오픈소스 프로젝트에 BSD 스타일 라이선스를 사용해야 하는 이유

차례

1. 소개. 1
2. 아주 간략한 오픈소스 역사. 1
3. BSD 라이선스 관점에서 본 유닉스. 2
4. FreeBSD의 현황과 BSD 라이선스. 2
5. GPL의 유래. 2
6. 리눅스와 LGPL의 기원. 3
7. 오픈 소스 라이선스와 방치 문제. 3
8. 라이선스로 할 수 없는 일. 3
9. GPL의 장점과 단점. 4
10. BSD의 장점. 4
11. BSD 라이선스 사용에 대한 구체적 권장사항. 5
12. 결론. . . . . 6 참고 문헌. 6

1. 소개

이 문서는 소프트웨어와 데이터에 BSD 스타일 라이선스를 사용하는 사례를 제시하며, 특히 GPL 대신 BSD 스타일 라이선스를 사용할 것을 권장합니다. 이 문서는 BSD와 GPL 오픈 소스 라이선스 소개 및 요약으로 생각하셔도 됩니다.

2. 아주 간략한 오픈소스 역사

“오픈 소스”라는 용어가 사용되기 훨씬 전부터 소프트웨어는 프로그래머들의 느슨한 연합에 의해 개발되고 자유롭게 교환되었습니다. 1950년대 초부터 SHARE 및 DECUS와 같은 조직이 컴퓨터 하드웨어 회사가 하드웨어 제품에 번들로 제공하는 소프트웨어의 대부분을 개발했습니다. 당시 컴퓨터 회사들은 하드웨어를 주축으로 사업을 하고 있었기 때문에 소프트웨어 비용을 절감하고 더 많은 프로그램을 사용할 수 있게 하면 컴퓨터 회사의 경쟁력이 높아졌습니다.

이 모델은 1960년대에 바뀌었습니다. 1965년 ADR은 하드웨어 회사로부터 독립하여 최초의 라이선스 소프트웨어 제품을 개발했습니다. ADR은 원래 IBM 고객이 개발한 무료 IBM 패키지와 경쟁하고 있었습니다. ADR은 1968년 소프트웨어 특허를 획득했습니다. 프로그램의 공유를 막기 위해 제품 수명 기간 동안 요금을 나눠내는 장비 리스 방식으로 제품을 제공했습니다. 따라서 ADR은 소유권을 유지하고 재판매 및 재사용을 통제할 수 있었습니다.

1969년 미국 법무부는 IBM이 무료 소프트웨어를 IBM 하드웨어에 번들로 제공함으로써 비즈니스를 파괴했다는 혐의로 IBM을 기소했습니다. 이 소송의 결과로 IBM은 소프트웨어를 번들에서 분리하였고, 소프트웨어가 하드웨어와 분리된 독립적인 제품이 되었습니다.

1968년 인포매틱스(Informatics)는 최초의 상용 킬러 앱을 출시하여 소프트웨어 제품, 소프트웨어 회사, 매우 높은 수익률이라는 개념을 빠르게 정립했습니다. 인포매틱스는 현재 컴퓨터 업계에서 표준이 된 영구 라이선스를 개발했는데, 이 라이선스는 소유권이 고객에게 이전되지 않습니다.

1

3. BSD 라이선스 관점에서 본 유닉스

최초의 유닉스 구현을 소유한 AT&T는 미국의 반독점법에 묶여 공공 규제를 받는 독점기업이었기 때문에 법적으로 소프트웨어 시장에 제품을 판매할 수 없었습니다. 하지만 학술 기관에는 미디어 가격(순수한 패키지 가격)을 받고 제공할 수 있었습니다.

