

About the LEF Directors.

CSC사의 Leading Edge Forum (이하 LEF)은, 현재와 미래의 우리에게 영향을 미칠 산업의 현재와 미래의 방향을 연구하는 세계적인 사고적 리더쉽(thought leadership) 프로그램이다. CSC 혁신국 산하의 LEF는 기술혁신 지원과 보조금 제도, 응용연구, 혁신적 솔루션에 대한 포상, 그리고 각종 연구소들과의 제휴를 통해 신기술을 개발하고 있다. LEF는 기술시장의 동향과 모범사례(best practices)를 조사하여, CSC와 고객, 그리고 제휴 파트너간의 협력 및 혁신을 촉진시키고 있다.

기술경향을 주제로 연속 출간되고 있는 동 보고서에서, LEF는, 혁신이 현재 및 가까운 미래에 시장에서 어떤 역할을 할지에 주목하고 있다. 뿐만 아니라 기술의 현황과 미래를 연구하고 예기함으로써, 독자들에게 전술적 의사결정과 전략적 기획 간에 요구되는 균형을 형태를 보여준다.

LEF 경영진 소개

Paul Gustafson

Director, LEF, & Senior Partner

CSC Consulting Group

기술자로서 많은 업적을 쌓아온 Paul Gustafson은 신기술, 응용기술 연구 및 전략 분야의 세계적인 석학으로써, LEF의 현 소장이다. Paul Gustafson은 포럼의 프로그램을 주도적으로 구성하고, 연구 기술 아젠다의 방향을 제시한다. 기술경향들과 자신들 상호관련성, 그리고 비즈니스에 끼칠 영향들을 파악하는데 탁월한 능력을 지니고 있는 Paul Gustafson은, 자신의 이러한 통찰력을 활용하여 고객의 전략수립, CSC 연구활동, 그리고 리더쉽 개발 및 혁신전략 연구에 도움을 주고 있다. 그는 전략적 기술을 주제로 한 많은 논문과 저술을 발표했으며, 회사 경영진을 대상으로 빈번하게 강연을 하고 있다. pgustafs@csc.com

William Koff

Vice President, LEF

Bill Koff는 CSC내 기술인 모임을 주도적으로 이끌고 있다. 그는 CSC 내 각 사업부문의 기술직 임원들로 구성된 LEF 집행위원회 의장직을 맡고 있기도 하다. Bill Koff는 CSC내 연구활동, 혁신, 기술경향 주도 모임, 그리고 제휴 파트너사들을 주도 할 뿐만 아니라, CSC내 우수 센터 선정에도 주요한 역할을 수행하고 있다. 또한 그는 CSC 및 그 고객들을 대상으로 IT산업의 주요 추세와, 기술혁신, 그리고 첨단기술에 대한 전략적 투자 관련 컨설팅을 하고 있다.

인기 있는 초빙연사이기도 한 그는, 기술과 아키텍처, 그리고 관리 등의 주제로 강연활동을 하고 있으며, 그의 주요 관심사는 시스템 아키텍처, 디지털디스럽션(digital disruption) 및 공개소프트웨어 운동이다. wkoff@csc.com

Open Source : Open for business

목차

- 2. 리눅스의 비즈니스 진입
- 6. 보물 상자 : 기술 경향
- 6. 커뮤니티 문화 : 오픈소스의 정수, 커뮤니티
- 11. 스택 상위계층으로의 이동 : 리눅스가 오픈소스의 전부는 아니다!
- 23. 미션 크리티컬 : 오픈소스는 강력하다!
- 31. 달콤한 부분 : 오픈소스로 인한 비용 절감
- 44. 소프트웨어 혁명 : 오픈소스는 개발을 촉진하고, 새로운 아이디어를 길러낸다!
- 49. At Your Service : 서비스는 비즈니스의 새로운 기회를 창출한다.
- 55. 투명인간 : 오픈소스는 바로 우리 곁에...
- 60. 시장의 힘 : 오픈소스는 경쟁과 증진하고, 기존의 강자에게 도전한다.
- 65. 새로운 영역 : 오픈소스는 많은 분야에서 성장하고 있다.
- 69. 흥미 요소 : 오픈소스는 당신이 창조력이 넘치게 해줄 것이다

- 74. 법률 및 비즈니스 문제들
- 84. 시작하기
- 87. 부록 : 유용한 웹사이트
- 91. 감사의 말

Getting Down to Business

아래의 상황들로 인해 일대 지각변동이 일어나고 있다

- 소프트웨어는 무료라는 전제하에서 조직이 운영되고, 소프트웨어의 유료 구매에 대해서는 정당한 사유가 요구됨
- 구성원이 800,000 이상인 전 세계적인 소프트웨어 개발 커뮤니티가 주요 소프트웨어 벤더들에게 강력하게 도전
- 회사가 이전에 볼 수 없던 타임투마켓(Time-to-Market) 단축, 혁신, 그리고 품질 향상성취
- 범용 컴퓨팅 플랫폼들이 가격실행력 (price-performance)을 기반으로 막대한 이익을 점점 더 많은 회사에 제공하면서, 독점 솔루션들에게 도전
- 회사가 소프트웨어 개발 능력 향상을 위해 글로벌 개발 커뮤니티(Global Development Community)를 주목
- 각국 정부들이 독점 소프트웨어에 반하는 정책을 발표
- 소프트웨어 벤더들이 그들의 주장을 증명하도록 요구 받음

이 지각변동이 바로 “오픈소스” (Open Source) 이며, 리눅스 OS로 인해 유명해진 소프트웨어 개발 모델이다. 리눅스가 스타가 되면서, 오픈소스 성격이 강한 다양한 소프트웨어가 현재 나와 있으며, 이들은 웹서버, 애플리케이션 서버, DB, 콘텐츠 관리, 오피스시스템, 브라우저, 개발 툴, 보안 등 IT 전 분야에 걸쳐 있다. 오픈소스 소프트웨어의 출현과 함께, 수백만의 소프트웨어 개발자들이 협업을 위한 글로벌 커뮤니티로 모여 들면서, 오픈소스 소프트웨어는 독점 소프트웨어보다 크고 강력한 조직을 갖추게 된다.

각종 조직들은 중요한 사업 추진을 위해 의식적으로 오픈소스 소프트웨어를 채택하고 있다. 독일의 Borse 그룹, 독일은행, 덴마크 정부, 블루스코프 제철, NASA, 연합통신 (AP), JP Morgan Chase, 그리고 Google 등이 실재 사례가 될 수 있다. 뿐만 아니라 다수 국가 정부가 주도적으로 공개소프트웨어 관련 정책을 추진하고 있는데, 여기에는 브라질, 중국, 인도, 한국, 일본, 유럽, 호주, 미국 그리고 UN 등이 포함되어 있다. 이들 국가의 정부들은 오픈소스 정책과 대안 소프트웨어를 현재 검토하고 있다. 뿐만 아니라 IBM, Intel, 휴렛 팩커드, 오라클, SAP, 선 마이크로시스템, 델 등의 주요 IT 벤더들도 오픈소스를 지지하고 있다.

3P

오픈소스의 매력은 “무료”라는데 있다. 즉 누구든 오픈소스를 사용하고, 수정하고, 응용 제품을 창조하고, 그리고 재 배포할 수 있다. 따라서 라이선스 비용이 필요 없다. 오픈소스 소프트웨어의 개선, 변경 및 수정을 위한 글로벌 개발커뮤니티는 진입에 제한이 없으며, 종종 독점 소프트

웨어 벤더들보다 훨씬 빨리 작업이 이뤄지기도 한다. 당신은 유지보수나 성능향상을 위해 어떤 벤더에 의지할 필요가 없으며, 따라서 한 벤더의 제품에 대한 종속도 없다.

물론, 오픈소스가 만병통치약(silver bullet)은 아니다. 즉 오픈소스라는 이유만으로 태생적으로 좋은 소프트웨어일 수는 없다. 오픈소스 소프트웨어가 모든 상황에서 최선일 수는 없으며, 하룻밤 사이에 독점 소프트웨어를 대체할 수도 없다. 시장에는 수많은 우수한 독점 소프트웨어가 나와 있으며, 이들은 활용되어야 한다. 하지만, 소프트웨어 벤더들이 공개소프트웨어의 장점을 인식하고, 자신들의 제품과 서비스에 오픈소스를 채택하면서, 독점과 오픈 소스간의 경계가 희미해지고 있다는 점은 흥미롭다 하겠다.

결국, 회사는 자신들의 사업 상황에 가장 적합한 소프트웨어에 집중할 필요가 있으며, 이것이 바로 왜 오픈소스가 비즈니스 레이다 망에 존재할 필요가 있는지에 대한 이유이다. 오픈소스 소프트웨어는 매일매일 점점 더 강력한 IT 옵션으로 성장해왔다. CIO Magazine은 “2003년에 이 (오픈소스)혁명을 받아들이지 않는 CIO는 2004년에 값비싼 수업료를 지불하게 될 것이다” 라고 경고했다. Gartner는 2004까지 설치될 시스템의 30 % 정도에 대해 오픈소스가 고려될 것이라고 주장한다.

Open source for Open Business

아이디어와 혁신의 분출

오픈소스 운동의 핵심개념은 “자유로운 접근”(Open Access)과 “협력”(Collaboration)이라고 할 수 있다. 인터넷은 21세기의 오픈소스 운동과 아이디어의 자유로운 이동을 가능케 하는 핵심 매체로서의 기능을 수행하고 있다. 오픈소스는 아이디어의 생산과 개발에 관한 하나의 철학으로써 기존의 여러 가지 비즈니스 관행에 문제를 제기하고 있다. 즉, 어떻게 사업을 할 것인가 (독점되지 않는 연구개발), 지적재산권의 기본사항 (좀 더 공개적인 판권), 소프트웨어 개발의 성격 (밀실에서 개방된 공간으로, 독점에서 참여로) 등에 관한 것이다.

4P

“개방”과 “협력”을 속성으로 하는 오픈소스는 혁신을 증진시킨다. IBM의 John Wolpert는 Harvard Business Review지에 기고한 “혁신의 구속들로부터의 탈출”에서 아래와 같이 주장했다. “혁신은 개방적인 프로세스를 통해 촉진된다. 창조적인 아이디어는 자사의 비밀주의와 타협 없이, 외부 회사들과 자유롭게 접촉하고 그들의 통찰력과 역량, 그리고 지원을 이용할 수 있어야만 한다.” 회사에 구속에 얽매이지 않는 글로벌 개발 커뮤니티의 전문성을 공유함으로써, 오

오픈소스 운동은 기존의 기능을 혁신하고 개선시키며, 또는 새로운 기능을 창조할 수 있는 동력을 제공한다.

벤더들의 반응은 리눅스가 자사의 지적재산권을 침해했다고 주장하는 SCO 그룹의 소송에서부터, 오픈소스 소프트웨어와 부분적인 관련성이 있을 수도 있는 MS-Sun 합의, 그리고 벤더간 공동전선 형성에 대한 필요성 제기 등에서 감지되고 있다. Martin Fink는 그의 저서 “리눅스와 공개소프트웨어의 비즈니스와 경제학”에서, 소프트웨어 산업에 대한 오픈소스의 영향을 제약산업에서의 저네릭 드럭 (generic drugs, 주1)의 영향에 비유한다. 그는 이와 같은 범용화 (commoditization)와 가격 하락을 통해, 소비자의 접근은 용이해지고 벤더간 경쟁은 치열해진다고 강조했다.

구텐베르크의 인쇄술을 통해 부족한 인쇄물을 사람들이 공유할 수 있었던 것과 마찬가지로, 오픈소스를 통해 사람들은 부족한 소프트웨어 자원을 공유할 수 있게 된다. 공개적이고 협력적인 오픈소스의 접근방식을 통해 공정한 경쟁이 가능해지고, 누구든 소프트웨어 개발에 참여하고 또 대기업에 도전할 수 있게 된다. 오픈소스는 기술적인 문제일 뿐만 아니라 정치적이고 사회학적 운동이기도 하다.

5P

오픈 소스 : “Open for Business”를 통해 우리는 오픈소스 운동의 10가지 경향을 파악한다. 본 보고서는 기술, 미션 크리티컬 애플리케이션, 기회 활용, 그리고 법률 및 비즈니스 문제들을 상세히 다루고 난 뒤, 귀사의 오픈소스 활용에 관한 핵심적인 제안으로 마무리할 예정이다. 결론에서, 경향들은 일련의 핵심 비즈니스 기회들에 비춰 각각의 경향을 평가하는 매트릭스의 관점에서 서술된다. (84 페이지 참고)

핵심 비즈니스 기회들이 이 10가지 경향의 기반이 되며, 오픈소스 소프트웨어를 비즈니스 전략의 핵심으로 설정한다. 오픈소스 활용을 통해 우리는 제품 및 서비스의 타임투마켓(time-to-market)을 단축하고, (예: 독일은행) 긴급한 사업상 문제에 대한 새로운 해결책을 제시할 수 있으며, (예: 블루스코프 제철), 뿐만 아니라 표준화와 기술투명성을 통한 상호운영성을 확보할 수 있다. (예: 덴마크 재무성)

오픈소스 운동, 기술, 비즈니스 영향 등 오픈소스 관련 세부내용이 뒷 편에 계속된다.

6P

보물상자 : 기술 경향

오픈소스 운동은 제한된 그룹의 사람이나 제품에 한정되지 않기 때문에 그 폭이나 깊이가 상당하다고 할 수 있다. 따라서 기술과 성공사례로 충만한 보물상자로 불릴 만 하다. 이 장에서는 오픈소스 운동의 핵심을 이루고 있는 10가지 경향을 다루도록 한다.

- 커뮤니티 문화
- 스택 상위계층으로의 이동
- 미션 크리티컬 (사업의 핵심분야)
- 달콤한 혜택 (Sweet Spot)
- 소프트웨어 혁명
- At your Service
- 투명 인간
- 시장의 힘
- 새로운 영역
- 흥미 요소 (Fun Factor)

커뮤니티 문화 (커뮤니티: 오픈소스의 정수)

오픈소스 운동의 핵심은 커뮤니티이다. 커뮤니티는 소프트웨어의 영감을 제공하고, 개발하며, 보수하고 성능을 향상시키는 역할을 한다. 커뮤니티는 소프트웨어의 재사용을 통해 새로운 혁신을 이뤄내기도 한다. 커뮤니티 참가자들은 느슨한 구조의 메리트 시스템 속에서 전문지식을 공유하고 얻는다. 즉, 커뮤니티 문화는 이익이 아닌 참여에 기저를 두고 있다.

과연 무엇 때문에 사람들은 오픈소스 운동에 동참하려 하는가? 어떤 이유로 그들은 자신의 시간을 희생하려 하는가? 자신들의 회사에서 성공사례로 인정 받을 수 있는 어떤 것을 그들은 커뮤니티에서 배우고 있는가?

커뮤니티 문화

컴퓨터의 여명기랄 수 있는 1940년 이래 IT 분야에서의 커뮤니티는 지속적으로 존재해왔다. 당시의 소프트웨어는 주로 메인프레임에 사용되었고, 컴퓨터를 사용법을 알았던 대학 및 정부 지원 연구실의 프로그램 매니아들, 즉 소수의 엘리트 집단 사이에서 자유롭게 공유되었다. 오늘날, 인터넷의 네트워크 효과(Network Effect)로 인해 전 세계의 프로그래머들과 유저들은 커뮤니티에 깊숙이 관여할 수 있게 되었다.

왜 인간들은 커뮤니티에 관심을 가지게 되는 것일까? 회의론자들은 인간은 단지 금전에 의해서만 동기부여가 될 수 있다고 주장할 것이다. 하지만 이는 사실이 아니다. 적십자 (Red Cross)에

서 평화봉사단 (Peace Corps)에 이르는 저 수많은 자원봉사 기관들을 보라. 세계의 많은 이가 좀 더 나은 세상을 위한 자발적인 노력 덕분에 살아가고 있는데, IT 분야라고 이와 달라야 할 이유가 있는가? 오픈소스는 분명 다른 사회적 의미가 있지만, 지금의 IT 사회가 커뮤니티의 개념에 반대할 이유는 없다.

7P

강력한 독점 소프트웨어 산업이 존재하고 있긴 하지만, 역사적으로 봤을 때 소프트웨어 개발은 커뮤니티와 공유에 기반하여 이뤄져 왔다. 즉, 1969년, 4개 노드의 상호연결로 시작된 인터넷 자체가 공개적이고 협력적인 시도였다. 이후, 현 오픈소스 운동의 선구자적 역할을 한 많은 시도들이 있었는데, Richard Stallman의 GNU 프로젝트, MIT의 Athena 프로젝트와 공공도메인 소프트가 예가 될 수 있다. (Richard Stallman은 자유소프트웨어재단(Free Software Foundation, 이하 FSF)를 창립하고, 현재 가장 보편적으로 사용되고 있는 오픈소스 라이선스인 “General Public License” (이하 GPL)을 고안했다. (법률 및 비즈니스 문제 편 참고)

오픈소스 운동은 1990년대 리눅스의 출현으로 힘을 받기 시작했고, 특히 인터넷의 상용화로 추진에 탄력을 받게 된다. 2000년, Eric Raymond는 “Cathedral & Bazaar” 라는 제목의 논문을 통해 이 새로운 형태의 개발의 성격을 논한 것은 유명하다. 논문에서 그는 “즉, 독점개발은 마치 성당과 같아서 거대하고 폐쇄적이며, 느리고 심지어 끔찍스럽기 까지 한데 반해, 오픈소스 개발 형태는 마치 바자회 같아서 새로운 아이디어와 접근방식에 탄력적이며 개방적이고, 신속하며, 그리고 매우 독립적이다.” 라고 주장했다.

이 개발형태의 정수가 바로 커뮤니티이다. 일반적으로, 커뮤니티는 GNU의 Stallman과 리눅스의 Tovalds와 같은 최초코드를 개발한 한명의 천재 프로그래머에 의해 씨가 뿌려지며, 이후 커뮤니티가 만들어져 최초 프로그래머가 작업한 것을 개선해 나간다.

Raymond는 리눅스의 발전상황을 아래와 같이 묘사했다. “거의 처음부터, 리눅스는 인터넷을 통해 자발적으로 접속한 수많은 사람들에 의해 일상적으로 해킹되었으며, 리눅스의 가장 중요한 특징은 기술적인 면보다는 이러한 사회학적인 면에서 나타난다.” 그는 리눅스의 커뮤니티적 접근이야말로 리눅스의 핵심 성공요인이라고 주장했다.

커뮤니티 뜨다 (The Community Takes Off)

1990년대 후반, 역사상 최대의 오픈소스 프로젝트였던 리눅스 개발 주변의 활기 만큼이나, 네트워크 효과도 한창이었다. 오픈소스 개발 커뮤니티는 그때에서야 신뢰할 수 있는 집단으로 인정 받았으며, 이후 오늘날까지 계속해서 성장을 거듭해왔다.

