소스 프로젝트'라는 개념이 생겼어요.

자유소프트웨어운동에서 오픈소스 소프트웨어운동으로 바뀌면서 가치관에도 약간의 변화가 생겼어요. 서로 협동해서 더 나은 소프트웨어를 만들자 는 취지에는 변함이 없지만 자유소프트웨어 진영은 상업화를 싫어했어요. 반면 오픈소스와 상업화가 양립할 수 있다고 생각했어요. 때문에 일부 자유소프트웨어 진영에서는 오픈소스 용어의 사용을 거부하고 자유소프트웨어를 고수하기도 해요. 하지만 양측이 서로를 적이라고 생각하지는 않아요. 두 진영의 적은 명백히 독점소프트웨어에요.

Q. 당시 오픈소스 SW에 참여한 기업이 있나요?

A. 리눅스가 세상에 알려지고 진취적인 몇몇 작은 회사들이 오픈소스로 소프트웨어를 개발했겠지만, 오픈소스 참여를 선언한 첫 대형 기업은 **넷스케이프** 라고 알려져 있어요. 당시 넷스케이프는 한창 마이크로소프트(MS)의 인터넷익스플로러(IE)와 웹 브라우저 경쟁을 하고 있었어요. 그러나 MS의 윈도우즈에 IE 끼워팔기 전략에 밀려 점유율이 점점 떨어졌고, 이것에 변화를 주기 위해 정책을 오픈소스로 바꾸었어요. 비록 이론처럼 좋은 결과를 거두진 못했으나 이를 기반으로 지금의 모질라(Mozilla)가 탄생했어요. 모질라사의 유명한 제품인 파이어폭스도 오픈소스 브라우저예요.

리눅스가 성장할 때 즈음에는 오라클, 시스코, 닷컴 등의 업체들이 리눅스를 지원하기 시작했어요. 소프트웨어 회사에서 리눅스를 지원한다는 것은 리눅스에 신빙성이 생겼다는 것과 같아요. 리누스 토발즈가 포춘 잡지의 표지를 장식했고, 뉴스에서는 오픈소스라는 단어를 밥먹듯이 언급했어요. 오픈소스 역사에 한 획을 그은 큰 사건이에요.

오픈소스의 현재 및 미래

Q. 현재 오픈소스 소프트웨어는 국내에서 어느 정도까지 발전하였나요?

Figure 8: SOSCON

A. 우선 한국 정부에서는 조금씩 오픈소스 기술 지원을 늘리고 있어요. 먼저 미래창조과학부 산하 공공기관인 정보통신산업진흥원(NIPA, National IT Industry Promotion Agency)은 국내에서 예산을 들여 가장 적극적으로 지원해주는 곳인데요, 현재 미래창조과학부 산하 기관인 정보통신산업진흥원에는 내부에 '공개SW역량프라자'가 존재하며 여기서 오픈소스 기술에 대한 세미나, 컨설팅, 개발자대회 등을 지원하고 있어요. 그 중 가장 눈에 띄는 것은 '오픈소스 프런티어'라는 사업인데, 이 사업에서는 실력 있는 오픈소스 개발자를 발굴하고 급여 및 사무실 임대 혜택 등을 지원하고 있어요. 현재 3기까지 배출했으며, 한 기수당 15 ~ 20명 정도 참여하고 있답니다. 오픈소스 프런티어 출신 개발자들은 이미 존재하던 오픈소스 프로젝트부터 자신이 직접 개발한 오픈소스 프로젝트까지 다양하게 관리한다고 하네요. 정부 뿐만 아니라 국내 기업들도 오픈소스 소프트웨어 도입이 활발히 이뤄지고 있어요. 삼성전자는 2014년 9월 리눅스재단의 후원으로 국내에서 처음으로 오픈소스 컨퍼런스, 일명 'SOSCON'을 개최했어요. 이 컨퍼런스에 참가한 사람들은 사물인터넷(IoT)과 플랫폼 등의 주요 기술 전망뿐만 아니라 오픈소스를 통한 협업과 공유의 가치 등 여러 주제에 대해 발표를 한다고 하네요. 이처럼 삼성전자 외에도 여러 기업들이 오픈소스 기술에 관심을 가지는 결정적 이유는 기업들이 오픈소스 문화에 동참하면서 플랫폼을 확산하고 새로운 시장을 형성하고 있기 때문이라고 해요. 참고로 삼성전자가 공개한 오픈소스 기술들은 '커밋 101'나 '삼성전자 깃허브 계정'에서 볼 수 있다고 하니, 한번씩 보는 것도 좋을 것 같아요.