한 OS 컨퍼런스에서 유닉스를 공적으로 사용할 수 있다는 사실이 알려지면서 많은 대학들은 빠르게 유닉스를 도입했습니다. 유닉스가 매우 저렴한 16비트 컴퓨터인 PDP-11에서 실행되고 시스템 프로그래밍에 매우 적합한 고급 언어로 코딩되어 있었다는 점이 큰 도움이 되었습니다. DEC PDP-11은 고객이 자체 OS를 쉽게 작성할 수 있도록 설계된 개방형 하드웨어 인터페이스를 갖추고 있었는데, 당시에는 이러한 인터페이스가 일반적이었습니다. “좋은 하드웨어가 있으면 소프트웨어는 하늘에서 내려온다”는 DEC 창립자 켄 올슨(Ken Olsen)의 유명한 명언이 있습니다.

유닉스의 창시자 켄 톰슨(Ken Thompson)은 1975년 모교인 캘리포니아 버클리 대학교(UCB)로 돌아와 커널을 한 줄 한 줄 가르쳤습니다. 그 결과 BSD(버클리 표준 배포판)로 알려진 진화하는 시스템이 탄생했습니다. UCB는 유닉스를 32비트로 변환하고, 가상 메모리를 추가하고, 인터넷의 기본이 되는 TCP/IP 스택 버전을 구현했습니다. UCB는 “BSD 라이선스”라는 이름으로 미디어 비용으로 BSD를 사용할 수 있게 했습니다. 과거에 AT&T에서 유닉스를 구입한 고객들은 이제 UCB에서 BSD 테이프를 주문하게 되었습니다.

1980년대 중반, AT&T에 대한 정부의 반독점 소송으로 인해 AT&T가 해체되었습니다. AT&T는 여전히 유닉스를 소유하고 있었으며 이제 이를 판매할 수 있게 되었습니다. AT&T는 공격적인 라이선스 확보에 나섰고, 당시 대부분의 상용 유닉스는 AT&T에서 파생된 제품이 되었습니다.

1990년대 초, AT&T는 BSD와 관련된 라이선스 위반으로 UCB를 고소했습니다. 이 과정에서 UCB는 AT&T가 BSD로 인한 많은 개선 사항을 승인이나 대가 없이 AT&T의 제품에 통합했다는 사실을 발견했고, AT&T와 UCB 간의 오랜 법정 소송이 이어졌습니다. 이 기간 동안 일부 UCB 프로그래머들은 BSD와 관련된 모든 AT&T 코드를 다시 작성하는 프로젝트에 착수했습니다. 이 프로젝트는 BSD 4.4-lite라는 시스템을 탄생시켰습니다(라이트라고 명명한 것은 6개의 주요 AT&T 파일이 부족해 완전한 시스템이 아니었기 때문입니다).

조금 후에 닥터 돕스(Dr. Dobbs) 잡지에 실린 긴 기사 시리즈에서는 누락된 6개의 4.4 라이트 파일에 대한 BSD 라이선스 대체 파일과 함께 BSD 파생 386 PC 버전의 유닉스에 대해 설명했습니다. 386BSD로 명명된 이 시스템은 전직 UCB 프로그래머 윌리엄 졸리츠(William Jolitz)가 만든 것입니다. 이 시스템은 현재 사용 중인 모든 PC BSD의 기반이 되었습니다.

1990년대 중반, 노벨(Novell)은 AT&T의 유닉스 권리를 인수했고, 소송을 종료하기로 (당시엔 비밀리에) 합의가 이루어졌습니다. UCB는 곧 BSD에 대한 지원을 종료했습니다.

4. FreeBSD의 현황과 BSD 라이선스

지난 몇 년 동안 FreeBSD에 적용된 소위 new BSD license는 사실상 프로그램이나 그 소스로 무엇이든 할 수 있지만, 그 어떤 보증도 없고, 어떤 저작자도 책임을 지지 않는다는 선언입니다(기본적으로 누구에게도 소송을 제기할 수 없습니다). 이 새로운 BSD 라이선스는 제품 상용화를 장려하기 위한 것입니다. 모든 BSD 코드는 코드의 가용성이나 향후 행위에 대한 제한 없이 판매되거나 독점 제품에 포함될 수 있습니다.