커뮤니티 신뢰도의 한 예로 세계 최대의 오픈소스 개발 웹사이트인 SourceForge를 들 수 있다. 82,000건 이상의 프로젝트를 주관했고, 860,000명 이상의 개발자를 등록회원으로 보유한 SourceForge는 커뮤니티의 커뮤니티라 불린다. 커뮤니티들에게 신용을 대여함으로써, SourceForge는 이들 커뮤니티들이 프로젝트 호스팅, 버전관리, 버그 및 이슈 추적, 메일링 리스트, 이메일 아카이브, 프로젝트 관리 및 협력 리소스를 제공하는 세련된 사이트가 관리할 만한 가치가 있음을 보여준다. 이것은 방관의 프로세스(renege process) 관리를 잘 하고 있는 것이다.

8P

과연 이들 오픈소스 개발자들은 어떤 사람들인가? 2002년, 보스턴 컨설팅사가 SourceForge 사용자를 대상으로 한 조사에 따르면, 응답자는 평균 30세였고, 대략 11년간의 프로그래밍 경력을 가지고 있는 것으로 나타났다. SourceForge에는 대학생이나 직장 초년병들 뿐만 아니라 IT 전문가들도 다수가 참여하고 있었던 것이다. 이들 직장인 유저들을 포함한 전체 유저들의 주요 직종은 아래와 같다.

1. 프로그래머 (45 %)
2. 학생 (20%)
3. 학계 (7 %)
4. 시스템 관리자 (6 %) 또는 IT 관리자 (6 %)

개발자의 신뢰도에 대한 또 하나의 바로미터를 우리는 Gforge에서 찾을 수 있다. Gforge는 협업을 통한 오픈소스 개발 환경이다. 우리는 Gforge.org 사이트에서 특정기술을 가진 개발자를 서치할 수 있다. Gforge에서 개발자들은 아주 교묘한 방식으로 상호간 랭킹을 매기고 있다. 즉, 당신의 랭킹이 높을수록, 타인에게 부여한 당신의 랭킹이 더 많은 비중을 가지게 된다.

참가자 이외에, 커뮤니티의 신뢰도는 커뮤니티의 제품에도 영향을 받는다. 즉 구축한 작품의 기능, 업무수행 능력, 기술수준 및 사용 정도에 따라 커뮤니티의 신뢰도가 달라진다. CSC E-Business & Technology Center (독일 비스바덴 소재)의 솔루션 아키텍트인 Gabor Herr씨는 “Apache 소프트웨어가 너무나 우수하기 때문에 사람들이 그것의 오픈소스 커뮤니티에 참여한다” 라고 주장한다. 역으로, 폐쇄적이고 형편없는 제품의 커뮤니티에는 사람들이 모여들지 않을 것이다.

커뮤니티에 대한 신뢰는 개발자들을 오픈소스 프로젝트에 참가하게 하는 중요한 동기배이터 역할을 한다. 개발자들을 끄는 오픈소스의 주요 매력은 아래와 같다.

1. 기술적 난제 해결
2. 세계 최고의 인력들과의 교류
3. 타 커뮤니티들이 활용할 수 있는 성과 창출의 가능성
4. 커뮤니티의 두드러진 멤버로서의 기술 및 명성 (상품성) 제고
5. 오픈소스 소프트웨어에 대한 유료 서비스 제공의 가능성 (At Your Service 편 참고)

이외에, 개발자들은 자신의 영역을 넓히거나, 본업에서는 잘 다루지 않는 제품을 사용할 기회에도 역시 동기부여를 받는다. 예를 들어, 비디오 프로그래밍의 경우 개발자들은 단지 즐거움 때문에 참여하기도 한다. (Fun Factor 편 참조)

9P

부의 공유 (Share the Wealth)

오픈소스 커뮤니티에서 취득한 성공사례는 소속회사의 방법론(Methodologies)에 채택될 수 있다. 어떤 경로를 통해서건, 오픈소스 프로젝트에 참여한 개발자들은 오픈소스 방식의 장점을 공유하게 되고, 이를 통해 개발과 프로젝트 관리의 새로운 경지를 회사에 보여주게 된다.

이것은 실제로 CSC에서 일어나고 있는 일이다. 소규모의 한 팀이 오픈소스 프로젝트에 최초로 참가한 이후, 오픈소스의 방법론과 정신을 통합하여 “개발표준사례”(Standard Development Practice)를 제작하였다.

CSC E-Business & Technology Center (독일 비스바덴 소재)의 선임 아키텍트인 Stefan Hohn 씨는 “우리는 오픈소스 커뮤니티에서 익힌 것과 유사한 접근방식과 행태를 사용하게 되었습니다. 오픈소스 프로젝트에 참여하면서부터, 우리는 최고의 오픈소스 프로세스를 우리의 개발 방법론 (methodology)에 점차 포함시켜 나갔고, 이를 통해 우리의 경쟁력은 향상되었으며, 마침내 개발속도, 품질, 그리고 가격 면에서 해외 프로젝트들을 압도하게 된 겁니다.” 라고 설명한다.

2001년, CSC는 독일증권거래소(Deutsche Borse Group) 소속의 한 계열사에 설치할 시스템 제작 작업을 시작했다. CSC는 부분적으로 오픈소스 소프트웨어를 사용하여 비용을 절감해 달라는 요청을 받았다. CSC내 담당 팀은 시스템의 기본 인프라로 안정성이 증명된 오픈소스 컴포넌트를 사용했다. 오픈소스 소프트웨어 사용이 처음은 아니었지만, 해당 팀이 프로젝트 관련 기술지원을 커뮤니티에 전적으로 의존한 것은 이것이 최초의 일이었다. 이것은, 팀원들이 벤더에게 핫라인으로 전화 할 것이 아니라, 지원을 받을 줄 알아야 한다는 것을 의미했다. 팀원들은 인터넷을 뒤져서 정보를 구했고, 이전에 전혀 안면이 없던 사람에게 이메일을 보내고, 존재조차 몰랐던 온라인 포럼에 질문을 하고, 내용이 천차만별인 수많은 관련 문서들을 검토했다.

놀랍게도 이 작업들은 효과가 있었고, 이로 인해 이후 훨씬 더 큰 프로젝트를 수주하게 된다.
(Mission Critical 편 참조)

Hohn씨는 “그것은 기대했던 것보다 훨씬 우수했어요. 그러나 우리는, 자신들이 익숙해져 있던 것과는 완전히 다른 방법으로 문제를 해결하는 방법을 배워야 했지요.” 라고 회상한다.

커뮤니티로부터의 교훈 (Lessons from the Community)

아래의 네가지 접근방법이 오픈소스 세계에서 나타나 CSC 방법론에 편입되었다.

1. 공동 소스코드 관리
2. 협업을 통한 개발
3. 자동 리그레션 테스트 (regression test)
4. 신속한 개발 방법론

Hohn씨는 “오픈소스 세계와 처음 조우했을 때, 우리는 개발 커뮤니티가 가진 사회적 활력을 느끼며, 이들의 프로세스에서 배울 것이 많다는 것을 인식했습니다. 그래서 우리는 소스코드 관리를 위한 우리의 첫번째 협업툴을 채택했습니다.” 라고 말한다.

이것은 SourceForge가 사용하는 코드버전관리시스템인 CVS이다. CSC 영국의 GPES Unix Software Factory가 내부 유닉스 시스템 관리를 위해 CVS를 현재 활용하고 있다.

최초의 오픈소스 프로젝트의 성공에 고무된 독일증권거래소는 거래소 규정처리프로세스를 지원하는 리포팅 시스템 구축에 오픈소스를 더 큰 규모로 적용하기로 결정했다. 거래소는 새로이 구축된 시스템에는 소프트웨어 라이선스 비용이 들지 않을 것을 요구했다.(기존의 독점 DB는 제외). 이것은 오픈소스를 의미한 것이었다.

해당 프로젝트 팀은 CSC 사무실, 고객 사이트 (거래소), 그리고 협력업체 사무실 등에 분산되어 있었다. 개발자간 직접적인 접촉이 어려운 상황에서, 오픈소스 개발경험이 있는 CSC는 프로젝트 팀이 사용할 웹 기반의 협업환경인 CSC Toolbox를 개발했다. 첫 단계로, 프로젝트 팀은 CVS를 Toolbox에 통합시켜 소스코드에 대한 분산접속이 가능토록 했다.

10P

회사 종업원들은 오픈소스 프로젝트와 커뮤니티에 참여하고, 그리고 회사에 최고의 접근 방법

을 제공할 수 있다.

그러나, SourceForge의 모든 정보가 무료 제공인 반면, CSC의 프로젝트는 보안이 요구되었다. 프로젝트 팀은 암호와 인증을 Toolbox에 추가하여, 전송 중 소스가 훼손되지 않고, 단지 인가된 인력들만이 협업 환경에 접근할 수 있도록 했다.

프로젝트가 진행됨에 따라 더 많은 서비스들이 추가되었다. 문제 추적 툴(issue-tracking tool)인 Mantis도 그 중 하나인데, 이 툴을 통해서 클라이언트는 협업환경에 참여할 수 있다.

테스팅은 오픈소스의 개념이 채택된 또 하나의 분야이다. 분산개발 환경에서는 소프트웨어 버전 관리가 어려워질 수 있다. 오픈소스 세계에서는 이러한 문제를 유닛 테스트와 리그레션 테스트를 통해 해결하고 있는데, CSC는 이 방식을 도입했다. 이 방법을 도입함으로써, 프로젝트 팀은 현격한 품질 개선, 개발속도 향상, 그리고 프로젝트 조기 완성이라는 효과를 보았다.

그 후 동 프로젝트 팀은 다른 고객으로부터 더 큰 규모의 전략 프로젝트를 위한 개발 프로세스의 검토 의뢰를 받고, 오픈소스 방식과 유사한 좀 더 신속한 방법론을 제안했다. 고객사가 보유한 기존의 개발모델은 완만한 업무흐름(workflow)과 형편 없는 품질이라는 문제를 안고 있었다.

효과가 입증된 CSC 팀의 개발모델에 근거하여, 그 고객은 CSC의 추천을 받아들이고, 시간 보다는 품질을 중시하는 방향으로 자신의 모델을 변경시켰다. 그리고 품질 측정치를 경영진이 좀 더 명확히 이해할 수 있도록 했다. 시험 결과치는 개발자로부터 부장에 이르는 모두가 볼 수 있도록 공개하여, 누구든 진행상황을 감시, 감독할 수 있도록 하였다.

CSC팀은 자연스럽게 오픈소스 소프트웨어를 사용하여 고객의 새 개발환경을 지원하였다. Junit과 CruiseControl은 기술적 이정표였다. Junit은 오브젝트(object)와 같은 작은 단위의 코드에 사용되는 테스트 제작에 사용된다. CruiseControl은 테스트를 구축 및 운용하고, 관리 대쉬보드(management dashboard)를 만든다. 이것은 테스트가 완전 자동화되어야 하는 분산개발환경에서의 핵심 기능이다. 이 프로젝트를 위해, 70명의 개발자로 이루어진 팀이 수 주만에 대략 1,600번의 유닛 테스트를 제작했다.

이러한 경험을 바탕으로, CSC는 오픈소스의 영향을 받은 자신들만의 방법론을 더욱 더 정교하게 가다듬었고, 이후의 프로젝트에서 반복적으로 사용하게 된다.

교차발표관리시스템 (Cross Release Management System)도 여기에 포함이 되는데, 이것은 신속한 개발을 위한 노력의 하나로, 이를 통해 CSC는 해외의 경쟁자들을 압도할 수 있었다. 이 시스템은 본래 독일은행에서 사용하기 위해 개발한 애플리케이션이었다. 경쟁자들과 같은 가격

이었지만, CSC는 신기술과 월등한 품질을 제공하고, 조기에 개발을 완료하였으며, 뿐만 아니라 개발자들을 사이트에 파견하여 생길 수 있는 리스크를 줄여주었다.

이 성공을 통해, CSC는 최근 독일은행으로부터 은행의 핵심업무 관련 시스템 구축 프로젝트를 수주하여, 오픈소스 기반에서 신속개발 (agile development)을 사용하여 개발하게 되었다. 이러한 오픈소스 식의 신속한 접근방식은 매달 성과물을 내야 하는, 빡빡한 스케줄의 프로젝트에 적당하다.

CSC는 자신들의 오픈소스 경험으로부터 계속해서 학습을 하고 있다. CSC에서와 같이, 회사 직원들은 오픈소스 프로젝트와 커뮤니티에 참여 한 뒤, 회사에 최고의 문제 해결방식을 제공 할 수 있다. 이 접근방식들은 본질적으로 커뮤니티에 의존한다. 커뮤니티는 문제들, 특히 복잡한 문제를 해결하고, 업무수행을 극대화 할 수 있는 강력한 접근방법을 제공한다. 무엇보다도, 오픈소스 커뮤니티들은 현재 오픈소스 소프트웨어와 애플리케이션들을 소수 언어로 번역하여, 컴퓨팅을 개도국에 소개하는데 큰 도움이 주고 있다. 이러한 것, 즉 로컬라이제이션을 통한 컴퓨팅의 글로벌라이제이션은 약간 복잡적일 수 있는데, 오히려 이러한 것이 오픈소스 커뮤니티와 잘 어울린다고 할 수 있다.

11P

스택 상위 계층으로의 이동 (리눅스가 오픈소스의 전부는 아니다)

오픈소스 소프트웨어가 소프트웨어 스택에 광범위하게 유포됨에 따라, 개발자 커뮤니티는 바빠지게 되었다. 오픈소스는 최 하단인 운영체제 레벨의 리눅스에서부터 데이터베이스, 애플리케이션 서버, 개발 툴, 그리고 비즈니스 및 데스크탑 애플리케이션들에까지 영역을 확장하고 있다. 오픈소스는 단지 리눅스에 한정된 것이 아니라, 일련의 기능과 기회를 제공하는 다양한 기술들의 컬렉션이라고 할 수 있다.

오픈소스의 컬렉션이 무엇으로 구성되어 있는지를 이해하기 위해서, 우리는 Flashmap Systems의 분류법을 채용한 소프트웨어 스택을 연구할 것이다. (기업 인프라, 비즈니스 애플리케이션, COTS 애플리케이션). 우리는 시스템 인프라에 사용되는 오픈소스 소프트웨어의 풍요로운 포트폴리오를 보게 될 것이다. 상위 레벨의 비즈니스 애플리케이션이 드물긴 하지만, 오픈소스 소프트웨어가 일상의 비즈니스 프로세스에 점차 진입하면서, 상황이 좋아지고 있다.

주의사항 : 아웃소싱과 비즈니스간의 경계가 희미해지는 최근의 경향으로 인해, 한 조직에서 소프트웨어 스택의 시작과 끝이 어디인지는 명확하지가 않다. 우리는 Flashmap의 분류법을 사용하여 대부분의 조직에서 공통적이고, 오픈소스 컬렉션 중 명품을 발굴하는데 관련된, 핵심적인

스택에 초점을 맞출 것이다.

우리가 발견하게 될 것은 현존하는 오픈소스 소프트웨어의 단지 일 부분임을 밝힌다.

Enterprise Infrastructure

: 컴퓨터와 유저들을 정보의 흐름과 처리를 지원하는 소프트웨어와 연결시켜주는 물리적 하드웨어 (12 페이지 차트 참고)

플랫폼

: 모든 시스템 및 애플리케이션 소프트웨어의 실행 기반

오픈소스 소프트웨어 중 가장 널리 알려져 있는 리눅스는 서버 운영시스템으로써, MS의 Windows NT에 뒤 이어 세계 제 2위의 시장점유율을 기록하고 있다. IDC는 리눅스 플랫폼이 플랫폼 전체 평균 수익성장율의 4배 이상의 수익을 2007년까지 매년 올릴 것이라고 예상하고 있다.

리눅스 이외에도, 다수의 오픈소스 계열 운영시스템이 있는데, 이들은 모두 리눅스와 같이 유닉스 계열이다. BSD Unix (Berkeley Software Distribution)는 운영시스템으로써 오픈소스가 된 첫 번째 케이스이다. FreeBSD, Open BSD, 그리고 연구 및 생산 환경에 적합한 고급 운영시스템인 NetBSD 등 다수의 오픈소스 운영시스템이 BSD Unix를 기반으로 개발되었다. 이외에도 BSD Unix에 기반한 다수의 플랫폼들이 현재 개발 중에 있다. 독점소프트웨어 세계에서는 Sun이 Unix 계열의 주력상품으로 Solaris를 출시한다고 발표했다.

심지어, 오픈소스로 전문화된 운영시스템도 개발되었는데, 유닉스 계열 시스템에서 운용되는 행위 기반 운영시스템인 Robotic Operating System이 일례가 될 수 있다.

플랫폼의 다른 중요한 요소는 네트워크를 이용하여 얼마나 확장 시킬 수 있는가 이다. 수십, 수백, 수천의 이종 시스템을 연결시켜 대형의 가상 컴퓨터를 구축하는 Grid 컴퓨팅이 좋은 예라고 할 수 있다.

Globus 주도로, Grid 컴퓨팅을 위한 주요한 오픈소스 작업이 이뤄진 적이 있다. Globus 툴킷은 기본적인 기술로서, 그리드 컴퓨팅 수행을 위한 사실상의 표준으로 간주된다. 툴킷의 최신 버전인 Globus Toolkit Version 3 (GT3)은 오픈소스 방법론을 따른다. Open Grid Services Architecture (OGSA)에 기반을 두고 개발된 GT3는 IBM, Platform Computing 등 주요 벤더들이 제안하는 그리드 컴퓨팅 솔루션의 핵심 요소이다.

또 하나의 유명한 작업은 분산된 클러스터들을 중심으로 디자인 되었으며, 오픈소스 방식의 협업으로 개발된 Sun의 Grid Engine 프로젝트에서 수행되었다. Gridware를 확보함으로써, Sun은 그리드 컴퓨팅의 기본 구축블록인 “compute 군(farm)”을 용이하게 설치할 수 있게 되었다.

12P

리눅스는 고성능 컴퓨팅(High Performance Computing, HPC) 클러스터에게 극적인 영향을 끼치게 된다. 즉, 리눅스 시스템 사용으로 인한 비용절감에 의해 이전의 진입장벽이 낮아 졌고, 이로 인해 점점 더 많은 HPC 시스템들이 나타나게 되었다. 예를 들어, Linux Network 사는 세계에서 가장 강력한 리눅스 클러스터를 국립로렌스리버모어 연구소와 국립 로스알라모 연구소에 각각 구축하도록 선정되었다. 세계 최고의 클러스터 구축을 가능케 한 것이 바로 Linux Network사가 개발한 오픈소스 BIOS인 LinuxBIOS였다. LinuxBIOS를 통해 Linux Network사는 경상비를 절감하고 수천개의 노드로 이루어진 클러스터의 확장성을 개선할 수 있었다.