Q. 미국은 '오픈소스의 선두주자'라는 타이틀을 가지고 있는데요, 어떠한 방식으로 이 타이틀을 얻었나요?

Figure 9: Meaghan Smith

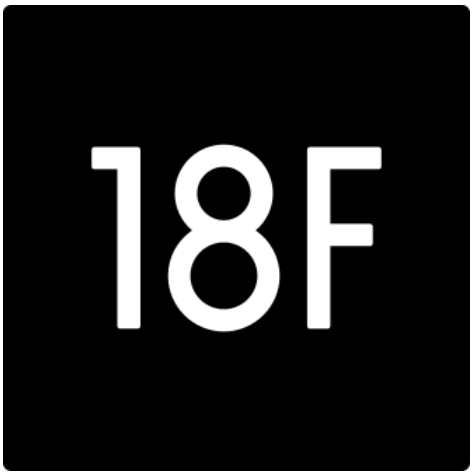


Figure 10: 18F

A. 여러 나라의 정부들은 대형 IT기업들이 오픈소스 기술에 관심을 둘 때, 상대적으로 오픈소스 기술을 사용하거나 직접 만드는 것에 소극적이었어요. 왜냐하면 정부는 기업과 달리 새로운 기술을 만들고 판매하는 곳이 아니었기 때문이죠. 또한 안정성과 보안성을 요구하는 정부에서 오픈소스 기술을 선뜻 사용하기도 어려웠어요. 그러나 미국은 오바마 정부가 들어선 이래로 기술분야에 파격적인 지원이 이루어졌어요. 예를 들어 '모두를 위한 컴퓨터과학'이란 캠페인을 진행하면서 어린이 코딩 교육을 국가차원에서 적극 장려하고 있고, IT 업계 거물들을 영입하는 데도 노력을 기울였어요. 대표적으로 구글 임원 출신인 '메간 스미스'가 미국 정부의 최고기술관리자(CTO, Chief technology officer)로 임명되었으며, 또한 정부 내부에 *'18F'*라는 이름의 기술 혁신팀을 별도로 구성하기도 했어요. 18F : 미국 정부에 기반을 둔 디지털 서비스 대행사. 이것의 목적은 디지털 서비스 및 기술

제품을 제공하는 방식을 변화시키는 기관을 지원하는 것임. 오바마 정부가 이렇게 오픈소스 기술을 적극적으로 지원한 이유는 단 두 가지였어요. 첫째는 정부 기관끼리 협업하여 혁신을 끌어내는 데도 도움이 될 것이며, 둘째로는 정부 기관 스스로 소스코드를 검토하고 품질을 높이는 능력을 키움으로써 코드의 보안성, 안정성, 효율성을 높일 수 있을 것이라 생각해서였어요. 다시 말해, 단순히 비용절감을 위한 정책이 아니라' 정부 스스로 기술력을 높이고 혁신을 위해서 오픈소스 기술을 지원하고 배포하는 것'으로 해석 할 수 있는데요, 이러한 미국의 자세로부터 '오픈소스의 선두주자'라는 타이틀을 얻을 수 있게 되었죠.