새로운 BSD 라이선스를 “퍼블릭 도메인”과 혼동하지 마세요. 퍼블릭 도메인의 항목도 누구나 무료로 사용할 수 있지만 소유자가 없습니다.

5. GPL의 유래

1980년대 말과 1990년대 초, 유닉스의 미래가 매우 혼란스러웠을 때 라이선스를 중요하게 고려한 GPL이 나타났습니다.

Emacs의 개발자인 리처드 스톨먼(Richard Stallman)은 MIT에서 연구실을 자체 개발 시스템에서 독점 시스템으로 전환할 당시 직원으로 근무하고 있었습니다. 스톨만은 시스템에 사소한 개선 사항을 법적으로 추가할 수 없다는 사실을 알게 되자 화가 났습니다. (스톨만의 동료들 중 다수는 MIT에서 개발하고 MIT가 라이선스를 부여한 소프트웨어를 기반으로 한 두 개의 회사를 설립하기 위해 회사를 떠났는데, 이 소프트웨어의 소스 코드에 대한 접근 권한에 대해 이견이 있었던 것으로 보입니다). 스톨만은 상용 소프트웨어 라이선스에 대한 대안을 고안해냈고 이를 GPL, 즉 “GNU 공중 라이선스”라고 불렀습니다. 또한 그는 비영리 재단인 자유 소프트웨어 재단(FSF)을 설립하여 모든 관련 소프트웨어를 포함한 전체 운영 체제를 독점 라이선스의 적용을 받지 않는 것으로 개발하고자 했습니다. 이 시스템은 “GNU는 유닉스가 아니다”라는 뜻에서 GNU라고 불렀습니다.

GPL은 표준 독점 라이선스와 반대되는 개념으로 설계되었습니다. 이를 위해 GPL 프로그램을 수정할 경우 해당 프로그램의 소스를 사용자가 원하면 접근할 수 있도록 GPL 커뮤니티에 환원해야 했고, GPL 코드를 사용하거나 링크하는 모든 프로그램은 GPL에 따라 사용해야 했습니다. GPL은 소프트웨어가 독점적인 소유물이 되는 것을 막기 위한 것입니다. GPL의

2

마지막 단락에 명시되어 있듯이:

“본 일반 공중 사용 허가서는 귀하의 프로그램을 독점 프로그램에 통합하는 것을 허용하지 않습니다.”[1]

GPL은 복잡한 라이선스이므로 GPL을 사용할 때 지켜야 할 몇 가지 규칙이 있습니다:

• 소프트웨어 배포, 지원 또는 문서화 비용은 얼마든지 청구할 수 있지만 소프트웨어 자체는 판매할 수 없습니다.

• 일반적인 원칙상, 프로그램을 컴파일하는 데 GPL 소스가 필요한 경우 해당 프로그램은 반드시 GPL에 따라야 합니다. 또한 GPL 라이브러리에 정적으로 링크하려면 프로그램은 GPL을 따라야 합니다.

• GPL에 따르면 GPL 소프트웨어와 관련된 모든 특허는 모든 사람이 자유롭게 사용할 수 있도록 라이선스가 부여되어야 합니다.

• 여러 프로그램을 하나의 디스크에 넣을 때처럼 단순히 소프트웨어를 한데 모으는 것은 GPL이 아닌 프로그램에 GPL이 적용된 프로그램을 포함하는 것으로 간주하지 않습니다.

• 프로그램의 결과물은 2차적 저작물로 간주되지 않습니다. 따라서 gcc 컴파일러는 법적 문제 없이 상업적 환경에서 사용할 수 있습니다.
• Linux 커널은 GPL에 따라 사용되므로 Linux 커널과 정적으로 링크된 모든 코드는 GPL을 준수해야 합니다. 로드 가능한 커널 모듈을 동적으로 링크하면 이 요구 사항을 우회할 수 있습니다. 이를 통해 기업은 바이너리 드라이버를 배포할 수 있지만 특정 버전의 Linux 커널에서만 작동한다는 단점이 있습니다.