13P

오픈소스 고성능 컴퓨팅 클러스터는 연구소뿐만 아니라 산업계에서도 사용되고 있다. 한 대형 에너지 회사가 현재 원유발굴을 위한 탐사에 이 클러스터를 사용하고 있다. (Sweet Spot 편 참고)

기타 중요한 오픈소스 HPC 소프트웨어에는 Scyld Beowulf, OSCAR, 그리고 openMosix 등이 있다.

Middleware

: 클라이언트와 서버를 연결하는 마법의 접착제(magic glue)

인터넷과 그 다양한 프로토콜은 오픈소스 운동의 뿌리인 정부와 대학의 연구 프로젝트에서 탄생하였다. 따라서 TCP/IP는 물론 거의 모든 인터넷 프로토콜의 최초 참조구현 (reference implementation)은 오픈소스 소프트웨어 상태에서 진행되었으며, 따라서 BSD Unix의 일부분이 되었다.

1990년대에는, 암호 관련 소프트웨어에 대한 미 정부의 수출 제한 조치로 인해, 다수의 보안 툴들이 외국(미국 입장)에서 개발되었다. 이러한 오픈소스 솔루션 덕분에 사람들은 미 정부의 제한사항들을 우회할 수 있게 되었고, 나아가 혁신을 불러 일으켰다. 이 툴들이 범용화 되면서, 제한조치는 완화되었다.

예를 들어, 이전에 SSLeay로 알려졌던 OpenSSL은 Eric Young과 Tim Hudson 두 사람에 의해 만들어졌다. 이 두 호주사람은 애초에 네스케이프의 SSL/TLS 프로토콜의 무료 버전을 제공하기 위해 SSLeay 저작을 시작한 것이었다. OpenSSL은 미국 이외 지역에서 구현되었기 때문에, 인터넷 전송을 위한 완전한 암호 프로토콜로 사용될 수 있었다. (예: HTTPS). 오늘날, OpenSSL은 Apache등 다수의 오픈소스 제품의 암호 라이브러리이며, 미국을 포함한 전 세계 국가들에서 광범위하게 사용되고 있다.

또 하나의 중요한 보안 프로토콜로서 SSH가 있다. SSH는 이후 자신의 오픈소스 버전인 OpenSSH의 원형이 된다. 현재, OpenSSH는 유닉스와 Windows 시스템의 안전한 원격 관리의 사실상의 표준 역할을 하고 있다.

LDAP의 상용(Commercial-grade) 오픈소스 애플리케이션 군 및 개발 툴인 OpenLDAP는 안정성 있는 디렉토리 서비스를 제공한다.

메시지에 편향된 고전적인 성격의 미들웨어 군에, 다수의 오픈소스 소프트웨어가 나와 있다. 이들 중 많은 수가 자바 메시지 서비스 (JMS) API에 기반을 두고 있으며, 대표적인 소프트웨어로 JbossMQ, Open JMS, Open Source Message queue (OSMQ), 그리고 JORAM이 있다.

큐잉 시스템 (Queuing System)은 비동기적으로 애플리케이션들을 연결시켜준다. 반면, 애플리케이션들을 다른 환경의 플랫폼 상에서 동기적으로 연결시키기 위한 기술이 “Web Services”이다. MS의 .Net 프레임은 가장 우수한 제품들 중 하나이다. 하지만, Web Services를 사용하는 오픈소스 프레임과 라이브러리들이 급속하게 성장하고 있다.

DotGNU 프로젝트는 3개의 소 프로젝트로 구성되어 있다. Portable.Net은 무료의 이식가능한 소프트웨어로 .Net의 대체를 목표로 하고 있다. 웹기반의 다수 유저간 협업 세트로서 PhpGropWare는 다양한 Web Services 컴포넌트들을 제공하며, 이들 모두는 XML-RPC를 통해 접근이 가능하므로, 쉽게 자신들의 Web Service 애플리케이션에 통합시킬 수 있다. 마지막으로 DotGNU의 실행환경인 DGEE가 있다. DGEE는 DotGNU의 핵심적인 Web Service 컴포넌트와 웹서비스 요청의 수락, 인준, 그리고 만족시키는 기능을 제공한다.

아파치소프트웨어재단 (Apache Software Foundation)의 Web Services [Project@Apache](#) 역시 같은 맥락에서 추진되었다. 동 프로젝트는 몇 개의 소 프로젝트로 구성되었으며, 이들 중 가장 유명한 것이 Axis이다. Apache Axis는 World Wide Web 컨소시움에 대해 단순객체접근프로토콜 (SOAP, Simple Object Access Protocol)을 구현한 것이며, 자바 뿐만 아니라 C++에서도 가능하다. Axis에 근거하여, 많은 프로젝트가 발주되었고, WSIF (호출 프레임워크), addressing, 그리고 WSS4J (Web Services의 보안)와 같은 고도의 작업이 이행되었다.

Serverware

다수의 클라이언트가 접속하는 백엔드의, 멀티스레드 및 다중프로그램 컴퓨터인 서버에서 구동되는 소프트웨어

오픈소스 소프트웨어 중 운영시스템 다음으로 잘 알려진 종류의 소프트웨어는 서버소프트웨어이다. 영국의 인터넷 서비스 회사인 Netcraf의 2004년 8월 조사에 따르면, Apache 웹 서버는 전 세계 웹서버 시장의 68 %를 점유하고 있다. 따라서, Apache는 독점 및 오픈소스를 통틀어 가장 광범위하게 사용되는 웹서버임을 알 수 있다.

14P

Apache 프로젝트는 오픈소스 사상 두 번째로 큰 프로젝트로서 (리눅스가 첫번째), Apache 웹 서버를 포함한 다수의 프로젝트가 포함되어 있다. Apache는 1995년, 전미 슈퍼컴퓨팅 애플리케이션스 센터(National Center for Supercomputing Applications)가 제작한 HTTPD 웹서버에 패치를 원하던 사람들에 의해 시작되었다: 즉, 패치 웹서버가 등장한 것이다.

웹서버 프로젝트는 시간이 경과하면서 확대되어갔다. 1999년, Apache 소프트웨어 재단이 구성되어, Jakarta, PHP, Struts & Apache XML 등과 같은 오픈소스 계열의 주요한 프로젝트들을 주관했다.

Apache의 성공에도 불구하고, ACME http, Roxen, jigsaw, Amiga Apache Webserver, EMWAC HTTP Werver, TUX 등 같은 오픈소스 계열의 기타 웹서버가 많이 나와 있다.

웹서버의 단짝은 애플리케이션 서버이다. BEA Systems, IBM, Oracle, Sun 등 거대기업들의 텃바구니 사이에서도, 오픈소스 애플리케이션 서버들은 성공과 함께 확실한 자리매김을 해왔다. 최초로 주목을 끈 것은 Tomcat이었다. Tomcat은 Sun의 서브렛(servlet) 스펙의 표준을 구현토록 디자인 되었으며, 현재 그 사례구현(reference implementation)으로 간주되고 있다.

Jboss Application Server는 5백만회 이상의 다운로드를 기록한 주요 오픈소스 J2EE 애플리케이션 서버다. Jboss는 25 % 정도의 시장을 점유하며, Websphere (IBM)와 Weblogic (BEA)에 이어 3위를 기록하고 있다. Sun으로부터 공식적인 인가를 받진 못했지만, Jboss는 Sun의 J2EE-1.3의 스펙과 완전히 일치하고 있어, 거대 독점 소프트웨어 벤더들과 직접 경쟁하고 있다. 기타 오픈소스 애플리케이션 서버로는 Jonas와 Geronimo 등이 있다.

FTP 서버나, 팩스 서버, 및 메일 서버로 사용되는 제품이 수도 없이 계속해서 나오고 있는 현실

을 고려할 때, 오픈소스 소프트웨어로 제작된 모든 서버를 동 보고서에서 언급할 수는 없다. 하지만, 중요성을 인정 받고 있는 파일공유 서버 중 하나인 Samba는 짚고 넘어가야 할 것 같다. Samba는 유닉스와 윈도우 환경간의 차이를 매워 주는 오픈소스 서버 세트이다. 유닉스 플랫폼, 특히 리눅스 상에서, Samba는 SMB/CIF 등의 MS 네트워크 프로토콜을 이용하여 파일과 프린터에 대한 균일한 접근을 가능케 한다. 리눅스에서의 백오피스 마이그레이션을 고려할 때, Samba는 윈도우 기반 클라이언트들을 연결시키는데 사용되는 소프트웨어 통합 백본 들 중 하나이다.

15P

데이터베이스 서버들을 조사하는 것도 역시 중요하다. 지난 몇 년간 오픈소스 계열의 데이터베이스 서버가 많이 나왔지만, 대부분의 회사가 사용하는 것을 주저해왔다. 이러한 현상은 아마도 DB에 저장된 정보가 오픈소스 소프트웨어로 처리하기에는 너무나 중요한 것으로 인식되기 때문인 것 같다. 더구나, 회사들은 새로운 DB를 구축하기 보다는 기존 DB를 강화하기를 선호한다.

그럼에도 불구하고, Oracle, IBM, Sybase와 같은 거대 벤더들의 제품을 사용하지 않는 프로젝트들도 있다. 그들은 대신에 MySQL, MaxDB, PostgreSQL, 또는 Berkeley DB와 같은 오픈소스 제품을 활용한다. MySQL은 전 세계적으로 5백만회 이상 설치가 이뤄진, 오픈소스 계열에서 가장 성공한 관계형 데이터베이스로 꼽힌다. MySQL을 배포한 MySQL AB사는 무료 배포/사용 뿐만 아니라 독점적 사용도 지원하기 위해 GPL 및 상용(유료) 라이선스를 모두 채용한 이중 라이선스 (dual-license) 모델을 고안하여 소프트웨어를 제공한다. (Legal and Business Issues 편 참조). MySQL AB사는 라이선싱 수입 이외에, 서비스 및 트레이닝 제공을 통해서도 매출을 올리고 있다.

MaxDB는 독일의 Software AG사가 개발한 상용 데이터베이스관리시스템(DBMS)인 ADABAS D를 기반으로 하고 있다. SAP AG사는 ADABAS D의 판권을 자신들의 기업용 R/3 소프트웨어 세트의 대안으로 구매한 뒤, 개발 및 고객맞춤 (customization)을 계속했다. 전략적인 고려와, DB가 보편화되었다는 것을 보이기 위해, SAP은 자신들의 SAPDB를 GPL 하에서 출시하였으나, SAPDB에 대한 수요와 인기가 기대에 미치지 못하면서, MySQL은 SAPDB에 대한 교차 라이선스를 가지게 되었다. MySQL과 SAP간의 전략적 제휴는, SAP R/3와 같은 고가의 기업용 애플리케이션을 운용할 수 있는 오픈소스 계열의 기업용 데이터베이스의 공동 개발 및 영업을 위해 이뤄졌다.

PostgreSQL 역시 오픈소스 계열의 주요 데이터베이스이다. 애초에는 표준적인 기능을 갖추지 않은 연구 프로젝트로 시작했으나, 오픈소스 개발 커뮤니티에 인계된 후 다양한 기능을 갖춘 관

계형 SQL 데이터베이스로 태어났다.

Berkeley DB는 많은 무료 및 상용 제품 내부에 들어가는 가장 보편적인 오픈소스 소프트웨어 컴포넌트 중 하나이다. 다양한 기능을 갖춘 애플리케이션 데이터베이스 라이브러리로서, Berkeley는 DNS 도메인 네임 서버, 메일 서버, 네트워크 설비, 지식관리, 콘텐츠 관리 및 문서 관리를 위한 지속적인 스토리지를 제공한다.

지난 5년간, 포탈은 기업 인프라의 중요한 부분이 되었다. 오픈소스 커뮤니티에도 포탈 관련 활동이 눈에 띄게 많아졌다.

오픈소스에서 가장 많이 알려져 있는 포탈 프레임워크는 Apache JetSpeed이다. 프레임워크로서 안정되어 있고 그리고 상용 벤더들이 상용포탈의 서버 기반으로 사용하기도 했지만, JetSpeed는 필수적인 프레임워크 기능에만 충실하려고 한다. 스펙트럼 상에서 JetSpeed의 반대편에 eXo 플랫폼이 있다. EXo 플랫폼은 오픈소스 계열의 통합 애플리케이션 세트로서 다수의 종속적이고 느슨하게 결합된 서비스를 제공한다. 하위 계층 서비스에는 로깅, 지속성, XML 프로세싱 및 캐싱이, 중간계층에는 포트릿 컨테이너(portlet container)와 워크플로 서비스가, 그리고 상위계층 서비스에는 이-카머스 및 퍼블리싱 서비스가 있다. 따라서, Exo는 다양한 기능을 포괄하는 총괄적(full-functional) 접근방식을 취한다고 할 수 있다.

2003년 자바 포탈 스탠다드 (Java Portal Standard, JSR-168)가 발표된 이후, 포탈들은 품질, 특성, 그리고 표준화 면에서 많은 발전이 이뤄졌다. 발표된 기준과 맞추기 위해, JetSpeeds나 오픈소스 포탈인 Liferay와 같은 기존의 프로젝트에도 그 기준이 적용되었다.

이외에도 널리 알려진 오픈소스 포탈로서 PHP-Nuke가 있다. 오픈소스 커뮤니티의 작업 덕분에, PHP-Nuke에는 날씨, 전자상거래, 사진 갤러리, 채팅 및 비디오게임 등 개인화를 위해 사용할 수 있는 모듈이 500개 이상 개발되어 있다.

콘텐츠관리시스템 (CMS)은 포탈과 보완적인 관계이다. 저명한 오픈소스 CMS인 OpenCms는 HTML에 관한 지식 없이도, 세련된 웹사이트를 제작하고 관리할 수 있도록 지원한다. 유명한 MS 애플리케이션과 유사한 사용자 인터페이스를 가진 깔끔하고 단순한 편집기를 활용하여 사용자들은 콘텐츠를 용이하게 제작할 수 있다. CSC는 OpenCms에 두가지 기여를 했다; 온라인 상의 도움말을 독어에서 영어로 번역하고, 소프트웨어 수정을 통해 OpenCms가 Oracle 데이터베이스 지원을 확장할 수 있게 한 것.

마지막으로, 오픈소스 세계에 퍼져 있는 이종의 CMS로 wiki라는 솔루션이 있다. Wiki는 가상의 공유 공간으로서, 단순한 마크업 (mark-up) 언어와 브라우저를 사용하여 문서를 집단적으로 만들 수 있다. Wiki 기술의 특징적인 면으로 페이지 제작과 업데이트 과정의 속도를 들 수 있다. (wiki는 하와이 언어로 “아주 빠르다”를 의미함). 일반적으로 수정 (modification)은 검토 없이 즉시 받아들여진다. 우수한 협업 툴인 wiki는 관련 인력들이 흩어져 있는 분산환경의 프로젝트에서 특히 유용하다. 현재까지, 온라인 오픈소스 백과사전인 영어판 Wikipedia가 세계 최대의 wiki이며, 다음으로 독어판 Wikipedia가 있다. (New Domains 편 참조)

Manageware

: 운영시스템, 네트워크 장비, 서버웨어 제품들 및 애플리케이션들의 관리를 감시, 감독하는 수많은 제품들

기업과 시스템 관리는 Openview (HP), Tivoli (IBM), Patrol (BMC), 그리고 Unicenter (CA)와 같은 포괄적인 시스템을 가장 자주 필요로 하는 분야이다. 상기 독점 소프트웨어와 경쟁이 될 만한 오픈소스 계열의 통합 시스템 관리 소프트웨어 개발을 위한 프로젝트는 현재로서는 나타나지 않고 있다.

데이터센터에 저장된 정보의 통합과, 다수의 톨과, 서버, 그리고 관리해야 할 애플리케이션 간의 상호작용에 대한 표준 미비가 그 이유일 수도 있다. 지난 2003년 가을에서야, 약 40개 시스템 관리 솔루션 벤더들이 모여, 데이터센터간 상호운용성과 톨들간 통합을 용이하게 하기 위한 DCML (Data Center Mark-up Language) 기구를 결성했다. DCML은 콘텐츠의 HTML, 그리고 네트워킹에서의 IP의 역할을, 데이터센터에 대해서 자신들이 수행할 것이라고 홈페이지에서 선언했다. 즉, 데이터센터 환경과 관리정책을 설명하는데 벤더 중립적인 (vendor-neutral) 방식을 채용함으로써, 상호운용성을 제고하고 독점적 접근을 의미 없는 것으로 만들었다.

한번 표준이 정해지면, 독점 및 오픈소스 제품이 나타나고, 그리고 애플리케이션을 관리하기 위한 좀 더 표준화된 방식들을 고안한다. 그때까지, Covalent Technologies 같은 벤더들은 오픈소스 제품들을 억제하기 위한 상용 (유료) 웹애플리케이션 관리 솔루션을 제공할 것이다. Covalent Technology사는 리눅스, Apache, Tomcat, Jboss, 그리고 MySQL 등의 주요 오픈소스 제품들을 자사의 Covalent Enterprise Ready Server에서 지원토록 하고 있다.

서버운영(server administration) 기능을 제공하는 몇 안 되는 오픈소스 제품 중에 Webmin이 있다. Webmin은 유닉스 및 리눅스와 같은 유닉스 파생제품의 시스템 운영을 위한 웹기반 인터페이스이다. 우수한 서버용 front-end 운영 툴이긴 하지만, Webmin은 네트워크관리시스템(nms)의 대체품은 아니다.

특정분야를 겨냥한 오픈소스 계열의 시스템관리용 툴이 있는데, 이들은 현재의 관리표준 (management standard)인 SNMP와 JMX에 기반하고 있다.

비교적 오래된 표준인 SNMP는 많은 오픈소스 제품의 지원을 받고 있다. 이들 중 대부분이 라이브러리들로, 애플리케이션들이 SNMP를 통해 정보를 제공하게 하는 기능을 한다. 소수의 전단부 (front-end) 툴이 SNMP 데이터를 수집하고, 조직하고, 그리고 시각화하는데 사용되고 있다. Tcl/Tk 기반의 관리콘솔(management console)인 Scotty는 전체 네트워크 인프라에 걸쳐 SNMP에 대한 통합된 시각을 제공하는 한 예이다. 그리고 오픈소스 네트워크관리시스템 (NMS)인 OpenNMS가 있다. OpenNMS는 SNMP 기반의 기능을 제공하고 네트워크상에서 제공되는 서비스들을 모니터 할 수 있다.

자바 서버 애플리케이션에서, JMX는 제어기능(manageability)의 표준이 되었다. JMX는 다양한 오픈소스 제품들에서 사용이 되고 있다. 애플리케이션 쪽에서는, MX4J가 제어기능 (manageability)을 위한 프레임워크를 제공한다. 클라이언트 쪽에서는 MC4J가 상용제품 수준의 편리한 대쉬보드와 기타 다양한 기능을 갖춘 세련된 관리콘솔을 제공한다.