Q. 그렇다면 미국을 제외한 다른 나라에서는 오픈소스 소프트웨어가 어떻게 발전되었나요?

A. 우선 영국에 대해 살펴보면, '인터넷의 아버지'이자 '월드와이드웹(WWW)'을 만든 팀 버너스 리가 태어난 나라에요. 이러한 배경 덕에 영국 내에서는 일찍부터 인터넷이 가진 개방성의 의미를 아는 사람이 많고, 오픈데이터, 오픈소스 운동 등도 많이 퍼져 있어요. 정부 역시 마찬가지죠. 영국은 국무조정실에 '거버먼트 디지털 서비스(GDS)라는 팀을 만들어 기술 혁신과 관련된 프로젝트를 진행하고 있어요. GDS는 별도의 홈페이지와 깃허브 계정으로 오픈소스 코드를 계속 공개했는데, 여기에는 디자인 가이드라인, 라이브러리, 인프라 기술, 모니터링 도구 등이 포함돼 있어요. 또한 외부 오픈소스 기술을 정부 시스템과 인프라에 적용해야 한다는 가이드라인도 공개한 상태에요. 영국 정부는 2016년 7월 구인구직 사이트 링크드인에 '오픈소스 팀을 이끄는 팀장을 구한다'는 공지도 직접 올려 세간의 관심을 받기도 했어요. 여기서 찾고 있는 인재를 보면 단순히 공무원이 아닌 업계에서 개발자로 실무 경험이 많고 프로그래밍 실력 및 협업 능력이 뛰어난 사람이에요. 이 채용 페이지만 봐도 영국 정부가 외부 오픈소스 커뮤니티와 열심히 소통하려 노력하고, 내부 기술적인 문제를 오픈소스로 적극적으로 해결하려는 의지를 엿볼 수 있어요. 불가리아 정부는 2016년 7월부터 오픈소스 소프트웨어 사용과 개발을 권장하는 법을 시행하고 오픈소스 소프트웨어 도입을 정책적으로 장려하고 있어요. 사실 일부 진영에서는 정부가 오픈소스 기술을 도입하는 것은 매우 위험한 일이라고 주장하기도 했는데요, 가장 큰 이유가 보안 때문이라고 합니다. 오픈소스 정책 자문위원 중 한 명인 '보이다르 보자노프'는 "불가리아 정부는 그동안 '은닉을 통한 보안'이라는 접근방법으로 보안정책을 수립했으나 수많은 보안 취약점이 정부 웹사이트에서 발견되었고, 수년 동안 고쳐지지 않고 방치되어 패치가 적용되지 않고 있다. 그래서 앞으로는 이러한 접근 방식보단 오픈소스 기술을 선택해 개발단계에서 아예 보안 취약점을 즉각적으로 발견하고 수정하는 방식을 택할 것이다."라고 말했어요. 여기서 오픈소스 기술 채택으로 인한 2가지 장점이 존재하는데, 첫째는 정부가 국민의 세금으로 구입하고 관리하는 기술과 제품들을 투명하게 공개하는 것이 더 쉬워져요. 둘째는 필요 없는 소프트웨어의 구입을 막는 효과도 있어요. 이러한 이유로 불가리아 정부는 오픈소스 소프트웨어를 도입하는 문화가 정착되면 좀 더 합리적인 소프트웨어 구입과 활용이 가능해질 것이라고 예상하고 있어요. 인도 정부는 2015년부터 오픈소스 기술을 사용하는 정책을 도입했어요. 이는 비용 절감을 위한 결정이라고 해요. 사실 인도는 2006년 부터 1만 2500여 개 학교에 리눅스 같은 기술을 교육시키며 재단 설립자인 리차드 스톨만이 직접 인도에 방문해 강연을 진행하기로 했는데, 인도는 이러한 교육을 통해 인도 정부가 특정 상업 기술에 종속되지 않도록 대비하고 있어요. 최근에는 한발 더 나아가 직접 오픈소스 기술을 개발하고 배포하는 프로젝트를 준비하고 있어요. 이를 위해 깃허브와 유사한 협업 플랫폼을 개발하고 있으며, 해당 협업 플랫폼

내에서 누구나 소스코드 수정을 요청하고 기여할 수 있는 구조를 구축하고 있어요. 이처럼 세계 각국에서 오픈소스를 통한 기술 공유와 지원에 적극적으로 나서고 있는데, 앞으로 더 많은 국가에 확산되어 정부와 국민들이 오픈소스를 통해 서로 소통하고 발전했으면 하는 바램입니다!

Q. 현재까지 이용되고 있는 과거의 오픈소스 소프트웨어는 무엇이 있나요?

Figure 11: MySQL

A. 필자는 MySQL에 대해 소개하고자 해요. MySQL은 전세계적으로 가장 널리 사용되고 있는 오픈소스 데이터베이스이며, MySQL AB사가 개발하여 배포, 판매하고 있는 데이터베이스에요. MySQL 데이터베이스는 GPL(GNU Public License)를 준수하는 오픈소스 데이터베이스이며, GPL을 준수해서 사용하는 모든 사용자에게 무료로 배포되고 있어요. 최근에 이 MySQL을 활용해 은행 IT에 새 이정표를 제시한 은행이 있어요. 바로 카카오뱅크인데요, 현재 대다수 시중은행은 x86서버와 오픈소스SW를 '검증되지 않은 솔루션'으로 치부해왔어요. 이는 검증된 안정성을 우선시하는 업계 인식때문이었는데, 2017년 7월 27일에 공식 출범한 카카오뱅크는 전산시스템을 MySQL 계열 오픈소스 DB을 사용해 핵심업무 관련 DB인프라 상당 부분을 담당한다고 해요. 보수적인 세계에서 카카오뱅크의 대규모 오픈소스DB 인프라는 엄청난 모험 또는 무모한 시도로 비치기 십상인데, 5일만에 100만 계좌를 돌파할만큼 많은 접속을 별 탈 없이 소화했어요. 현재까지도 '무장애' 운영 기록을 세우고 있으며, 이러한 카카오뱅크의 오픈소스DB 활용은 저비용으로도 금융권 전산시스템에 요구되는 고안정성을 충족할 수 있는 메시지로 비치죠. 출범 약 1년 전, 카카오뱅크도 다른 은행과 같이 오라클 DBMS를 채택했어요. 그러나 22%에 달하는 유지보수요율이 금융사 입장에서는 부담이 되었고, 처음에는 2017년 1월 오픈이 목표였지만 MySQL로 데이터베이스를 바꾸면서 출시예정일도 그만큼 늦어졌다고 볼 수 있어요. 현재까지 이용되고 있는 과거의 오픈소스 소프트웨어는 MySQL 말고도 gcc(GNU Compiler Cillection) 컴파일러, PHP 등이 있답니다.