부분적으로는 그 복잡성으로 인해 오늘날 전 세계 많은 지역에서 리눅스 및 관련 소프트웨어와 관련된 GPL 적법성이 무시되고 있습니다. 이로 인한 장기적인 파급 효과는 불분명합니다.

6. 리눅스와 LGPL의 기원

상업용 유닉스 전쟁이 치열하게 벌어지는 동안 리눅스 커널은 PC 유닉스 클론으로 개발되었습니다. 리누스 토발즈(Linus Torvalds)는 리눅스가 존재할 수 있었던 이유로 GNU C 컴파일러와 관련 GNU 도구의 존재를 꼽습니다. 그는 리눅스 커널을 GPL로 배포했습니다.

GPL은 GPL에 따라, 어떤 코드에 정적으로 링크하는 모든 것도 GPL에 따를 것을 요구한다는 사실을 기억하세요. 따라서 이 코드의 소스는 프로그램 사용자가 사용할 수 있도록 제공되어야 합니다. 그러나 동적 링크는 GPL 위반으로 간주되지 않습니다. 독점 애플리케이션을 Linux에 설치하라는 압력이 압도적으로 커졌습니다. 이러한 애플리케이션은 종종 시스템 라이브러리와 링크해야 합니다. 그 결과 LGPL(“라이브러리”, 이후 “Lesser”, GPL로 이름이 변경됨)라는 수정된 버전의 GPL이 탄생했습니다. LGPL은 독점 코드를 GNU C 라이브러리인 glibc에 링크할 수 있도록 허용합니다. LGPL이 적용된 라이브러리에 동적으로 링크된 소스 코드를 공개할 필요는 없습니다.

임베디드 시스템에서 자주 요구되는 것처럼 애플리케이션을 glibc와 정적으로 링크하는 경우 애플리케이션을 독점적으로 유지할 수 없으므로 소스를 공개해야 합니다. GPL과 LGPL 모두, 라이선스의 관리를 받는 코드를 수정할 경우 이를 공개하도록 요구합니다.

7. 오픈 소스 라이선스와 방치 문제

독점 소프트웨어와 관련된 심각한 문제 중 하나는 ‘방치(Orphaning)’로 알려져 있습니다. 이는 어떤 사업의 실패 또는 제품 전략의 변경이 그에 의존하는 시스템의 거대한 피라미드에 영향을 미치는 것을 말하며, 이로인해 기업이 완전히 통제력을 상실하여 발생할 수도 있습니다. 수십 년간의 경험에 따르면 현재의 시장 상황과 전략은 언제든 급변할 수 있기 때문에 소프트웨어 공급업체의 일시적인 규모나 성공이 해당 소프트웨어의 지속적인 사용을 보장하지 않습니다.

GPL은 독점 지적 재산권과의 연결 고리를 끊어 방치화를 방지하고자 합니다.

BSD 라이선스는 소규모 회사에게 법적 복잡성이나 비용 없이 상용기업에 의해 성능을 보장받는 소프트웨어와 동등한 효과를 제공합니다. 만약 BSD 라이선스를 받은 프로그램이 방치 된다면, 다른 회사가 그 프로그램에 의존하고 있는 프로그램을 독점적인 방식으로 인수할 수 있습니다. 개발 프로세스가 단일 회사나 제품 라인의 생존에 의존하지 않기 때문에, 소규모 비공식 컨소시엄에 의해 BSD 코드베이스가 유지되는 경우 더 나은 상황이 발생합니다. 소스 코드의 단순한 물리적 가용성보다 개발 팀이 정신적으로 그 영역에 있을 때의 생존 가능성이 훨씬 더 중요합니다.

8. 라이선스로 할 수 없는 일

어떤 라이선스도 미래의 소프트웨어 가용성을 보장할 수는 없습니다. 저작권자는 전통적으로 언제든지 저작권 조건을 변경할 수 있지만, BSD 커뮤니티에서는 그러한 시도가 단순히 소스를 포크하게 만든다고 가정하고 있습니다.