Manageware 관련하여 가장 중요한 것은, IT 인프라 상의 시스템과 애플리케이션들을 관리, 감독하고 평가하는데 오픈소스 계열의 수많은 툴들이 항상 사용 가능하다는 것이다. 그러나, 아직까지 포괄적이고 통합된 접근책을 제시하는 오픈소스 프로젝트는 한 건도 없었다. 각각의 툴들은 잘 알려져 있지 않은 상태이며, 따라서 관리 대쉬보드에 사용되기 위해서는 상당한 통합을 필요로 할 것이다. 시스템 관리 및 감시 분야의 시장이 점점 더 확대되면서, 이 분야에 대한 오픈소스 세계의 포괄적인 접근을 기대해본다.

17P

Clientware

: 클라이언트 또는 독립형(stand-alone) 컴퓨터에서 사용되는 소프트웨어

리눅스는 서버 플랫폼으로 인식되고 있다. 그렇다면, 데스크탑 컴퓨터에서의 리눅스는 어떤가? 리눅스가 데스크탑용 OS로 적합하다는 인식은 완만하게 확산되어 가고 있다. 클라이언트 웨어에서 중요한 것은 개발자나 관리자가 아닌, 보통 사람들이 이를 사용하고 있다는 것이다. PC, 워크스테이션, 휴대용 컴퓨터, PDA, 및 스마트폰 등의 클라이언트 측의 핵심 특징은 사용자 인터페이스 (user-interface)이다. 사용상의 편리함이 최선인 것이다. 이것은 특히, 일반 유저들은 리눅스에 쉽게 접근하기가 힘들기 때문에 중요하다. 왜냐하면, 리눅스는 애초부터 개발자에 의해, 개발자를 위해서 개발되었기 때문이다.

따라서, 오픈소스가 솔루션들은 MS 윈도우 데스크탑에서 일반적으로 구현되는 것과 경쟁하기는 힘들 것이라고 생각해왔다. 그러나 시대가 변하고 있다. 비 윈도우 계열의 데스크탑 OS인 X Window System (X)는 상업적 영향력을 끼친 최초의 오픈소스 프로젝트였다. X는 기본적인 구축 블록(Building block)으로, 이 블록 위에서 Solaris나 HP/UX와 같은 상용 OS가 그들의 사용자 인터페이스를 구축한다. X가 없었다면, KDE나 GNOME와 같은 오픈소스 소프트웨어는 제작되지 못했을 것이다.

X는 스크린 상에서 윈도우(window)의 이동과, 마우스와 키보드와의 상호작용과 같은 GUI (Graphic User Interface)의 기반을 제공한다. X 자체가 사용자 인터페이스를 제공하는 것은 아니다. 그것은 사용자 소프트웨어가 담당한다. 따라서 X 기반의 환경은 아주 다르게 보일 수 있다.

X11 (X의 최신 버전)에 기반한 KDE와 GNOME은 리눅스 기반 데스크탑에 사용되는 2개의 오픈소스 GUI이다. Trolltech의 Qt Toolkit에 기반한 KDE는 윈도우와 유사한 데스크탑 환경을 제공한다. MS는 KDE로부터 영감을 받아 데스크탑 스타일, 테마, 라운드 타입 윈도우, 그리고 투명 윈도우 등 많은 변화를 윈도우에 주었다.

18P

바퀴를 다시 한번 발명(시간낭비) 하고 있습니까? 아니면 오픈소스를 사용하고 있습니까?

스택의 핵심

최적의 인력과 개발툴을 활용한 효율적인 소프트웨어를 개발이야말로 IT 비용을 절감할 수 있는 핵심 비결이라고 할 수 있다. 사람들은 공개소프트웨어를 주로 인프라와 연관시키지만, 공개 소프트웨어는 개발 툴킷에서도 역시 중요하다. 종종, 인력축소를 개발비 절감의 유일한 방법으로 간주하기도 한다. 하지만, 적절한 도구로 스마트하게 일하는 것이 경비절감에 더욱 도움이 될 수 있다. 이것이 바로 오픈소스가 궁극적인 재활용 라이브러리로서 강점을 발휘할 수 있는 분야이다. 당신의 개발자들은 바퀴를 다시 발명하고 있습니까(시간 낭비)? 아니면 오픈소스를 사용하고 있습니까?

CSC LEF의 Paul Gustafson 소장은 “오픈소스는 궁극적으로 봤을 때 세계의 재활용 라이브러리이다” 라고 말한다. CSC는 자사의 Proactive Service Management(PSM) 솔루션 개발에서 전체 솔루션의 약 1/4 정도의 부분에 대해 오픈소스 소프트웨어를 활용함으로써 개발시간을 상당히 줄였다. 오픈소스 소프트웨어가 없었다면, CSC는 아마도 훨씬 비싸면서도 시간이 더 소요되는 솔루션을 사용했을 것이다. (Software Revolution 편 참조)

오픈소스의 역사는 개발자에 의한 개발자를 위한 소프트웨어다. 따라서 개발 툴킷은 오픈소스 언어, 툴, 통합환경, 그리고 컴포넌트에서 풍부하다. 오픈소스 개발 툴킷의 핵심요소는 소프트웨어 스택의 핵심을 형성하고, 그 위에서 애플리케이션들이 활기차게 운용될 수 있도록 하는 것이다.

오픈소스 언어들

여러가지 의미에서, C 언어는 오픈소스 프로젝트의 아버지와 같은 존재이다. C 언어는 GNU 프로젝트와 리눅스를 증진시켰다. GCC (GNU C Compiler)는 최초의 GNU 운영시스템을 위해 Richard Stallman이 제작한 편집기(compiler)이다. 당시의 다른 언어나 편집기와는 달리, GCC는 무료로 배포되고, 다수의 시스템에 이식이 가능했기 때문에, 개발자들은 GNU 프로젝트에 사용할 툴들을 개발할 수 있었다. 리눅스 커널의 최초 개발자인 Linus Torvalds는 개발을 전적으로 GNU의 툴과 GCC에 의존했다. Torvalds의 프로젝트에 기여한 많은 이가 다른 개발 프로젝트에서 GCC를 기반으로 사용했다. 이후에 나타난 언어들과, 이들과 같이 개발작업에 투입된 툴들의 많은 수가 이후 오픈소스가 되었다.

C와 여타 scripting 언어들에 영감을 받아, 또 하나의 무료 소프트웨어 언어인 Perl이 1987년 나타났다. Perl은 NASA 제트 추진 연구소 연구원인 Larry Wall에 의해 제작되었다. 출시 이후, Perl은 급속히 인기를 끌게 됐고, 시스템 관리자에게 아주 유용한 도구가 되었다. Perl은 인터넷의 여명기에, 인터넷 애플리케이션을 제작하는 최초의 언어가 되었다.

현재, 가장 탁월한 웹프로그래밍 언어는 PHP이다. PHP는 Ramus Lerdorf의 온라인 이력서를 모니터 하기 위해 만들어진 Perl 스크립트 세트로 시작되었다. 좀 더 많은 기능이 필요하다는 것을 알게 된 Lerdorf는 C 기반의 확장된 규모로 구현하여, 데이터베이스를 확장하고 PHP 개발자들이 간략하면서도 동적인 웹 애플리케이션을 제작할 수 있도록 했다. PHP5는 다양한 기능을 갖춘 웹 애플리케이션 개발 플랫폼으로써, 현재 인터넷 도메인의 20 % 를 넘는 수백만의 사이트가 PHP를 설치하고 있다. 포털, wikis, CMS, 그룹웨어 및 포럼 등 수많은 오픈소스 프로젝트들이 PHP를 사용하여 제작되고 있으며, 향후에도 이 추세는 계속될 것이다.

19P

자바는, 오픈소스는 아니지만, 언급할 만한 가치가 있다. 왜냐하면, 자바는 모든 플랫폼에서 사용할 수 있는 최초의 언어이기 때문이다. (애플리케이션이 일단 완성되면, 많은 다른 시스템에서 운용이 가능). C, Perl, 그리고 PHP를 제외하면, Java는 오픈소스 커뮤니티의 가장 많은 지지를 받고 있는, 그리고 가장 많은 오픈소스 프로젝트에서 사용 되는 프로그래밍 언어일 것이

다.

오픈소스 툴들

오랫동안, 개발자가 애플리케이션을 제작하고, 검사하고, 문서화하기까지 하는 통합개발환경 (이하 IDEs)은 오픈소스 세계에서는 희귀한 것이었다. 자바 개발자들에게 선택할 수 있는 수많은 IDEs가 주어졌고, 따라서 통합개발환경들은 종종, 단독적으로 사용되었다.

Sun의 NetBeans가 IDE 시장에 나타나서, 마침내, 2000년에 오픈소스가 되면서 상황이 달라졌다. (Market Force편 참조). 1년 뒤, IBM은 Eclipse라 불리는 신제품 IDE를 출시를 발표했다. Eclipse는 오픈소스로 출시되기 이전에 IBM 내부에서 많은 개발과정을 거쳤다. BEA Systems는 전 세계 개발자 커뮤니티에 대한 Beehive의 영향력을 활용하고, 개발을 지속시키기 위해, 자신의 IDEs가 플랫폼으로 사용하고 있는 BeeHive를 Apache 오픈소스 프로젝트로 출시하겠다고 발표했다.

선택할 수 있는 복수의 IDEs를 가진 개발자는 IDEs들을 통합해갈 것이다. 무료 배포로 인한 IDEs의 확산으로 인해, 개발자의 효율은 향상되고, 통합의 향상성 및 편리성이 제고되며, 그리고 거의 매일 새로운 기능들이 생겨난다. 개발자들은 툴을 사용하면서 개선시키며, 이 개선사항들을 타인들과 공유한다.

재활용과 보상

무료로 구할 수 있는 오픈소스로 무장한 개발자들은 꿈이 실현되는 것을 현재 목격하고 있다. 즉, 백지상태에서가 아니라, 툴킷의 요소들을 사용하여 애플리케이션들을 조립할 수가 있게 된 것이다.

언어와 IDEs뿐만 아니라, 툴킷에는 복잡한 프레임워크에서부터 다양한 기능성을 갖춘 라이브러리에 이르는 수많은 다양한 요소가 존재한다. Struts, Java Server Faces, Cocoon, Spring 및 Webworks와 같은 프레임워크는 프로젝트들의 시동을 걸고, 동일한 베이스라인을 제공한다. Junit 같은 유닛 테스트 라이브러리들은 코딩의 품질을 전체적으로 향상시켜준다. Hibernate와 Castor 같은 정교한 컴포넌트들 덕분에, 개발자들은 전문적인 기능들을 언제든지 자유롭게 사용할 수 있게 되었다.

숙달된 개발자에 의해 요소들이 조립될 때, 무엇이든 이뤄질 수 있다는 것은 하나의 경이로움이다. 다른 이들이 컴포넌트에 대한 생각과 함께, 실제로 구축하고 있을 때, 왜 시간낭비를 하고 있는가? 이런 꿈 같은 일이 일어날 수 있는 것은 코드의 사용과 연구, 개선 및 공유가 모두 무료

이기 때문이다.

적합한 인력과 적절한 조합의 툴을 사용하여, 소프트웨어 개발의 효율을 증진시킴으로써, 관리자는 원가 경쟁에서 승리하고, 개발자들도 환호한다.

20P

GNOME(GNU Network Object Model Environment)은 리눅스와 유닉스, 그리고 유닉스 유사 OS에서 유용하며, GNU 프로젝트의 공식 데스크탑이다. 첨단 외양과 느낌을 지닌 사용성(Usability) 덕분에, GNOME은 기존의 Unix 데스크탑인 CDE를 대체하면서, 월등한 기능을 보여주고 있다.

또 하나의 초현대적인 데스크탑 환경으로써, 차세대 데스크탑 환경의 원형으로 여겨지는 Sun의 Project Looking Glass가 있다. 이 프로젝트는 Sun 내부에서 진보적인 개발의 하나로 시작되었고, 커뮤니티로부터 의견을 수집하고, 신기술에 대한 사람들의 관심을 증진시키기 위해 최근 오픈소스로 출시되었다. 동 프로젝트는 3D 윈도우잉(Windowing)과 첨단 기술을 채택하고 있다. 앞으로 지켜봐야 할 흥미로운 프로젝트이다.

워드프로세서, 스프레드시트, 프리젠테이션, 그리고 개인정보관리 소프트웨어로 구성된 오피스 세트(office suite)는 데스크탑에서 가장 널리 쓰이는 소프트웨어로써, GUI의 꼭대기에 위치한다. OpenOffice는 현재 시장을 주도하고 있는 MS의 오피스 세트에 대한 오픈소스 계역의 대안 소프트웨어다. OpenOffice의 홈페이지에 따르면, OpenOffice는 1,600만회 이상의 다운로드와 셀 수 없을 정도로 많은 CD롬 설치로 세력을 키워가고 있다고 한다. 반대로, MS는 전 세계에 걸쳐 3억명의 사용자와 94%의 시장점유율을 주장하고 있다. (Market Force 편 참조).

OpenOffice는 자신을 세계적으로 주요한 오피스 세트를 만들어야 하는 커뮤니티로서의 미션을 가지고 있다. 이들이 만들어야 하는 오픈소스 오피스 세트는 모든 주요한 플랫폼에서 운용 가능하면서, 오픈소스 컴포넌트에 기반한 API를 통하여 모든 기능과 데이터를 사용할 수 있어야 한다. 이 소프트웨어의 주요한 특이점은 문서파일 형식으로 독점형식(format)이 아닌 압축 XML을 사용하는 것이다.

많은 회사들에게, 오피스 세트는 정보화 시대인 요즘, 여전히 가장 중요한 애플리케이션이다. 여타 오피스 소프트웨어와의 호환성문제가 OpenOffice와 StarOffice를 제작한 Sun 등의 경쟁자들이 해결해야 할 과제이다.

이외에도 오픈소스 프로젝트 관리 소프트웨어인 Open Workbench가 있다. Open Workbench는

유명한 MS의 Project와 기능면에서 유사하여, 리소스의 제한을 고려한 스케줄 제작 기능을 포함한 강력하고 다양한 프로젝트 스케줄링 기능을 제공한다.

하지만, 오피스 세트만으로는 데스크탑을 완벽하게 구성할 수 없다. 즉 오피스 세트 이외에 브라우저가 필수적이다. 여기서 유명한 브라우저 전쟁 스토리를 반복할 필요도 없이, 브라우저 시장은 Mozilla Firefox, Nautilus, 그리고 Konqueror와 같은 오픈소스 소프트웨어들로 가득 차 있다. Firefox가 이전의 Internet Explorer 사용자와 Netscape에 실망한 사용자들 사이에서 점점 지지를 얻고 있다. (Firefox는 오픈소스로 공개된 네스케이프 코드 기반이다, Market Force 편 참조).

인터넷과 네트워크가 핵심인 애플리케이션의 입구로서, 브라우저가 중요하지만, 이멜, 일정, 주소록, 작업목록, 메모 등 개인정보관리도 역시 중요하다. Mozilla 프로젝트에서 곧 출시되는 것이 바로 유명한 오픈소스 이메일 애플리케이션인 Thunderbird이다. PC World 지는 “이 놀라운 정도의 다양한 기능을 갖춘 오픈소스 프로그램을 곧 사용할 수 있게 되었다”면서 흥분했다. 아직 개발 초기단계이긴 하지만, Thunderbird는 침단이 스팸 메일 여과기를 갖춘 안정되고 유망한 이메일 플랫폼을 제공한다. 여기에 Mozilla Calendar를 추가하면, 우리는 훌륭한 개인정보관리 (이하 PIM) 환경을 가지게 될 것이다.

21P

MS Outlook은 사실상 PIM이다. 왜냐하면, 윈도우 플랫폼 상의 MS Exchange 메시징과 협력 소프트웨어를 함께 사용할 경우, PIM의 모든 기능을 제공하기 때문이다. 리눅스 플랫폼 관련하여, 오픈소스 PIM이 몇 가지 있다. MS Exchange에 연동 가능한 Novell의 ximian Evolution, phpGroupWare, SsKYRiX, KDE Kolab Client 등이 여기에 포함된다.

높은 기대를 받고 있는 오픈소스 계열의 PIM으로 Chandler가 있다. Chandler는 Lotus Development Corporation의 창업주이자, Electronic Frontier Foundation의 공동 창시자인 Mitch Kapor의 독창적인 산물이다. Kapor는 Chandler를 개발하기 위해 오픈소스 애플리케이션 재단 (Open Source Applications Foundation)을 결성했는데, 이는 이메일, 일정, 연락처, 작업 목록, 메시지 및 기타 정보를 좀 더 직관적인 방법으로 관리함으로써, 데스크탑과 우리의 디지털적 삶을 바꿔보려는 야심 찬 시도였다. 예를 들어, Chandler는 사용자가 여러 폴더나 이메일 인박스를 뒤지게 하지 않고, 문맥에 따라 정보를 구성하여, 프로젝트나 주제에 관련된 모든 정보를 함께 제공한다.

Chandler가 여전히 개발 중인 상태에서 (2004년 말 완성 예정), Kapor는 오픈소스에 과증하게 베풀고 있다. Chandler는 GPL 하에서 배포되겠지만, 상용 애플리케이션에 대해서는 유료 라이선스 방식으로 판매될 것이다. 전 세계의 개발자들이 새 콘텍스트를 개발하여 코드의 품질

을 향상시켜 주길 희망해 본다. Kapor에게 있어, 오픈소프 모델은 소프트웨어 벤처기업이 직면한 경제적 어려움을 고려할 때, 혁신적 애플리케이션을 개발할 수 있는 유일한 방법이다.

마지막으로 GIMP를 언급함으로써, 오픈소스 클라이언트웨어에 대한 토론을 완결하고자 한다. GIMP는 유명한 사진 및 그래픽 툴로써, Adobe 포토샵과 같은 주류 제품들과 경쟁하고 있다. GIMP (GNU Image Manipulation Program)은 사진 수정, 이미지 합성 및 저작 등의 작업에 사용된다.

비즈니스 정보

: 원시자료를 의사결정에 사용할 수 있는 정보로 바꿔 주는 소프트웨어

스택의 인프라 부분이 오픈소스로 채워지고 있지만, 상위 계층은 상황이 다르다. 비즈니스 정보는 오픈소스 소프트웨어가 도전해야 할 분야이다.

OLAP (온라인 분석 프로세싱)은 현재 가장 강력한 비즈니스 정보 기술이다. 이 분야에서 가장 탁월한 오픈소스 프로젝트는 2003년에 발표될 때, 최초로 양산 가능한 오픈소스 OLAP 서버로 광고된 Mondrian이다. OLAP를 사용함으로써, 우리는 대량의 데이터를 거의 실시간으로 분석할 수 있다. 기존의 데이터베이스 처리와는 대조적으로, OLAP는 특정 데이터 수정보다는 대량의 데이터 분석에 초점을 맞추고 있다. Mondrian은 SQL과 여타 데이터 소스들로부터 자료를 읽어 들어와서, 메모리 캐쉬에 데이터를 모은다. Mondrian은 조회언어로 MDX를 사용한다. Mondrian은 플랫폼에 독립적이며, MS의 .Net 플랫폼 버전을 제공하는 자바 기반 서버에서 사용된다.