Q. 앞으로 오픈소스 소프트웨어는 어떻게 이용될까요?

Figure 12: Drone

Figure 13: Autonomous Car

A. 오픈소스로 개발하면 비용이 줄어들고, 리스크도 줄여줘요. 또한 오픈소스는 빠르고 유연한 개발이 가능하며, 신뢰성과 안정성이 높죠. 따라서 앞으로도 오픈소스는 어떠한 방식으로든 여러 기업들과 정부, 개인들이 사용할 거에요. 미래에 오픈소스는 다양한 방식으로 활용될 수 있어요. 대표적으로 자율주행차를 예로 들 수 있는데요, 현 시점에서 자동차는 운송기구가 아닌 SW라는 정의가 더 어울리게 되었어요. 여러 전문가들은 미래의 자동차에는 SW가 핵심 역할을 하게 될 것이며, 이 과정에서 오픈소스는 자동차의 빠른

혁신과 신뢰성을 끌어올릴 것으로 전망하고 있어요. 또한 요새 한창 뜨고 있는 드론에도 적용될텐데요, 약 6개월 전에는 드론 개발자를 위한 오픈소스 임베디드 보드인 '비글본 블루'가 출시 됐어요. 이 보드는 드론의 두뇌 역할을 한다고 합니다. 이와 같이 보드같은 오픈소스들이 더 많이 출시된다면, 미래에 드론이 비행하는데 있어서 충돌사고 없이 제 역할을 잘 수행할 수 있겠네요. 필자의 개인적인 소견으로는 오픈소스를 이용한 개인 사업이 늘어날 것이라 생각해요. 현재 오픈소스 소프트웨어가 기업 IT시장의 핵심으로 떠오르며 여러 유수의 기업들이 오픈소스에 관심을 갖고 있는데요, 이로 인해 오픈소스 시장이 더 활발해지고 그 환경 또한 더욱 더 발전하고 있어요. 또한 삼성과 같은 기업들이 위에서 언급했던 soscon과 같은 오픈소스 컨퍼런스도 개최하면서 꾸준히 오픈소스에 대한 관심이 늘고 있어요. 가까운 미래에는 젊은 청년들이 오픈소스 소프트웨어를 적극적으로 활용한 창업이나 스타트업의 Boom이 일어날 것이라 생각해요.

출처

책

10,000 피트에서 바라본 오픈 소스 소프트웨어, Androutsellis-Theotokis, 김종배 역, 한티미디어 2015 한국 오픈 소스 개발자들 이야기, 송우일, 인사이트, 2013
오픈소스 2.0, 김종배, 김두연, 류성열, 한티미디어, 2010
리눅스 비즈니스.com, 클리프 밀러, 이규원 역, 세종서적, 2000
거의 모든 인터넷의 역사, 정지훈, 메디치미디어, 2014

사이트

http://www.gnu.org/philosophy/philosophy.html
https://ko.wikipedia.org/wiki/사유_소프트웨어
http://egloos.zum.com/jamestic/v/2489732
https://github.com/ganadist/catb/blob/master/md/01_brief_history.markdown
https://joone.net/2017/01/
https://en.wikipedia.org/wiki/Unix
https://joone.net/tag/unix/
https://en.wikipedia.org/wiki/Open_Letter_to_Hobbyists
https://www.slideshare.net/AndreaTino1/halloween-documents
https://www.ituonline.com/course/linux-unix-certification-training-bundle
https://www.salon.com/2000/05/16/chapter_2_part_one/
http://www.gnu.org/gnu/thegnuproject.ko.html
https://www.freebsd.org
http://www.bloter.net/archives/209318
https://en.wikipedia.org/wiki/Cygnus_Solutions
https://byline.network/2016/04/1-95/
https://www.gnu.org/philosophy/free-software-for-freedom.ko.html