GPL은 라이선스 취소를 명시적으로 허용하지 않습니다. 그러나 한 회사(Mattel)가 GPL 저작권(cphack)을 구입한 후 저작권 전체를 취소하고 법정 소송을 제기하여 승소한 사례가 있습니다[2]. 즉, 해당 저작권에 기반한 배포판과 2차 저작물 전체를 합법적으로 취소한 것입니다. 더 크고 분산된 배포판에서도 이런 일이 일어날 수 있을지는 아직 미지수이며, 해당 소프트웨어가 실제로 GPL에 따라 배포되었는지에 대해서도 약간의 혼란이 있습니다.

또 다른 예로, 레드햇(Red Hat)은 FSF 컴파일러 도구 개발을 맡았던 엔지니어링 회사인 시그너스(Cygnus)를 인수했습니다. 시그너스를 인수할 수 있었던 이유는 이 회사가 GNU 소프트웨어에 대한 지원을 상품으로 판매하는 비즈니스 모델을 개발했기 때문입니다. 이를 통해 약 50명의 엔지니어를 고용할 수 있었고, 수정 사항을 주도적으로 제안하여 프로그램의 방향을 변경할 수 있었습니다. 도널드 로젠버그(Donald Rosenberg)는 “GPL과 같은 라이선스를 사용하는 프로젝트는… 누군가 더 나은 버전의 코드를 만들어서 기존 소유자보다 더 빨리 프로젝트를 장악할 수 있다는 끊임없는 위협에 시달리고 있다”고 말합니다. [3]

9. GPL의 장점과 단점

GPL을 사용하는 일반적인 경우는 gcc 컴파일러를 수정하거나 확장할 때입니다. 이는 특히 모든 소프트웨어 비용이 오버헤드로 간주될 가능성이 높은 환경에서, 일회성으로 특수한 CPU로 작업할 때와 같이 아주 적은 수의 사람만이 변경된 컴파일러를 사용하는 상황에 적합합니다.

GPL은 최종 사용자에게 매우 저렴한 제품을 제공하는 “박리다매” 환경에서 CD를 판매하는 소규모 회사와 같은 곳에 매력적입니다. 또한 GPL이 적용된 지적 재산권 세계에서 문서를 포함한 다양한 형태의 기술 지원을 제공함으로써 생존을 기대하는 기업에게도 매력적입니다.

GPL을 따라 충분히 공개하지 않거나 의도치 않게 GPL을 사용하는 것은 소프트웨어 회사를 약화시키려는 대기업에 매우 유리합니다. 즉, GPL은 마케팅 무기로 사용하기에 적합하여 전반적인 경제적 이익을 감소시키고 독과점 행위에 기여할 수 있습니다.

GPL은 소프트웨어를 상업화하여 수익을 창출하고자 하는 사람들에게 실질적인 문제를 야기할 수 있습니다. 예를 들어, 대학원생이 자신의 연구 결과를 상용화하기 위해 직접 회사를 설립하거나, 유망한 연구 프로젝트의 상용화를 전제로 회사에 입사하는 데 GPL은 어려움을 가중시킬 수 있습니다.

여러 소프트웨어 표준의 정적연결로 구현되는 작업을 하는 사람들에게 GPL은 표준의 독점적 구현을 사용할 수 없기 때문에 좋지 않은 라이선스인 경우가 많습니다. 따라서 GPL은 GPL 표준을 사용하여 만들 수 있는 프로그램의 수를 줄이는 결과를 낳았습니다. GPL은 독점적 제품을 개발하는 표준 메커니즘을 제공하지 않기 때문입니다. (Linux 애플리케이션은 정적으로 링크하지 않고 트랩 기반 API를 사용하기 때문에 적용되지 않습니다.)