OLAP 서버 이외에, 보고서 제작 기능이 필요하다. 이 기능은 오픈소스 Jpivot 프로젝트를 통해 확보할 수 있다. Jpivot을 활용하여, 유저들은 슬라이싱&다이스(slice & dice), 드릴업(drill-up), 및 롤업(roll-up)과 같은 OLAP의 기본 기능들을 수행할 수 있다. Mondrian과 Jpivot을 함께 활용함으로써, 오픈소스에서도 데이터웨어하우징이 가능해진다. 소규모의 데이터마이닝 애플리케이션에 사용된 Mondrian 및 Jpivot에 대한 을 제작하여 좋은 결과를 얻은 CSC는 금융서비스 관련 클라이언트에 사용할 기술검증(proof of concept)의 원형도 제작하였다. 하지만, 이것을 Hyperion, SAP, IBM, Oracle 또는 SAS와 같은 상용 제품들과 비교하는 것은 불공정하다. 이들 상용 제품들은 훨씬 다채로운 기능과 정교한 전단부(front-ends)를 제공할 뿐만 아니라, 비즈니스 아키텍처와 개발자에게 기술적 세부사항을 숨기기 때문이다.

COTS 애플리케이션

: 조직이 구축하기보다는 구매하는 패키지 애플리케이션으로, 산업간 수평적 애플리케이션들과 산업별 수직적 솔루션을 포함한다.

(번역 주: COTS는 제품으로서의 컴포넌트를 의미)

“off-the-shelf”와 “open source”는 일견 용어상의 모순으로 보이기도 한다. “off-the-shelf”가 가격이 붙여진, 완성되어 당장 사용할 수 있는 소프트웨어인 반면, 오픈소스는 무료로 배포되고, 또 자유롭게 변경이 가능한 소프트웨어이기 때문이다. 하지만, 오픈소스도 제품화 되고 있다는 점에서 상용제품이 될 수 있고, 보급 및 기술지원이 유료로 이뤄질 수도 있다.(At Your Service 편 참조). 그리고 off-the-shelf 소프트웨어도 변경(modify)에 제한을 두긴 하겠지만, 소스코드를 부분적으로 공개함으로써 오픈소스가 될 수 있다. 오늘날, 오픈소스가 다양한 툴을 제공하고 있긴 하지만, 비즈니스 프로세스와 Off-the-shelf 애플리케이션 분야에선 여전히 가야 할 길이 멀다. 하지만, 이 방향으로의 움직임이 있다.

이러한 움직임의 초기 징후가 시스템간 연결을 지원하는 메시지 시스템과 어댑터 프레임워크(adapter framework)가 포함된 전사적애플리케이션통합 (이하 EAI, Enterprise Application Integration)에서 나타나고 있다. 애플리케이션 통합과 e-business의 개선을 위해 사용되는 Openadaptor가 그 예라고 하겠다.(Market Force 편 참조). 다양한 소스로부터 기업의 민감한 데이터를 통합하여 의미 있는 비즈니스 정보로 변형시키는 ETL(Extract, Transform, 및 Load) 툴의 오픈소스 솔루션이 존재한다는 것은 놀라운 일이다.

이러한 오픈소스 계열의 역량에도 불구하고, 다수의 독점 EAI 플랫폼들이 훨씬 더 다양한 비즈니스 기능을 제공한다.(SAP 비즈니스 객체의 이해 또는, X.400 같은 채널프로토콜이나 EDI같은 포맷의 제공). 따라서 오픈소스는 현재로서는 약간의 빌딩블록만을 제공할 수 있을 뿐이며, 앞으로 강력한 비즈니스-레벨의 기능을 개발해 나가야 한다. 일반적으로 고도의 기능을 제공하는 EAI 시스템이 고가인 점은, 이 클래스의 소프트웨어가 범용화 되기 까지, 시간이 걸릴 것이라는 것을 암시한다.

Mission Critical 경향 편 에서 설명되겠지만, 많은 강력한 시스템들이 오픈소스 기반으로 핵심 비즈니스 기능을 제공하고 있다. 이러한 경향에 따라, 몇몇 전사적자원관리시스템 (이하 ERP, Enterprise Resource Planning)들이 오픈소스로 나타나기 시작했다. 그 중, ERP5는 트레이딩, 청구(invoicing), 회계, 제조, 공급망, 재고, 고객관리(CRM) 및 제품설계 기능을 갖추고 있다. ERP5 프로젝트는 소스코드의 공개뿐만 아니라, 모든 이가 ERP5 시스템을 도입하여 운용할 수 있도록 모든 필요한 법적, 사회적, 기술적 정보를 제공하여, 중소기업들도 업무 프로세스 혁신을 일으킬 수 있도록 하는 것을 목표로 하고 있다.

현재 제품화가 잘 진행되고 있는 ERP5, Compiere, 또는 GNU Enterprise 등의 오픈소스 ERP 또는 CRM 시스템을 연구하면서, 오픈소스 계열 기업솔루션 프로젝트 들이 점점 더 많은 분야에

서 강세를 보이고 있다는 놀라운 사실을 알게 되었다. 대부분의 프로젝트들이 아직은 포괄적이지 못하며, 또한 기업용 솔루션으로 고려하기에는 부족한 면이 있다고 할 수 도 있다. 반면, 이러한 오픈소스들은 분명 지속적으로 발전할 대안이기 때문에 중소기업들은 이들의 사용을 고려할 만한 가치가 있다. 이 대안 소프트웨어들은, 오픈소스가 현재 이동 중인 스택의 상위계층들을, 전 세계의 지칠 줄 모르는 오픈소스 커뮤니티가 제공하는 제안들로 채워감에 따라, 지속적으로 성장할 것이다.

22P

오픈소스를 활용한 개발의 차이점은 무엇인가?

오픈소스이든 아니든, 소프트웨어는 소프트웨어이다. 그렇다면, 독점소프트웨어와 비교하여, 공개소프트웨어를 사용한 실제 개발프로세스가 어떻게 다른가?

융통성

오픈소스 개발 환경에서, 개발자는 가능한 많은 소프트웨어 컴포넌트를 자유롭게 사용할 수 있다. 개발자들은 라이선스 비용이나 예산에 제한을 받지 않는다. 독점소프트웨어 개발의 경우, 예산 문제로 인해, 소수의 포괄적 컴포넌트를 사용하게 되어, 결국 중복이나 컴포넌트간 호환성 문제를 일으킨다.

종종 한 벤더의 제품을 사용할 것을 요구하는 독점 프로토콜에도 역시 제한 받지 않으며, 대신 최선의 선택을 할 수 있다.

개발자들은 역시 재정적 문제에도 제한 받지 않는다. 일반 개발환경에서, 개발자들은 비효율적인 컴포넌트라 하여 감히 내버리지는 못한다. 그것들을 구입하는데 소요된 투자금 때문이다. 오픈소스 소프트웨어 환경에서, 개발자들은 다양한 오픈소스 컴포넌트들을 테스트 한 뒤, 최선의 것을 선택할 수 있다.

기술지원

오픈소스에서의 기술지원은 광범위하고 유용하여, 실제 개발 프로젝트에서 많은 시간을 절감해 준다. 소스코드의 공개로 인해 신속한 문제해결과, 빨라진 성능개선 프로세스, 그리고 공동 습득이 가능하다. 코드샘플과, 토막지식, 그리고 튜토리얼은 개발자들이 오픈소스 컴포넌트를 이해하는데 도움을 준다.

기술지원 관련하여 또 하나 고려할 사항은, 독점소프트웨어가 비싸면 비쌀수록, 벤더업체를 제외하곤, 사용자 중 숙련된 전문가가 드물다는 것이다. 따라서, 벤더 다음의 2차적 지원을 받기 위해 개발자가 이용할 수 있는 커뮤니티가, 오픈소스에 비해 제한적이다.

혁신

오픈소스 소프트웨어는 기술혁신을 촉진시킨다. 왜냐하면, 오픈소스 환경에서는, 개발자들이, 유료일 경우 사용치 않았을 컴포넌트를 실험할 수 있기 때문이다. (예; 지속성 프레임워크, 리포트 툴, 애플리케이션 서버 등). 이것은 소프트웨어의 가격과는 상관없이 모두에 적용된다. 심지어 50 달러 짜리 소프트웨어 구매에도 사전 허가(approval)를 요구하는 회사가 있다.

재활용

소스코드 접근이 가능하기 때문에, 오픈소스 이용이 점점 늘어나고 있으며, 이로 인해 재활용이 증가하고, 소프트웨어 개발 프로세스에 종종 퍼져있던 NIH(Not-Invented-Here) 신드롬은 약화되고 있다.

(역자 주 : NIH Sysdrom : 외부의 연구나 기술을 충분한 검토 없이 거부하여 자사의 시간과 예산을 낭비하는 현상)

품질

개발자들이 컴포넌트를 처음부터 개발하고 테스트 할 필요 없이, 이미 개발되고 검증된 컴포넌트를 사용함에 따라 개발과정이 빨라지고 유연해졌다.

표준

일반적으로, 오픈소스 소프트웨어는 독점소프트웨어보다 표준에 더 충실하며, 이로 인해 상호운용성 면에서 더 뛰어나다. 오픈소스 개발 과정에서 새로운 표준이 형성되기도 한다. 새로운 표준들은 신속하게 오픈소스 프로젝트들에 도입되어, 개발자들이 첨단 소프트웨어를 스스로 제작하지 않고서도 적용해볼 수 있도록 한다. 신기술이 적절치 않을 경우, 낭비된 비용은 시간 정도이다. 궁극적으로 사람들은, 표준과 일치하고, 상호 교환 가능하다는, 더 안정된 것으로 인식되는 독점소프트웨어를 여전히 선택할 수도 있다. (예: 자바애플리케이션 서버).

라이선스

라이선스 문제는 오픈소스 소프트웨어에서 과소평가 되어서는 안 되는, 난해한 문제이다. 독점

소프트웨어의 라이선스에 관해서는 잘 알고 있는 회사들도, 오픈소스 소프트웨어 관련해선 이해도가 떨어진다. 오픈소스 소프트웨어는 무료 사용이 가능한 것으로만 종종 인식되며, 이것이 끝이다. 사실, 오픈소스 소프트웨어도 라이선스가 있으며, 다양한 종류의 오픈소스 라이선스가 있다. 회사들은 현재 사용 중인 오픈소스 라이선스의 규정들을 이해할 필요가 있다. (Legal & Business Issues 편 참조)

23P

미션 크리티컬 : 강력한 오픈소스

오픈소스 소프트웨어가 비즈니스의 핵심(mission-critical) 프로젝트에 동력원을 공급하면서, 소프트웨어 스택이 운영 준비를 하고 있다. 금융기관들에서, 정부기관 및 Google에 이르기까지, 오픈소스가 강력한 톨로 인식되고 있는 것은 명확하다. 아직 모든 오픈소스 소프트웨어에 해당되는 것은 아니지만, 오픈소스에게 비즈니스의 기회가 열려있는 것은 사실이다.

주요 분야에서 오픈소스 사용에 대한 주저함 또는 심지어 증오 또한 여전히 남아 있다. 오픈소스는 주요 프로젝트에서 사용하기에는 여전히 부족하거나 불안정하다고 주장하는 사람들이 있다. 즉, 오픈소스는 대규모 시스템에 적합하지 않고, 따라서 소기업이나 소수 매니아 들에게 적합하다는 것이다.

그러나, 주요 기관/회사들에 설치된 오픈소스들을 보면 이것이 사실이 아니라는 것을 알 수 있다. 오픈소스가 주요 프로젝트를 수주하고, 주요 임무를 수행하는 계층을 담당함에 따라, 현재 흐름이 바뀌어가고 있다. Linux, Apache, MySQL, Eclipse, Struts 등의 제품이 나와 있고, 기업들은 이들을 통합하여 원활히 업무를 수행하고 있다. 이들은 시험 프로젝트나 보조적인 기능들이 아니다. 오픈소스는 전면과 핵심에서 비즈니스를 운영하고 있다.

24P

비즈니스의 핵심에 선 오픈 소스

독일증권거래소 (Deutsche Borse Group, 메인 주 프랑크푸르트 소재)에서 오픈소스 소프트웨어는 광범위하게 사용되고 있다. CSC는 오픈소스 시스템 관련하여 독일증권거래소와 파트너 관계에 있다.

사업에 핵심적인 것을 포함한, 모든 인터넷 애플리케이션의 통합 아키텍처는, 상업용 CMS와 DB를 채택하기 했지만, 오픈소스 컴포넌트에 전적으로 기반하고 있다. 특히, Documentum과 Oracle 데이터베이스는 Jetspeed, Apache, 그리고 오픈소스 서치엔진인 Lucene으로 이루어진

포탈 플랫폼과 통합되어 있다. Lucene을 통하여, 사람들은 deutsche-boerse.com, 그리고 증권거래소의 자회사들인 www.clearstream.com, www.entory.com을 검색한다. 상기 오픈소스 컴포넌트들은 Tomcat 서브렛 엔진을 가진 Jboss 애플리케이션 서버에서 운용된다.

이외에도, 프랑크푸르트 독일증권거래소에 등록된 상장회사들이 의무적으로 제출하는 정규 공시 (연례보고서, 분기보고서 등)의 내용과, 제출시점을 관리하기 위한 보고시스템 또한 오픈소스 소프트웨어에 전적으로 기반하고 있다.

독일증권거래소 e-Business 담당 임원으로써 2003년까지 진행되었던 거래소 인터넷 시스템 재설치를 총괄했던 Rolf Barth씨는 “채택된 오픈소스 컴포넌트들은 최소의 비용으로 필요한 기능을 제공하여, 우리의 요구 사항들을 100% 충족시켰다”라고 말한다.

Rolf Barth씨는 또 “오픈소스 컴포넌트 관련 모든 질문과 요구는 프로젝트 팀이 처리할 수 있었다. 가끔씩, 프로젝트 팀이 문제에 대한 정확한 문서작업을 해야 했고, 또 상업용 소프트웨어라면 있었을 기술지원팀의 도움이 없어서, 스스로 버그를 해결해야 하는 경우도 있었다. 하지만 이러한 추가적인 노고가 들어간다고 해서, 상업용 소프트웨어가 요구하는 라이선스 및 유지보수 비용이 정당화되는 것은 아니다. 공식적으로 오픈된 이후, 포탈 플랫폼은 자체 개발팀에 의해 지속적으로 개선되어 왔다. 초기에 있었던 몇번의 경미한 문제 이후, 우리 포탈의 운영율은 99.5 % 에 달하고 있다.” 라고 말했다.

덴마크 재무성은 약 400개에 달하는 공공기관과 재무성간 회계정보 교환을 위해 오픈소스 제품으로만 제작된 데이터교환 시스템(data exchange system)을 운영하고 있다. 이 솔루션은 Linux를 돌리는 인텔 서버를 사용하여 초당 1.5 메가비트의 데이터를 전송할 수 있다.

CSC 덴마크의 eSolution 부문 책임자이며, 재무성 프로젝트의 솔루션 매니저였던 Hans Jayatissa씨는 “오픈소스 컴포넌트를 사용함으로써, 프로젝트 팀은 프로젝트를 전체적으로 단순화시키면서도, 고객의 요구사항을 정확히 충족시킬 수 있도록 설계된 솔루션을 제작할 수 있었다”고 말한다. 단독 벤더의 소프트웨어 세트가 필요 이상으로 기능이 많아서, 사용할 경우 부담되는 “all or nothing” 접근법인데 반해, 오픈소스 소프트웨어는 간결한 “mix & match” 접근이 가능하다.

25P

그림설명

호주 Port Kembla의 BlueScope 제철소 내 핫스트립밀 (Hot Strip Mill)에서, 공장을 운영하는 프로세스 컨트롤과 관리정보시스템의 주요한 부분은 오픈소스 소프트웨어로 구성되어 있다. 압

연기 (Mill)는 연간 약 250만 톤의 제강코일을 생산하고 있다. 그림에서, 가열된 열연 강판이 Finishing Mill을 통해 압연 처리 되기 이전 코일로 감겨지고 있다. BlueScope Steel은 CSC와 파트너쉽을 형성하여 공장 중단 사태가 발생하지 않도록 하기 위한 새로운 시스템을 설계 및 수행했다. 동 시스템은 매일 3 기가바이트에 달하는 프로세스 데이터를 기록하고 10개 이상의 컨트롤러 및 측정장비와 통신한다.

정부는 오픈소스 솔루션도 제안할 수 있는 CSC와 같은 대형 벤더가 있다는 것을 알고 안도했다.

Jayatissa씨는 “이것은 재무성이 서버환경에서 처음으로 사용한 오픈소스 소프트웨어였다. 이 프로젝트는 오픈소스 소프트웨어를 사용하려는 정부의 의지와 일치하는 것 같다”라고 언급했다.

정부는 오픈소스 솔루션도 제안 (그리고 수행) 할 수 있는 CSC와 같은 대형 벤더가 있다는 것을 알고 안도했다. 그때까지, 오픈소스 소프트웨어 솔루션은 단지 소규모 벤더들만이 제안하는 것처럼 보였다.

오픈소스를 채택함으로써, 재무성은 설치, 운영, 및 유지보수가 용이한, 단순하고, 신뢰할만하며, 그리고 확장가능한 솔루션을 가지게 되었다. 궁극적으로, 이것은 동 시스템을 보유하기 위한 총 비용의 절감을 의미한다. 재무성 솔루션은 제작을 위해 사용된 오픈소스 소프트웨어와는 명확히 구별되는 맞춤 코드이다. 따라서 재무성은 솔루션의 주인이며, 타 정부기관이나 벤더들에게 제공하거나 판매할 권리를 가진다.

동 시스템은 향후 재무성이 설치할 오픈소스 프로젝트의 맨 앞에 위치해 있다. 재무성의 데이터 교환시스템 책임자인 Peter Henningsen씨는 “데이터교환 솔루션은 좋은 출발이었으며, 덴마크 정부의 여러 분야에서 데이터 교환용 핵심 컴포넌트로 사용 될 수 있다”라고 말한다.

호주 뉴사우스웨일즈 주의 BlueScope 제철 (이전의 BHP 제철)은 Port Kembla의 Hot Strip Mill에 쓰일 실시간 프로세스 컨트롤과 정보관리시스템을 강화하기 위해 오픈소스 소프트웨어를 사용하고 있다. 일년 내내 쉬지 않고 돌아가는 시스템은, 고급 제강 제품의 스펙을 프로세스 컨트롤에 사용되는 하드웨어를 위한 지시문으로 번역하는 업무를 맡고 있다. 시스템이 2002년 론칭된 이래, 용광로는 최장 연속 운영 기록을 세우고 있다. CSC는 운영 첫해에 24시간 생산지원을 제공했다. 시스템은 오픈소스 개발 툴 이외에 Tomcat과 Apache를 사용했다.