GPL은 프로그래머들이 진화하는 프로그램 모음에 기여한 후, 이 모음 배포 및 지원에 경쟁하도록 만듭니다. 이러한 상황은 상업적 커스터마이징이나 기존(non-GPL) 라이선스에 따른 레거시 표준과의 통합이 필요한 매우 다양한 환경에 적용될 수 있는, 많은 필수 핵심 시스템 표준에 대해서는 비현실적입니다. 실시간 시스템은 정적으로 연결되는 경우가 많기 때문에 많은 임베디드 시스템 회사에서는 GPL과 LGPL을 잠재적인 문제로 간주하고 있습니다.

GPL은 수요에 관계없이 연구 및 개발 단계 노력의 가치를 유지하려고 합니다. 이를 통해 연구자와 개발자에게는 혜택을 극대화하고, 더 많은 배포를 통해 혜택을 받을 수 있는 사람들에게는 알 수 없는 비용을 요구합니다.

GPL은 연구 결과가 독점적인 제품으로 전환되는 것을 막기 위해 고안되었습니다. 하지만 이 단계는 전통적인 기술 이전 파이프라인의 마지막 단계로 간주되는 경우가 많으며 이런 과정은 일반적으로 최상의 상황에서도 충분히 어려운 일인데, GPL은 이를 아예 불가능하게 만들었다고 받아들여지고 있습니다.

10. BSD의 장점

BSD 스타일 라이선스는 장기간의 연구 또는 개발 환경이 필요한 기타 프로젝트에 적합한 선택입니다:

• 거의 비용이 없습니다

• 오랜 기간에 걸쳐 계속 개선될 수 있습니다

• 누구나 법적 문제를 최소화하면서 최종 결과물을 상용화할 수 있습니다.

아파치 프로젝트가 라이선스를 결정할 때와 마찬가지로 이 최종 고려사항이 가장 중요한 부분이 될 수 있습니다:

“이러한 유형의 라이선스는 공통 서비스를 위한 프로토콜을 구현하는 참조 코드의 사용을 촉진하는 데 이상적입니다. 우리 중 다수는 HTTP가 살아남아 진정한 멀티파티 표준이 되기를 원하며,

HTTP의 공용화라는 목표를 달성하는 데 도움이 된다면 Microsoft나 Netscape가 HTTP 엔진이나 우리 코드의 다른 구성 요소를 자사 제품에 통합하는 데 조금도 신경 쓰지 않을 것입니다. 이 모든 것은 전략적으로 볼 때 프로젝트가 계속 충분한 추진력을 유지해야 하며, 참여자들이 자신의 코드를 프로젝트에 기여함으로써 더 큰 가치를 실현할 수 있으며, 심지어 독점적으로 유지되었다면 가치가 있었을 코드도 마찬가지라는 것을 의미합니다.”

개발자들은 법적인 문제를 피할 수 있고 코드에 대해 원하는 모든 것을 할 수 있다는 점에서 BSD 라이선스가 매력적이라고 생각하는 경향이 있습니다. 반면, 시스템을 프로그래밍하기보다는 주로 시스템을 사용하거나 다른 사람이 코드를 발전시키기를 기대하거나 시스템과 관련된 업무로 생계를 유지하지 않는 사람들(예: 공무원)은 다른 사람이 개발한 코드를 강제로 공개하고 고용주가 저작권을 보유하지 못하게 하여 소프트웨어를 “묻어버리거나” 방치되게 할 수 있는 GPL이 매력적이라고 생각합니다. 경쟁업체의 도움을 받고 싶다면 GPL이 매력적입니다.

BSD 라이선스는 단순한 선물이 아닙니다. “왜 우리가 경쟁사를 도와주거나 경쟁사가 우리의 작업을 훔치도록 내버려둬야 하는가?”라는 질문은 BSD 라이선스와 관련하여 자주 제기됩니다. BSD 라이선스 하에서, 한 회사가 다른 회사가 전략적으로 간주하는 제품의 틈새 시장을 장악하게 되면, 다른 회사들은 최소한의 노력으로 시장 경쟁과 공정성을 높이는 경쟁적 BSD 프로젝트에 기여함으로써 동등성을 회복하는 것을 목표로 하는 미니 컨소시엄을 구성할 수 있습니다. 이를 통해 각 회사는 경제적 유연성과 효율성에 기여하면서 자신이 제공할 수 있는 이점을 통해 이익을 얻을 수 있다고 믿습니다. 협력하는 구성원들이 이를 더 빠르고 쉽게 수행할 수 있을수록 더 큰 성공을 거둘 수 있습니다. BSD 라이선스는 본질적으로 이러한 행동을 가능하게 하는 최소한도의 복잡성을 가진 라이선스입니다.

완전하고 경쟁력 있는 오픈 소스 시스템을 미디어 비용(배포 비용)으로 널리 사용할 수 있도록 하는 것은 GPL의 핵심 효과이며, 이는 합리적인 목표입니다. BSD 스타일의 라이선스는 개인들로 구성된 임시 컨소시엄과 함께 기술 이전 파이프라인의 배포-종료 단계에 구축된 경제적 예상을 파괴하지 않으면서도 이러한 목표를 달성할 수 있습니다.

11. BSD 라이선스 사용에 대한 구체적 권장사항

• BSD 라이선스는 연구 결과를 널리 배포하고 경제에 가장 큰 혜택을 줄 수 있는 방식으로 이전하는 데 바람직합니다. 따라서 NSF, ONR, DARPA와 같은 연구자금 지원 기관은
자금 지원 연구 프로젝트의 초기 단계에서 소프트웨어, 데이터, 결과물, 개방형 하드웨어에 BSD 스타일의 라이선스를 채택하도록 장려해야 합니다. 또한 구현된 오픈 소스 시스템과 진행 중인 오픈 소스 프로젝트를 중심으로 표준을 형성하도록 장려해야 합니다.

• 정부 정책은 연구에서 배포로 이동하는 데 드는 비용과 어려움을 최소화해야 합니다. 가능한 경우, 지원금은 상업화 친화적인 BSD 스타일의 라이선스에 따라 결과를 사용할 수 있도록 요구해야 합니다.

• 많은 경우, BSD 스타일 라이선스의 장기적인 결과는 연구 결과가 저작권이나 특허로 보호되고 대학의 독점적인 라이선스를 따를 때보다 대학의 연구 헌장에 명시된 목표를 더 정확하게 반영합니다. 연구 결과를 공개하고 상업적으로 성공한 동문들의 기부에 호소함으로써 대학이 장기적으로 재정적으로 더 나은 보상을 받는다는 일화가 존재합니다.

• 기업들은 사실상의 표준을 만드는 것이 중요한 마케팅 기법이라는 것을 오랫동안 인식해 왔습니다. 기업이 시스템을 발전시키는 데 있어 남들과 차별되는 이점을 가지고 있다면 BSD 라이선스는 이러한 특징에 잘 부합됩니다. 이 라이선스는 법적으로 더 많은 사람들에게 매력적으로 다가갈 수 있으며, 회사가 가진 전문성을 통해 회사의 통제권을 보장할 수 있습니다. 만약 누군가가 기술 표준을 훼손하거나 다른 표준을 채택하려고 할 때, GPL은 그러한 행동에 이용할 수 있는 적합한 수단이 될 수 있습니다. 그러나 GPL은 상업적으로 적용 가능한 표준이 아닌 제품군을 장려하기 때문에 장기적으론 해당 표준의 개선에 불이익을 줍니다. 이러한 제품군을 사용하면 상업화 및 법적 문제가 지속적으로 제기됩니다. 또한 어떤 표준은 GPL을 따르고 어떤 표준은 그렇지 않은 경우 표준을 혼합하는 것이 불가능할 수도 있습니다. 진정한 기술 표준은 비기술적인 이유로 다른 표준을 배제하도록 요구해서는 안 됩니다.