오픈소스는 미 항공우주국(NASA)의 화성탐사선 (MER)의 주요 시스템에서도 충실히 임무를 수행하고 있다. 탐사선 Spirit과 Opportunity로부터 수신한 데이터를 분석하고, 그들의 일일 활동을

계획하기 위해 NASA가 사용하는 툴이 오픈소스 소프트웨어를 사용하여 개발되었다. “탐사활동 계획기(Science Activity Planner)” 툴은, 만일의 경우 탐사선의 하루 작업 전체가 위험해질 수 있기 때문에 핵심 툴로 간주 되고 있다.

26P

예산상의 어려움 하에서 가능한 많은 요구사항들을 충족시키기 위해, 개발팀은 오픈소스로 눈을 돌렸다. 프로젝트는 지금까지 성공적이다. 오픈소스 컴포넌트는 시스템의 민첩성, 고성능 및 안정성에 크게 기여했다.

다시 지구로 돌아와, 미 해군은 자신들이 관리하는 광범위한 장비와 부품들에 발생할 수 있는 공학상의 변화를 조정하고, 기록하며, 그리고 추적하기 위해 Zope Corporation의 오픈소스 계열 CMS를 사용하고 있다. 이것은 결코 소규모 작업이 아니다. 변경사항이 제트엔진이나, 배, 또는 다른 복잡한 장비에 생겼을 경우, 엄청난 파급효과가 있을 수 있기 때문에, 모든 변경사항은 치밀하게 모니터 되고, 기록되어야 한다.

핵심업무 수행에 사용된 오픈소스

오픈소스는 주요업무 수행을 요하는 애플리케이션에서 중요한 역할을 담당한다. 오픈소스는 확장성, 작업처리량, 신뢰성 및 제어능력을 제공한다. 전 세계 600여 개에 달하는 제휴 신문 방송 사들에게 웹 콘텐츠를 제공하는 연합통신(AP)의 AP Hosted News는 대규모 서비스 업체다. AP는 오픈소스 MySQL 데이터베이스를 사용하여 AP Hosted News를 운영하고 있으며, 하루 수십만 건의 작업을 수행하고 있다.

그림설명 1

“Downcoiler Pulpit”라고 불리는 통제실에서, 운영자는 제철소의 권치기(downcoiler)와 권치기 다음 지역을 내다 본다. 그의 전면부에 위치한 모니터가 그의 프로세스 스크린이며, 머리 위에 위치한 비디오 모니터가 제철소 전체의 중요 사이트 화면을 보여준다. 여기서, 운영자는 오픈소스 기반의 시스템을 사용하여 최종 코일제품의 품질 모니터와, 검사결과 기록, 그리고 불량정보 기록 작업을 수행한다.

그림설명 2

화성 탐사선으로부터 수신한 자료의 분석에 NASA가 사용하는 핵심 툴인 “탐사활동계획기”(Science Activity Planner)는 오픈소스를 활용하여 개발되었다.

MySQL AB 홈페이지 자료

제휴 신문사들은 평균적인 뉴스 사이클 동안 매일 십오만 페이지 이상의 콘텐츠를 자신의 웹사이트로부터 AP Hosted News로 올리며, 뉴스가 많은 날에는 그 수치가 제휴언론사 당 세배 이상인 5십만 페이지 정도까지 늘어나기도 한다. 제휴 언론사들이 독자들을 계속 보유하기 위해 AP의 기술과 콘텐츠에 과중하게 의존하기 때문에, AP Hosted News 애플리케이션의 기반 데이터베이스는 최대 11,000명에 달하는 동시 접속자들을 지원하기 위해 최적의 작업 수행 능력을 유지해야 한다.

대규모 사용자 베이스에서 오픈소스를 고려하고 있는 또 하나의 애플리케이션은 호주세무서(ATO)이다. 2004년 2월, 수백만의 납세자에게 서비스를 제공하는 ATO는, 처음으로 오픈소스 소프트웨어를 채택하여, 자신들의 MS 환경을 리눅스와 같은 대안 소프트웨어에 개방했다. UN 뿐만 아니라 중국, 한국, 일본, 유럽, 호주 및 미국 등의 정부가 오픈소스 정책 및 대안을 고려하면서, 정부주도의 오픈소스 관련 프로그램이 다수 진행되고 있다.

벨기에에서는 오픈소스 툴이 자국의 새 전자증명카드 뿐만 아니라 경찰 시스템에도 설치되었다. 기술적 장점을 지닌 오픈소스 소프트웨어를 활용하여, CSC는 두 프로젝트 모두에서 핵심적인 역할을 수행했다. 천 만 명의 벨기에 국민들에게 발급된 신분증(ID cards) 프로젝트에 쓰인 툴에 관련한 논평을 하면서, CSC 브뤼셀의 보안솔루션 그룹 매니저인 Marc Stern씨는 “오픈소스 컴포넌트를 사용하는 것 만이 주어진 시간 내에, 요구되는 수준의 품질을 제공할 수 있는 유일한 방법이었다. 우리는, 커뮤니티의 도움으로, 몇 가지 기능을 추가하거나, 제거함으로써, 오픈소스 소프트웨어 개발에 적극적으로 참여했다. 신분증 소프트웨어는 벨기에 국민이 사용하는 거의 모든 데스크탑 컴퓨터에서 가동될 수 있어야만 했고, 이러한 적극적인 개발노력은 ID카드 시스템이 대부분의 플랫폼에서 부드럽게 운용될 수 있도록 하는데 큰 도움이 되었다.” 라고 말했다.

웹서치 엔진인 Google은 전형적인 확장 가능한 오픈소스 프로젝트이다. 동 시스템은 전 세계 12개 이상의 데이터 센터에 펼쳐져 있는 100,000개 이상의 리눅스 컴퓨터에 확산되어 있는 것으로 알려져, 매일 평균 약 2억회의 검색을 수행하는 것으로 알려져 있다.

초고속의 빠른 처리와 높은 신뢰성에 사업의 성공이 달려 있는 금융서비스 회사들이 오픈소스를 주목하기 시작했다. JPMorgan Chase는 비용을 절감하고, 고도의 사내 IT서비스 및 융통성을 제공하기 위해 그리드 컴퓨팅과 리눅스를 설치했다. 리눅스를 운용하는 700개 이상의 CPU

로 구성된 자사의 Compute Backbone은, 이전에 7개의 분리된 시스템에 흩어져 있던 컴퓨팅 리소스의 공유 풀을 관리한다. 이들 핵심 비즈니스 시스템은 사내 거래자들이 이자율, 주식, 외환 및 신용 파생상품을 평가하고 관리하는 것을 도와준다.

이외에도 월 스트리트의 몇몇 회사들이 그들의 데이터 센터에 리눅스를 사용하고 있다. 이들 중에 credit Suisse First Boston Corp.이 포함되어 있는데, 보도에 따르면 이 회사는 매일 약 6천만 건 이상의 금융거래를 처리하고 있다고 한다. 몇 년전, Reuter는 자사의 Reuter Market Data System을 리눅스로 포팅(port)했다. 실시간 시장정보와 금융뉴스를 제공하는 이 시스템은 주요 중계회사들이 이용한다.

금융권 외에, 거래량이 많은 환경으로 정부 선거가 있다. 2002년 4,800만 독일 국민이 의회선거를 했을 때, 임시 공식결과를 계산하는 시스템에 오픈소스 소프트웨어를 사용했다 (MySQL, Tomcat, Jboss, 등). 이 선거시스템은 8만개의 선거 개표소에서 들어온 자료를 처리하여, 실시간으로 그 결과를 분석 및 안내하였다. 완전한 분석작업은 다음 날 새벽 3시에 완결되었다.

맞춤에서 일반 비즈니스 애플리케이션까지 (From Custom To General Business Applications)

대기업을 위한 맞춤형 핵심 애플리케이션 외에도, 오픈소스는 중소기업이 사용할 수 있는 보급형 핵심 비즈니스 애플리케이션 시장에도 진출하고 있다. 오픈소스 ERP로서, 2백만불에서 2억불 사이의 매출을 올리는 회사를 목표로 잡고 있는 Compiere가 좋은 예라 하겠다. Compiere의 유저에는 미국의 자동차 부품 제조업체, 독일의 타이어 소매업자, 브라질의 금속 부품 제조업자, 그리고 싱가포르와 중국의 케이블 제조업체가 포함되어 있다.

인기 상승의 증거로써, Compiere는 630,000회 이상의 다운로드를 기록하고 있으며, 현재까지 가장 인기 있는 오픈소스 계열의 비즈니스 애플리케이션이 되었다. 2004년 2월, Compiere는 이달의 SourceForge 프로젝트로 선정되었고, SourceForge에서 가장 액티브한 상위 10개 프로젝트 (top-ten project)에 포함되었다. 현재까지의 진행상황을 고려할 때, Compiere는 중간 수준의 시장에서, 당분간 인기를 유지할 것으로 예상된다.

28P

보안: 비밀은 해결책이 아니다

오픈소스 커뮤니티는 자신의 코드를 검토하는 수많은 사람들 덕분에, 독점 벤더보다 우월한 위치에서 안전한 코드를 제공할 수 있다. Jason Arnold / Program Manager, CSC H.E.A.T

security product

일견, 오픈소스와 보안은 모순된 것으로 보이기도 하지만, 사실 이들은 매우 잘 어울리는 한 쌍이다. 즉, 소프트웨어의 공개성 이야말로 가혹한 검토와 테스트를 보장하고, 보안을 굳건하게 한다.

오픈소스 소프트웨어 사용에 관한 주요 걱정들 중 하나가 바로 보안에 허술하다는 것이다. 핵심 시스템에 사용된 보안 컴포넌트의 소스코드를 공개한다는 것은 시스템 운영 책임자에게 결코 유쾌한 생각은 아닐 것이다.

이러한 생각은 합리적으로 들린다. 악한이 소스코드에 접근할 수 있다면, 그가 쉽게 약점을 찾아내고, 또 시스템을 훼손시킬 수 있다. 설상가상으로, 만일 그가 코드를 변경시켜, 변경된 사항을 소프트웨어의 공식적 보급판에 성공적으로 저장시킬 수 있다면, 그는 쉽게 백도어를 삽입하고, 그것을 이용하여 전체 시스템에 “트로이 목마”를 밀어 넣을 수 있을 것이다. 이것은 아래의 추론으로 연결된다: 소스가 비밀로 유지되는 소프트웨어가 더 안전하다. 왜냐하면, 그 소스코드는 계속 비밀로 유지가 되고, 단지 소수의 신뢰 받는 프로그래머들만이 접근가능하기 때문이다. 이외에도, 악의적 행동과 책임감도 문제다. 만일 누구도 코드나, 오픈소스 커뮤니티의 문제 해결에 책임을 지지 않는다면, 누가 보안 문제를 해결할 것인가? 이것은 시스템 운영자에게 있어 악몽과 같은 것이다.

군중 속의 안전

이러한 걱정들은 근거가 없는 것으로 밝혀졌다. 사실, 군중 속에 안전이 존재한다는 것이 드러났다. 소스코드의 공개로 인해, 대규모 커뮤니티의 개발자들이 한 시스템을 검사/검토하여 보안상의 결점을 찾아낸다. 사실, 시큐리티 라이브러리, 암호 알고리즘, 인증 및 인가 시스템과 같은 핵심 보안 컴포넌트들은 많은 개발자들을 끌어들이며, 코드를 보고, 검토하고, 개선하게끔 하여, 궁극적으로 보안공격에 대한 방어력을 증진하고 성능을 개선시킨다.

개발자들이 버그를 찾아 내도록 하기 위해 금전적 보상이 제안되는 경우도 있다. 브라우저 상의 중요한 결점이 몇 개 발견된 이후, Mozilla 재단은 Mozilla Firefox 브라우저를 포함한 자사의 오픈소스 소프트웨어에서 보안전문가가 의해 심각한 버그가 발견할 때마다 500불을 내놓기도 했다.

개발자들은 또한 명성에 동기 부여 되기도 한다. 복잡한 시스템에서 보안상의 약점을 집어 낸다는 것은 동료들에게 인정 받고 찬사를 들을 수 있는 힘든 작업이다. OS나 선거용 소프트웨어 같이 대중의 많은 관심을 받거나, 광범위하게 사용되는 애플리케이션에서, 개발자들은 시스템의

보안을 확인해보려는 유혹을 더욱 느낀다. 왜냐하면, 보안상의 약점에 의한 영향이 훨씬 크고, 따라서 그것을 발견하여 보수한 데 따른 찬사 또한 크기 때문이다.

소스코드가 공개되어 있을 경우 그 시스템이 가지는 보안상의 약점을 찾는 것은 더욱 어렵다. 하지만, 그 약점에 근거로 익스플로이트(exploit)를 만들기는 더욱 어렵다. 버퍼 오버플로우(Buffer Overflow, BOF)는 C나 C++로 제작된 서버 소프트웨어에서 발생하는 일상적인 보안문제로, 과거 많은 익스플로이트(exploits)의 원인이었다. BOF는 버퍼속으로 읽혀질 데이터의 양을 검사하지 않아서 생기는 시스템 콜에서, 정적 버퍼(static buffer)를 부주의하게 사용하여 생긴다. 만일, 악의를 가진 사용자가 프로그램이 버퍼에 덧씌우기를 함으로써 특별히 준비된 데이터를 버퍼에 할당된 메모리 공간 외부에 위치시키게 되면, 그는 어떤 코드든 쉽게 실행시킬 수 있게 되고, 따라서 시스템을 훼손시키게 된다. BOF를 일으킬 수 있는 코드는 어떤 일정한 패턴을 따르기 때문에, 그 코드를 찾는 것은 비교적 쉽다. 반면, 이것을 익스플로이트(exploit)로 사용하려면, 특정 입력 데이터를 준비해야 한다. 하지만 이것은 실행하기가 상당히 어려운 작업이다.

29P

전 세계에 흩어져 있는 대규모의 개발 커뮤니티 덕분에 보안 관련 보수의 회전시간이 독점 소프트웨어에 비해 아주 빠르다. 인터넷에 공개된 오픈소스 제품이, 보안에 관련된 문제가 발견된 지 단 몇 시간 만에 해결책이 나온 것을 찾는 것은 결코 어려운 일이 아니다. 더구나, Red Hat과 Novelle/SUSE와 같은 대규모 오픈소스 공급자들은 자신들이 제공하는 모든 오픈소스 컴포넌트의 보안문제에 대한 해결책을, 한 곳에서 수집하고 공개하는 매카니즘을 고안했다.

비밀이 꼭 안전한 것은 아니다

독점 소프트웨어의 보안이 우수하지는 않다 라는 것을 보여주는 역사적 증거가 있다. 유명한 사례로서 음성 암호 알고리즘인 GSM을 들 수 있다. 세계에서 가장 광범위하게 사용되는 핸드폰 시스템인 GSM은 비밀리에 개발되었으나 (독점소스), 역분석공학에 의해 공개되었다.

또 하나의 사례로서 미국 오하이오에 소재한 Diebold Election Systems사의 전자투표시스템을 들 수 있다. Diebold Election Systems사는 미국 전자투표시스템 산업의 선도기업이다. 2003년 1월, 시스템의 소스코드가 파일공유 서버(FTP)로부터 부주의하게 대중에게 공개되었다. 이후, 볼티모어의 존스홉킨스 대학 연구팀이 코드를 분석했고, 곧 몇 개의 치명적인 보안상의 허점을 발견하게 된다. 허점 들 중에는 악의적인 투표자가 투표를 여러 번 할 수 있다는 것도 포함되어 있었다. Diebold사는 모든 국민들이 선거절차에 따랐다면, 이러한 허점은 예방되었을 것이라고 응답했다. 상기 예를 통해서, 우리는 소프트웨어 자체로는 결코 속임수를 막을 수 없다는 것과,

소스코드가 비밀로 유지될 경우, 우리는 그 수정에 대해 소프트웨어 벤더만 믿어야 한다는 것을 알게 되었다.

오픈소스 : 현명한 엔지니어링

컴퓨터 보안 전문가인 Bruce Schneier씨는 “우수한 보안을 위해 오픈소스가 필요하다고 생각합니다. 공개적 보안은 항상 독점적 보안보다 항상 더 안전합니다. 우리에게 있어, 오픈소스는 단순한 비즈니스 모델이라기 보다는, 현명한 엔지니어링 사례(engineering practice)입니다.” 라고 언급한바 있다. 이러한 이유로 오픈소스 알고리즘이 2002년, 노후된 자료암호표준 (DES, Data Encryption Standard)을 대신하여 진보적 암호 표준 (AES, Advanced Encryption Standard)으로 선정되었다. 이 알고리즘은 미 기술표준국 (US National Institute of Standard & Technology) 주도로 3년간의 글로벌 경쟁 끝에 선정되었다. AES는 현재 미국의 연방기구 뿐만 아니라 민간 분야에서도 광범위하게 사용되고 있다. 이 경쟁은 처음부터 제출된 모든 알고리즘들이 전 세계에 공개되어, 로열티 없이 사용할 수 있도록 하였다.

공개키 암호와 선마이크로시스템스의 보안담당 임원인 Whitfield Diffie씨는 ZDNet에 기고한 기사에서 아래와 같이 말했다, “컴퓨터 보안을 위해 비밀에 의존하는 것은 현실성이 없다. 프로그램의 정확한 작동원리가 사람들 사이에서 배포되는 것을 막을 수는 있겠지만, 하지만 경쟁자에 의해 코드가 역분석되는 것을 막을 수 있겠는가? 아마도 안될 것이다. 강력한 보안의 비밀은, 바로 비밀에 덜 의지하는 것이다.”

30P

데스크탑 컴퓨터 공략

기능과 (MS 오피스와의)호환성의 문제가 크게 보이는 데스크탑 시장에서, 오픈소스는 완만한 움직임을 보여 왔다. 사무직 근로자들에게, 사무자동화는 마치 생명과도 같은 것이다. 즉, 사무자동화는 매일을 업무를 수행하기 위한 핵심 시스템인 것이다. 사무시스템의 사용과 조직 내 외부로부터의 문서 공유나 변경이 용이하지 않을 때, 업무수행은 불가능해진다.

그러나 오픈소스가 데스크탑 컴퓨터 시장의 잠식을 시작했다는 징후가 보이고 있다. 리눅스 채택율이 점차 증가하고 있고, 그에 따라 우리는 오픈소스 계열 오피스 채택율도 완만하게라도 동반 상승하기를 기대하고 있다. 공개 오피스 시스템은 주로 엔지니어링 부서를 통해 대기업에서 분투하고 있다. (Sweet Spot 편 참조). 정부와 기업은 리눅스 데스크탑 컴퓨터에 대하여 진지한 질문을 던지고 있다.