• 다른 회사의 상용 제품의 핵심이 될 수 있는 진화하는 표준을 홍보하는 데 관심이 있는 기업은 GPL을 주의해야 합니다. 어떤 라이선스를 사용하든, 결과물인 소프트웨어는 일반적으로 엔지니어링 변경의 대부분을 실제로 수행하고 시스템 상태를 가장 잘 이해하는 사람에게 귀속됩니다. GPL은 결과물에 법적 마찰을 더할 뿐입니다.

• 오픈 소스 코드를 개발하는 대기업은 직원이 이직을 하더라도 그 소프트웨어를 계속 사용할 수 있기 때문에 프로그래머들이 오픈 소스를 높이 평가한다는 사실을 인지해야 합니다. 일부 기업에서는 특히 관련된 소프트웨어가 직접적으로 전략적이지 않은 경우 이러한 행동을 고용 특전으로 장려하기도 합니다. 이는 사실상 잠재적인 기회 비용 손실은 있지만 직접적인 비용은 들지 않는 선급 퇴직금과 같습니다. 직원들이 회사 밖에서 동료들의 찬사를 받기 위해 일하도록 장려하는 것은 때때로 회사에게 있어선 거의 단점이 없는 값싼 이동식 혜택을 제공할 수 있습니다(역자 주: 원문에서 “cheap portable benefit”이라고 표현한 것은, 아무래도 회사가 직원을 타 회사에서 데려 왔을때 단순히 그 직원이 개발한 소프트웨어를 마음대로 쓸 수 있다는 뜻 보다는, 향후 그 소프트웨어가 업그레이드 될 때의 이득과 직원의 기술력까지 완전히 다 얻을 수 있다는 의미로 쓰인 것 같습니다).

• 방치되기 쉬운 소프트웨어 프로젝트를 가진 소규모 회사는 가능하면 BSD 라이선스를 사용해야 합니다. 회사의 규모가 어떻든 모든 회사는 진정한 BSD 스타일의 오픈 소스 프로젝트
형태의 프로젝트를 통해 최소한의 법적 및 조직 오버헤드를 유지할 수 있어 서로에게 이득이 되므로 이런 방법에 대해 진지한 고려를 해야 합니다.

• 비영리 단체는 가능하면 오픈소스 프로젝트에 참여해야 합니다. 서로 다른 라이선스 하의 코드 혼용과 같은 소프트웨어 엔지니어링 문제를 최소화하려면 BSD 스타일의 라이선스를 권장해야
합니다. 특히 개발도상국과 교류하는 비영리 단체의 경우 GPL을 주의 깊게 살펴야 합니다. 법 적용에 많은 비용이 드는 일부 지역에서는 GPL에 비해 새로운 BSD 라이선스의 단순성이 상당한 이점이 될 수 있습니다.

12. □□

□□ □□ □□□ □□□□ □□□□ □□□□ □□ □□□ GPL□ □□, BSD □□□□□□ □□ □□□ □□ □□□□ □□□ □□ □□□□. □□□ □□□□□□ □□□ □□□ □□ BSD □□□ □□□□□□ □□□□□□ □□ □□□□ □□□ □ □□□□. □□ □□, BSD □□□□□□ □□ □□□ □□ □□□□□□ □□ □□□□□□ □□ □□□□.

□□, BSD □□□□□□ GPL□□ LGPL □□□□□□□ □□ □□□□□ □□□□□ □□ □□□, □□□□□ □□□ □□□□□ □□ □□□ □□□ □□ □□ □□ □□□ □□□ □□□□□ □ □□□ □□□ □ □□□□.

□□ □□

- [1] http://www.gnu.org/licenses/gpl.html
- [2] http://archives.cnn.com/2000/TECH/computing/03/28/cyberpatrol.mirrors/
- [3] Open Source: the Unauthorized White Papers, Donald K. Rosenberg, IDG Books, 2000. Quotes are from page 114, "Effects of the GNU GPL".
- [4] In the "What License to Use?" section of http://www.oreilly.com/catalog/opensources/book/brian.html

□ □□□ http://alumni.cse.ucsc.edu/~brucem/open_source_license.htm□□ □□□□ □□ □□□□ □□□ □□□□