공개 오피스 기술은 3년 전에 비해 훨씬 강력하다. 강력한 경쟁자들 중에는 오픈소스 계열 오피스로서 리눅스, 솔라리스 그리고 윈도우에서 운용되는 OpenOffice, StarOffice, Sun의 오피스 세트(리눅스, 솔라리스, 윈도우에서 운용), 그리고 StarOffice, 리눅스 및 일련의 오픈소스 데스크탑 툴들을 포함하여, MS 윈도우를 노골적으로 겨냥한 Sun의 Java Desktop System 등이 있다.

영국 상무성은 Sun의 Java Desktop System을 자국의 공공기관 용 데스크 탑 솔루션으로 하기 위한 5년 계약서에 서명했다. 아일랜드의 최대 금융기관인 아이리쉬 연합 은행 (Allied Irish Bank)은 7,500명의 데스크탑 사용자를 Java Desktop System으로 옮기는 것을 계획하고 있다. 브라질, 중국, 한국, 인도와 같은 개도국 정부들도 독점 오피스보다 많은 언어로 번역되어 있는 OpenOffice 사용을 증진하고 있다.

관련하여, HP는 리눅스 PC에 OpenOffice를 탑재한 최초의 PC 제조업체로서, 2004년 3월부터 아시아 시장에서 OpenOffice 탑재 PC를 판매하고 있다. 오픈소스 소프트웨어를 발전시켜온 비영리 기구인 오픈소스 개발 연구소(The Open Source Development Labs)는 리눅스를 기업용 데스크탑 컴퓨터에 설치하기 위한 노력을 주도해왔다. HP, sun, IBM, Intel, Novell, Red hat 그리고 OpenDesktop.org가 참여하고 있는 연구소 소속의 “데스크탑 리눅스 워킹 그룹”은 리눅스 전체 솔루션뿐만 아니라 비즈니스에서 사용되는 기타 OS와의 상호운용성을 조사하는데 초점을 맞추고 있다.

CSC LEF의 Bill Koff 부회장은 “데스크탑 컴퓨터가 조만간 리눅스로 대거 이동할 것 같습니다. 회사들이 기업 인프라에 설치된 오픈소스 소프트웨어에 만족감을 느끼면서, 데스크탑에 설치된 오픈소스 소프트웨어에도 곧 만족해할 것입니다” 라는 의견을 개진한다.

MS 오피스와 윈도우 사용자들의 두터운 기반을 고려할 때, 사무용 데스크탑 컴퓨터 시장은 오픈소스에게 여전히 힘든 분야가 될 것이다. 하지만, 그 대안 소프트웨어가 이미 나와 있고, 계속 발판을 마련해 나갈 것이다.

회사들은, 자신들의 IT 전략을 고려할 때, 오픈소스가 백오피스 뿐만 아니라, 기업의 핵심분야에서도 가능한 대안을 제공한다는 것을 기억할 필요가 있다. 찬반 양론을 모두 고려하라. 분명, 다양한 애플리케이션들이 확장성, 처리량 그리고 신뢰성에 대한 다양한 요구사항을 가질 것이다. 오픈소스 소프트웨어가 자신들의 기대를 충족시켜 주는 이들도 있을 것이고, 그렇지 못한 경우도 있을 것이다. 하지만, 오픈소스가 현재 많은 대규모 핵심 애플리케이션에서 역할을 수행하고 있고, 오픈소스 기술이 진보하는 한, 계속해서 그럴 것이라는 것은 명백하다. 오픈소스는 비즈니스 기회에 열려있고, 현명한 조직은 핵심 비즈니스 수행을 위해 오픈소스를 구매할 것이다.

달콤한 부분

오픈소스 활용을 통한 비용절감 목표 달성

오픈소스 소프트웨어의 비용절감 효과에 관하여 엇갈리는 내용의 보고서들이 있지만, 회사들은 목표로 설정한 분야에서 비용이 절감되는 것을 실감하고 있으며, 분야들도 계속해서 확대될 것이다. 오픈소스 소프트웨어가 스택의 상위계층으로 이동하면서, 비용을 절감할 수 있는 기회도 점점 더 커져가고 있다. 소프트웨어 및 장비 벤더들은 비용절감과 다른 이점 때문에 오픈소스 소프트웨어를 신 제품에 점점 더 많이 채택하고 있다. 점점 더 많은 오픈소스 소프트웨어가 제작되고 사용됨에 따라, 비용절감의 가능성도 함께 확대되고 있다.

오픈소스로 인한 비용절감의 핵심은 소프트웨어 라이선스 비용이 들지 않고, 범용 하드웨어 사용으로 인해 하드웨어 비용이 절감되고, 그리고 예기치 않은 작업중단 시간이 줄어드는데 있다. 오픈소스 소프트웨어가 설치되지 않는다 하더라도, 독점소프트웨어의 라이선스 비용과 기술지원 비용을 낮추기 위한 강력한 협상 도구로 사용 가능하다.

물론, 오픈소스 소프트웨어의 가능한 비용 절감액을 정확히 알아내기 위해서는 비용이 포괄적으로, 그리고 건 별로 조사되어야 한다. 비용은 한 분야에서 줄어드는 반면, 다른 분야에선 늘어날 수 있다. 라이선스 비용에서 절감한 것은 기술지원, 트레이닝, 컨설팅, 그리고 IT 경계 밖의 기타 비용으로 상쇄될 수도 있다. 그리고 이것이 새로운 프로젝트가 아니라면, 기존의 애플리케이션 패키지에서 오픈소스 패키지로의 이동에 대한 비용과 작업중단은 엄청날 것이다.

오픈소스 소프트웨어 사용의 경우, 비용을 절감할 수 있는 상당한 기회가 존재한다. 더 적은 비용으로 더 많은 것을 얻기 위해서, 기관들은 이러한 비용절감 기회가 자신들의 사업에 어떻게 적용될 수 있는지를 이해할 필요가 있다.

CSC의 군수 엔지니어링 및 통합 디비전의 Aaron Fuller 사장은 아래와 같이 말했다. “이러한 것이 민간기업에 적용되는 것과 마찬가지로, 군수업체들에게도 똑 같이 적용된다. 오픈소스 소프트웨어와 시스템 아키텍처는 국방시스템의 비용효과(cost effectiveness)를 개선할 수 있는 극적인 기회를 제공한다. 냉전 이후와 9/11 이후의 테러리즘과의 전쟁으로 인해, 국방 시스템은 다수 국가의 무기시스템과 군대가 신속하게 모여 새로운 군대조직을 형성하는 공동 및 연합 작전에 초점을 맞추고 있다. 다양한 무기시스템과 병력들이 공동으로 싸우기 위한 노력이, 단독작전을 전제하여, 독점적인 솔루션으로 설계된 비호환적인 컴퓨터와 소프트웨어로 인해 많은 장애를 겪고 있다.”

“오픈소스 소프트웨어와 오픈 아키텍처 기반의 연합국 군대는 통합된 군사 조직으로 함께 일할 수 있는 상호운영성을 가질 수 있다. 오픈소스와 오픈 아키텍처는 국방분야의 이 혁명에서 가장 강력한 요소들이다. 미국과 그 연합국들의 국방예산은 중요성이 커진 상호운영능력을 향상시키기 위해서 급격히 늘어나지는 않을 것이다. 오픈소스와 오픈아키텍처 기술을 활용하면 뽁뽁한 예산속에서도 상호운영성 개선이 가능하게 될 것이다.”

뽁뽁한 예산에 직면해 있는 정부와 민간 양측은 모두 제한된 리소스를 최대한 활용해야 할 상황에 있으며, 오픈소스는 고려할 수 있는 중요한 대안이다. 오픈소스 활용을 통한 비용절감의 세 목표분야는 소프트웨어 스택, 확장 운영 그리고 소프트웨어 개발이다.

스택에서의 비용 절감

비용 관점에서 소프트웨어 스택을 조사함으로써, 기관들은 비즈니스의 많은 분야에 비용절감 기회가 숨어 있다는 것을 알게 된다. 물론, 각각의 비즈니스 시나리오가 틀리므로, 각 조직은 비용절감 기회가 어디에 존재하는지를 알아내기 위해 자신들만의 계산을 스스로 해야 한다. 아래가 연구할 만한 분야들이다.

플랫폼 – 데이터센터의 리눅스

데이터센터에 리눅스를 사용함으로써 플랫폼 구매비용을 줄일 수 있는 기회가 있다. 물론, 이 기회는 업무량에 따라 달라질 수 있다. 최근까지, 리눅스는 대규모로 확장할 필요가 없거나, 수평적으로 확장할 수 있는 애플리케이션에 적합했다 (예: 서버를 늘리는 등). 하지만, 최근의 리눅스 버전은 수직적 확장을 충분히 지원하여 대부분의 현실세계의 업무부하에 대응할 수 있다. (한 서버에 8개의 CPU가 합리적인 것으로 간주된다.)

32P

수평적 확장은(애플리케이션이 확장을 지원한다는 전제에서) 초기설정 상태에서 가능하며, 범용 서버 사용으로 비용 최소화가 가능하고, 리스크를 최소화할 수 있다. 많은 애플리케이션의 웹서버가 이 방식으로 작업하고 있으며, 요구되는 작업 부하량을 충족시키기 위해 프로세스 스레드를 추가한다. (다음 페이지의 Serverware 참고). 리눅스는 수평적 확장에 제한이 없으므로, 분명 여기서 좋은 대안이다. 불리한 점은, 다수의 애플리케이션 및 운영시스템 인스턴스에 대한 지원을 수반하고, 부가적인 연결들(네트워크와 외부 저장장치에 요구되는 다수의 연결들), 그리고 서버 1대별로 라이선스가 주어진 패키지 소프트웨어에 대해서는 소프트웨어 라이선스 벌금 (penalty)이 주어질 수도 있다. 이상은 각 프로젝트에 대해 평가될 필요가 있는 장, 단점 들이다.

많은 타입의 작업부하에 대해, 리눅스를 활용한 수평확장은 좋은 접근방법이다. 수평적 접근방식은, 현재의 시장상황에서 수직적 접근방식에 비해, 많은 자금과 OS 비용을 줄일 수 있다. 라이선스 비용의 추가 지출 없이, 리눅스를 크게 늘림으로써, 장기적으로 상당한 비용을 절감할 수 있다. 이것은 또한 수직적 솔루션에 대해 가격 하락에 대한 압력으로 작용한다.

수직적 확장은 기술적인 면에서 좀 더 직설적이다. 왜냐하면, 단지 한 개의 애플리케이션 및 OS 인스턴스, 한 개의 네트워크 연결, 한 개의 저장장치 연결, 한 세트의 라이선싱 조건 등등 이 존재하기 때문이다. 그러나, 사이징 관련한 리스크가 훨씬 크다. 만일 서버가 과소평가 되었거나, 예기치 못한 성장이 발생하면, 조직은 대당 가격이 훨씬 비싼 대용량 서버를 채택하여, 서버를 교체할 필요가 있다.

온라인 여행사인 Orbitz는 수평적 확장 방식(수백대의 리눅스 PC)을 택하여 수직적 확장 방식(메인프레임)을 따랐던 기존 경쟁사들보다 가격 경쟁력을 갖추게 되었다. 즉, 수평적, 오픈소스 접근방식을 채택함으로써, Orbitz는 더 빠르면서도 싼 비행노선을 찾을 수 있는 혁신적인 방법을 개발해낼 수 있었다. Orbitz는 티켓예약 프로세스에서 중간상인을 배제할 수 있게 되었으며, 이로 인한 비용절감은 IT 비용 절감과 함께, 최저의 항공운임을 저비용으로 제공한다는 Orbitz사의 사업전략의 핵심을 이루게 되었다.

한 대형 에너지 회사는, 수평식 확장 방식이 오픈소스 소프트웨어 및 일반 하드웨어와 결합했을 때 어떻게 상당한 비용을 절감할 수 있는지를 보여준다. 이 회사는 고도의 지진 연구를 위한 자사의 고성능 컴퓨팅센터를 유닉스를 운영하는 특별한 하드웨어에서 리눅스를 운영하는 범용 컴퓨터 집단으로 업그레이드 하면서 5천만 달러를 절감한 것으로 추산한다. 센터에서, 회사는 1,000개 이상의 리눅스 시스템을 운영하고 있는데, 이것은 영리 목적으로 리눅스를 활용한 세계 최대의 사이트라고 할 수 있다. 동 리눅스 시스템은 Sun이 제작한 오픈소스 그리드 엔진을 사용하여 설치되었다. 회사 추산에 따르면, 리눅스 기반 클러스터로 옮기면서 3백 5십만 달러에 달하던 시스템 유지보수 비용이 이십오만달러로 줄어들어, 약 93 % 정도의 비용을 절감할 수 있었다고 한다. 초기 설치된 클러스터는 1 테라플랍(teraflop, 1초에 1조회 연산 수행)의 슈퍼컴퓨팅 기능을 수행했으나, 이후 컴퓨터와 프로세서를 추가 설치함으로써, 현재 8 테라플랍으로 확장되어 있다.

33P

클러스터 설치를 통해, 예전에 28일 걸리던 작업처리(processing)가 현재는 하루 만에 수행되고 있다.(지질정보의 실시간 분석을 궁극적인 목표로 하고 있다). 이렇게 작업시간이 줄어들면서, 잠재적 시추지역으로 탐사중인 해외 지하지역의 사진을 더 신속하고 명확하게 확보할 수 있

게 되었다. 해외 시추정에 구멍을 뚫을 때마다 약 오십만 달러가 소요된다는 것을 고려할 때, 정확한 위치를 잡는 것은 아주 중요하다. 사진들을 훨씬 빠르게 제작할 수 있게 되면서, 탐사팀은 조정을 통해 훨씬 신속하게 시추 후보지역에 집중할 수 있게 되었다. 이 기능은 오픈소스로 생겨한 오십만 달러의 비용절감 보다도 훨씬 중요한 것일 수 있다.

일반 하드웨어가 이미 설치되어 있을 경우, 리눅스로 인한 비용절감은 라이선싱 및 기술지원 비용에서 나와야 하기 때문에 불명확 할 수도 있다. 리눅스용 기술지원 패키지가 윈도우나 유닉스의 라이선스나 기술지원보다 더 비쌀 수도 있기 때문에, 기관들은 비영리 리눅스 보급판(무료) 택할 수도 있다. 그러나, 상업적 보급판이 가지는 무결성과 보안이 없을 수도 있기 때문에, 이것은 값비싼 대가를 치를 수도 있다.

수행능력면에서, 리눅스는, 다양한 기능으로 인해 오히려 업무수행이 느려질 수도 있는 윈도우보다, 분야에 따라서는 더 나은 업무수행을 할 수 있다. CSC의 글로벌 인프라 서비스 아키텍처 팀의 기술전략 부문 매니저인 Tim Dooley씨는 “경험에 비춰볼 때, 업무수행 능력은 약 20~30 % 정도 개선된다. 하지만, 이것은 업무부하에 많이 달려있다.” 라고 강조한다.

서버웨어

웹서버

일반적으로, 스택의 위로 이동할수록, 비용이 상승하면서, 동시에 비용 절감의 가능성도 커지게 된다. 서버웨어 분야에서, 웹서버 소프트웨어는 오픈소스 활용을 통해 비용을 쉽게 절감할 수 있는 쉬운 분야이다 (low-hanging fruit). (비교적 적은 노력으로, 효과를 얻음). 비용절감은 주로 경미하거나, 아예 0인 구매비용에서 나오며, 큰 조직인 경우 그 절감액이 수백만 달러에 이르기도 한다.

CSC LEF의 Paul Gustafson 소장은 “범용 웹서버 소프트웨어 구매에 1달러라도 쓴다면, 당신은 너무 많이 쓰는 겁니다.”라고 주장한다. Apache는 이러한 주장을 할 수 있을 정도로 충분히 안정되고 또 광범위하게 사용되고 있다 (웹서버의 2/3 정도).

물론, 그 조직이 정교한 웹서버를 필요로 하는 경우, 고도의 기능을 제공하고, 통합 프레임워크를 구성하며, 또는 독립적인 소프트웨어 벤더의 기술지원을 제공하는 독점 웹서버를 요구될 수도 있다. 하지만, Apache로 충분한 경우가 대부분이다. Apache는 매몰비용(sunk cost)이 전혀 없어, 새로운 시스템에 최적인 진입전략(entrance strategy)이다.

Apache는 다수의 OS에서 운영되기 때문에, Apache에 의한 비용절감을 실현하기 위해 리눅스로 바꿀 필요도 없다. 그러나, 리눅스 기반에서 설치되었을 때, Apache는, 대중에 노출된 웹서

버의 핵심 요소라 할 수 있는 무결성과 보안 면에서, 최고의 모습을 보여준다.

애플리케이션 서버

오픈소스인 J2EE 애플리케이션 서버에게 주어지는 기회가, 소규모의 비핵심 애플리케이션에서 조직의 핵심 시스템으로 점점 확대되고 있다. 핵심시스템의 애플리케이션 서버는 이전까지 Websphere(IBM)과 WebLogic (BEA) 같은 대형벤더의 독점 제품들의 영역이었다. 오픈소스 애플리케이션 서버인 Jboss는, 자체의 안정성과 완성도, 그리고 개발자들간의 인기로 인해, 저비용의 귀중한 대안 소프트웨어가 되고 있다.

34P

다수의 독립솔루션업체(ISV, Independent Software Vendors)들은 오픈소스인 J2EE 애플리케이션 서버를 저비용의 개발 플랫폼으로 사용하고 있다. (BroadVision, SeeBeyond 등). ISV들은, 자신들의 소프트웨어가 종종 오픈소스 환경에서 더욱 잘 운용되긴 하지만, 오픈소스 및 독점 J2EE 환경, 양측 모두를 지원하고 있다. 이러한 개발 경향이 소프트웨어 비용을 절감할 수 있는 방안으로 계속되길 기대한다.

Jboss를 애플리케이션 추천 리스트에 올린 대기업들도 있다. CSC는 고객들이 Jboss를 기반의 핵심기능 애플리케이션을 개발하여 설치할 수 있도록 지원해왔다. 덴마크 재무성의 데이터교환 시스템이 좋은 사례이다. 덴마크 재무성은 Jboss와 기타 오픈소스 소프트웨어 사용함으로써 개발 및 설치 시간에서 대략 20%를 절감한 것으로 CSC는 추정한다. (Mission Critical 편 참고)

또 하나의 사례는 CSC가 Jboss, Jetspeed (포탈 프레임워크), 및 Struts (애플리케이션 프레임워크)을 사용하여 제작한 미 해군 포탈시스템이다. 해군은 개발비용 및 시간을 줄이기 위해 오픈소스를 사용코자 하였다. 해군에 따르면, 백 만 달러 이상의 라이선스 비용과 6개월 가량의 개발시간을 줄일 수 있었다고 한다. (최초 개발에 12개월 소요). 독점 포탈 소프트웨어에서는 일반적인 좌석당 라이선스 비용을 지불하지 않음으로써, 해군은 엄청난 비용절감을 실현할 수 있었다. 이러한 비용절감은 포탈 사용자가 늘어날수록 계속 증가할 것이다.

기술지원도 하나의 요소이다. 오픈소스의 경우, 새로운 기능과 수행방법, 그리고 커뮤니티에 의해 새 기능을 수행시키는 방법을 찾아낼 수 있는데, 실제로, Jepspeed에 포틀렛 카테고리 필터링(portlet category filtering)의 추가를 요구하는 포탈 프로젝트에서 발생했다.

35P

오픈소스 소프트웨어와 표준 인터페이스 때문에, 오픈소스 포탈은 새로운 기능과 애플리케이션

의 수행을, 독점소프트웨어로 구축되어 몇 가지 독점적 기능들을 사용하는 해군의 다른 유사한 포털보다, 더 쉽고 신속하게 운영하고 있다. 오픈소스 포털을 개발한 미 해군 기상학 및 해양학 연구센터 (FNMOC)는 레가시(legacy) 애플리케이션들을 변환(convert)시켜 새로운 것을 개발할 수 있는 프로세스를 발견하였고, 이로 인해 오픈소스 포털의 사용속도가 더욱 빨라지게 되었다. 변환된 애플리케이션 중에는 항공기 운항 최적 경로 시스템(OPARS), 검색 및 구조 (SAR) 지원, 그리고 사용자가 관심 지역을 선택한 뒤 기상지도를 다운로드 받을 수 있는 MyWxMap 애플리케이션이 포함되어 있다.

FNMOC는 기상 및 해양 데이터를 각 배에 전송하기 위해 이 오픈소스 포털을 개발하였다. 해군 기상센터인 FNMOC는 복잡한 모델을 가동하여, 다양한 정보를 생산해낸다. 이 정보에는 기상 상황에 대한 원자료, 검색 및 구조 작전을 지원하기 위한 분석자료, 그리고 해군 항공기를 위한 최적항공경로 등이 포함된다. 과거에는 자료 요청이 이메일 또는 전화를 통해 FNMOC로 들어 왔으나, 현재 사람들이 브라우저를 통해 직접 자료 요청을 하고 수 분 내에 자료를 수신할 수 있다.

FNMOC 포털은 독점소프트웨어 기반의 포털의 대안으로, 해군의 웹기반 프로그램 수행을 전담하고 있는 웹 태스크포스 팀(Task Force Web)의 주목을 받게 되었다. 오픈소스 포털은 현재, 자체의 저비용과 매끄러운 개발경로 덕분에 독점 기반 포털에 대한 매력적인 대안으로 부상하고 있다.

FNMOC의 CIO인 Bruce Gritton씨는 “FNMOC 포털과 주변 소프트웨어(오픈소스인 NESSO 및 신청자에 한해 데이터를 배포하는 웹서비스인 RuleBot)는 FNMOC가 내부 프로세스를 현대화하고 ForceNet의 개발 및 통합 노드가 되기 위해 요구 받았던, 신속한 개발을 성취하는데 핵심적이었다” 라고 말했다.

포털의 성공을 증명하듯이, CSC는 현재 오픈소스 포털에 기반한 첫 번째 프로젝트가 될, 새로운 해군 프로젝트 작업을 진행 중이다. 해군의 파트너로서, CSC는, 해군 컴퓨터사건 대응팀이 내리는 정보확인취약 경보(IAVAs, Information Assurance Vulnerability Alerts) 이후, 해군 내의 경보 관련 진행상황을 감시하기 위한 웹기반의 애플리케이션 개발작업을 수행하고 있다. 오픈소스 포털이 제공하는 공개 인터페이스를 사용한 대응팀은 독점 소프트웨어 기반의 해군 포털을 사용했을 경우보다 수개월 먼저 동 프로젝트를 완성했다. 다시 말하지만, 이 프로젝트를 오픈소스 포털에 기반하여 개발한 핵심 요소는 비용과 소요시간이다.

데이터베이스

데이터베이스는 엄청난 가격표를 달고 있다 (기업용 라이선스의 경우 수백만 달러의 가격은 매우 흔하다). 따라서 데이터베이스 분야는 구매비용에서 상당한 비용을 절감할 수 있는 기회가

있을 수 있다. 오픈소스 데이터베이스는 수 년 전부터 존재해왔지만, 기관들과 애플리케이션 벤더들은 이들을 채택하는 것을 꺼려해왔다. 데이터베이스가 사업에 핵심적인 요소라는 것이 하나의 이유가 될 수 있다. 왜냐하면, 데이터베이스는 회사의 핵심 비즈니스 데이터를 저장하고 있기 때문이다. 또한 기존 데이터베이스에 대해 이뤄진 투자와 새로운 데이터베이스를 도입에 따른 만만치 않은 업무도 이러한 현상의 이유가 된다.

오픈소스 데이터베이스가 많이 좋아졌고, 조금씩 기반을 잡아가고 있다고 한다. 오픈소스 데이터베이스가 기존의 독점 데이터베이스를 대체할 수는 없겠지만, 다른 분야, 특히 새로운 애플리케이션에서는 가능한 대안이 될 수 있다. 2004년 3월에 발표한 오픈소스 데이터베이스에 관한 연구에서, Aberdeen Group은 전 세계적으로 오픈소스 데이터베이스의 설치가 주로 중소기업들 중심으로 천 만회 이상 이뤄졌다고 보고했다. (매출 규모가 2천5백만 달러 이하인 기업 또는 워킹 그룹)

Aberdeen 보고서에 따르면, 주요 기회분야는 의사결정 지원 부문이라고 한다. 애버딘은 오픈소스 데이터베이스 관련하여 유저들이 가장 높게 평가하는 항목으로 다양한 작업량(로드) 처리 능력과, 대량의 텍스트와 관계형 데이터를 함께 처리할 수 있는 능력을 꼽았다. 다음으로, 초고속의 업무수행 속도와 유저 수와 스토리지 양의 엄청난 확장성, 그리고 운영상의 용이성을 꼽았다.

36P

Metapa 같은 신생 벤더들은, 고객들이 의사결정 지원과 같은 비즈니스 문제를 해결하기 위해 오픈소스와 범용 하드웨어의 세력 확장을 기대하고 있다는 것에 동의한다. Metapa는 최근 데이터웨어하우징 관련 비즈니스 문제를 해결하는데 중점을 둔 오픈소스 클러스터 솔루션을 출시했다. Metapa Cluster DataBase는 범용 하드웨어 가격 수준의 클러스터(cluster)가 사용될 수 있도록 함으로써, 웨어하우징 문제에 대해 수행능력과 확장성 부분에서 상당한 혜택을 주도록 했다. 리눅스와 PostgreSQL 기반에서 구축함으로써, Metapa는, 기업들이 이제 예전 시스템에 들인 비용의 절반 이하의 비용으로도 데이터웨어하우징과 비즈니스 정보 시스템을 설치할 수 있게 되었다고 믿고 있다.

일반적으로 이러한 데이터베이스 클러스터 시스템은, 현재 많은 리눅스 시스템의 설치가 집중되고 있는 기업 가장자리에 배치된다. 기존 시스템들이 너무 느리거나, 현재의 오픈소스 소프트웨어에 비해 부적절할 때, 그리고 범용 하드웨어 가격의 블레이드 서버(blade server)와 결합될 때, 클러스터들이 종종 사용된다.

Metapa의 Dave Powell 사장은 “우리에게 아직 보이지 않는 것은, 핵심 애플리케이션과 데이터

를 다루는 기업의 핵심부위에 이러한 시스템이 설치되는 것입니다. 기업들은 이 시스템들을 주변부에 설치하여 이들에 대해 좀 더 알게 되고 또 신뢰감을 가질려고 합니다. 이러한 것은 클라이언트-서버 환경, 좀 더 최근에는 인터넷 기반 시스템으로의 이전 시, 고객들이 보여 준 행태와 매우 흡사합니다.” 라고 말한다.

Metapa는 자사의 Cluster Database를 미국 최대의 통신회사 한 곳이 방화벽 서비스를 제공하기 위해 구축한 고객포탈의 부분으로 설치했다. 이 통신회사는 이 오픈소스 기반 시스템을 오픈소스와 범용 하드웨어에서 나오는 명백한 가격적 이점과 더불어, 확장성과 수행능력에서의 장점 때문에 선택했다.

클라이언트웨어

데스크탑 컴퓨터

이전에 언급했지만, OpenOffice같은 오픈소스 소프트웨어는 데스크탑 컴퓨터용으로 점점 더 고려 될 것이다. OpenOffice는 몇 가지 면에서 비용절감이 가능하지만, MS와 같은 독점 소프트웨어와의 호환성 문제가 대량 보급에 장애가 되고 있고, 때문에 오픈소스 오피스 세트를 통한 비용절감은 미래의 기회가 될 것 같다.

오픈소스 오피스 세트는 낮은 구매비용과 강요된 업그레이드 부재로 인해 비용을 절감할 수 있다. 대기업에서 오피스 세트 소프트웨어는 카피(seat) 당 약 300 달러, 또는 10,000명 규모 회사에서 약 3백만 달러의 가격에 판매되고 있다. 이 구매비용이 0가 될 경우 엄청난 비용절감 효과를 거둘 수 있다.

18개월에서 24개월 간격으로 이뤄져, 엄청난 시간과 비용이 들어가는 소프트웨어 업그레이드가 불필요해지면, 또 비용절감이 가능하다. 업그레이드는 업무를 방해하기도 하지만, 실재로는, 불필요한 경우도 많다. 한 보고서에 따르면, 일반 조직의 약 40 %의 인력은 있으면 좋은 정도의 기능(훌륭하지만 없어도 되는)을 제외한, 충분히 좋은 오피스 소프트웨어 만으로도 효율적인 업무수행이 가능한 것으로 나타났다.

그러나, 간과할 수 없는 제한요소는, 오픈소스로서 완전한 세트를 갖추기에는 부족하다는 점이다. OpenOffice가 MS 오피스의 대안 소프트웨어이긴 하지만, MS의 Visio와 같은 기타 툴들에 비견될만한 대안은 현재로서는 존재하지 않는다. 데스크탑 컴퓨터에 탑재될 경우, 이러한 기능을 수행할 수 있는, 여러 가지 대안 툴들을(심지어 오픈소스가 아닌 툴을 포함하여) 채택하도록 요구 받을 것이며, 이에 따라 상당한 추가비용과 리스크, 그리고 작업중단 사태가 초래될 것이다. 다시 말하지만, 호환성이 문제이다.

이 문제는 시간이 걸리겠지만, 리스트의 최상위에 놓여 있다. OpenOffice가 호환성 문제를 해결하게 되면, 오픈소스 오피스 세트는 비용을 절감할 수 있는 중요한 기회를 제공하게 될 것이다. 호환성 문제는 오픈소스 커뮤니티나 소프트웨어 벤더 커뮤니티에 똑 같이 해결해야 할 어려운 문제이다. 오픈소스 소프트웨어와 독점 소프트웨어 진영간 뿐만 아니라, 독점 소프트웨어 간에도(같은 벤더가 다른 시점에 출시한 소프트웨어 포함) 호환성 문제가 있다. 비록, 호환성 문제가 부정될 수는 없지만, 해결은 시간 문제일 뿐이다.

전문 워크스테이션

엔지니어링 워크스테이션이나 POS 시스템과 같은 전문화된 클라이언트의 경우, 독점 유닉스 워크스테이션에서 인텔 기반 리눅스 워크스테이션으로 전환하거나, 다수의 컴퓨터를 복수의 운영시스템을 가동하는 하나의 리눅스 박스(Linux box)로 통합시킴으로써, 비용을 절감하고 생산성을 향상시킬 수 있다.

37P

대형 통신회사에 설치되어 가동 중인 리눅스 계열의 엔지니어링 워크스테이션 프로젝트는 상당한 효과를 내고 있다. CSC의 파트너인 이 통신회사는 자사의 R&D 디자인 엔지니어들이 사용하던 4,600대의 데스크탑과 독점 유닉스 워크스테이션을 대체했다. 그 첫 해에, 회사는 1,500대 이상의 보조컴퓨터를 없애고, 전체의 약 35%인 1,000 대 정도의 워크스테이션을 리눅스 데스크탑으로 대체했다. 지금까지의 과정에서, 회사는 약 8십9만 달러를 절감한 것으로 추산되고 있으며, 향후 3년간 진행될 경우, 총 4백십만 달러 정도를 절감할 수 있을 것으로 기대하고 있다.

이 통신회사는 리눅스 기반 호스트 운영시스템과 윈도우 2000 기반 게스트(guest) 운영시스템으로 구성된 독특한 워크스테이션을 개발했다. 이를 통해, 엔지니어들은 한대의 인텔 기반 컴퓨터로부터, 리눅스와 기존의 유닉스, 그리고 윈도우 환경에서 신 제품을 디자인할 수 있게 되었다. 이 솔루션은 기존 제품 디자이너들을 신제품과 연결시켜 주었고, 동시에 유연성과 민첩성을 제공했다.

동 프로젝트는 R&D의 효율과 효과를 향상시키기 위한 노력의 일환으로 추진되고 있다. 엔지니어링 R&D를 담당하고 있는 회사의 한 IT 지도자는 ‘우리의 목표는 디자이너의 생산성을 증진시키는 것입니다. 우리는, 프로젝트가 바뀔 때마다, 완전히 새로운 데스크탑 환경을 디자인 엔지니어들에게 제공하라고 요구하지 않는, 좀 더 유연한 솔루션을, 그들이 가질 수 있길 원합니다.’라고 말한다.

프로젝트는 성공적으로 수행되어, 디자인 생산성을 증진함과 동시에, 완강했던 유닉스 엔지니어들을 리눅스 팬으로 변화시켰다. 한 엔지니어는 “우리 디자이너들은 로딩과 빌딩에 걸리는 시간이 줄어드는 것을 좋아합니다. 로딩 및 빌딩 시간이 10배 이하로 줄어들어서, 이전에 몇 시간 걸리던 것이 지금은 몇 분에 끝나고 있습니다.” 라고 말했다.

그 IT 지도자는 “사람들이 오픈소스는 피크시간대에는 적절치 않다고 말했을 때, 나는 부정했습니다. 우리 회사의 R&D 환경에서, 오픈소스는 분명 준비가 되어 있습니다.” 라고 주장했다.

엔지니어링 환경에서, 주요 관심사는 애플리케이션의 가용성과 전달이다. 엔지니어링 애플리케이션은 매우 전문화되어 있고, 그들 중 많은 수가 리눅스를 지원하지 않는다. 따라서, 이 애플리케이션이 리눅스로 포팅되어 있지 않다면, 또는 리눅스로 포팅될 때까지, 오픈소스로 인한 비용 절감 혜택을 완전히 누리지는 못할 것이다.

이 포팅 문제는 앞서 언급한 에너지 회사에서 해결되었다. 이 회사는 자신들이 고도의 시각화를 위해 사용하던 애플리케이션을, 소프트웨어 벤더가 리눅스로 포팅시켰던 것이다. 이 에너지 회사는 1,500대의 노후한 유닉스 엔지니어링 워크스테이션을 리눅스 기반 범용 시스템으로 교체하고 있다. 30,000 달러에서 300,000 달러에 달하는 워크스테이션들이 20,000 달러의 박스로 대체되기 때문에, 3년 후, 교체 프로젝트가 완결되면, 약 2천5백만 달러의 비용이 절감될 것으로 예상된다.

통신회사에서와 마찬가지로, 절감된 비용의 일 부분은 사무용 보조 PC의 통합에서 나온다. 왜냐하면, 리눅스 시스템의 일부가 윈도우 및 그 오피스 소프트웨어를 사용하기 때문이다.

확장 작업에서의 비용 절감

오픈소스 소프트웨어는, 비용절감이 관련된 수백만의 유닛에서 쏟아지는 제조업이나 서비스 회사 같은 대용량 처리 상황에서, 주요한 비용 절감의 기회를 제공한다.

한 제조업체가 수 백만 대의 임베디드 리눅스 PDA나 전화기를 판매할 때, 비용절감 승수는 엄청나다고 할 수 있다. 오픈소스 소프트웨어를 가진 수백만 대의 데스크탑을 지원하는 서비스 회사도 마찬가지 이다. 제조업체나 서비스회사가 부담하는 오픈소스 소프트웨어의 총소유비용(TCO)은, 비용절감이 수백만 건의 경우에서 전체적으로 발생하기 때문에, 크게 줄어든다. 이것은 결국 소비자나 최종소비자(end-user)들의 비용이 줄어드는 것을 의미한다. 예를 들어, CSC는 아웃소싱 비즈니스의 일환으로 1백만대의 데스크탑 컴퓨터와 4만대의 중간급 서버를 관리하고 있다. 대조적으로, 일반적인 대기업은 5만대의 데스크탑 컴퓨터와 3천대의 서버를 보유하고 있다. 총소유비용은 이러한 작은 기반에 의해 구속받는다.

은행의 현금출납기, PDA, 그리고 핸드폰 같은 단일기능 장비들과 소비자 제품은 임베디드 리눅스 환경에서 잘 운용되며, 이것은 최소의 시스템 자원과 기술지원이 요구되며, 그리고 거의 추가비용이 들지 않는다. 리눅스는, 새로운 소프트웨어 개발에 대한 요구가 적기 때문에, 소비자 제품의 가격을 낮추고, 제품의 전체 개발과정의 속도를 높여주는 역할을 한다.

38P

예를 들어, 모토라는 제조원가를 낮추고 타임투마켓(time-to-market) 시간을 줄이기 위해, 스스로 개발할 필요 없이 이미 개발된 오픈소스 소프트웨어를 활용한다는 의도에서, 임베디드 리눅스를 핸드폰 등의 제품에 사용했다. (Invisible Man 편 참조)

소프트웨어 개발에서의 비용 절감

재활용(reuse)와 지속성(consistency)에 기반한 오픈소스 소프트웨어 개발 방식을 활용할 경우 상당한 비용절감을 실현할 수 있다. 앞서 언급한 통신회사는 효율성 증진을 지원하기 위해 R&D 분야의 소프트웨어 개발을 위해 오픈소스의 협업 모델을 고려 중이다. 신 제품을 개발하면서, 기업 및 통신회사 시장을 겨냥하여 하드웨어 및 소프트웨어를 설계한 이 회사는 각기 다른 100개 이상의 독점 플랫폼들을 하나의 작은 표준 세트로 통합하려 하고 있다. 회사는 복제 기능을 제거함으로써 소프트웨어 개발과 야외 유지보수에서 상당한 효율성 증진을 기대하고 있다.

이러한 효율 향상을 위해, 회사는 오픈소스 모델에 따라, R&D 분야를 협업을 통한 개발 커뮤니티로 한데 묶고 있는 중이다. 이를 통해 재활용과 소프트웨어 컴포넌트들의 통합이 늘어날 수 있을 것이다.

협업개발을 통한 비용절감은 반복적으로 일어날 수 있다. CSC는 VP/MS Model Manager 제품을 오픈소스 계열의 통합 개발 환경인 Eclipse에 기반함으로써, 제품 개발에 이용한 개발 프로세스를 활성화시켰다. (Market Force 편 참조). 결과적으로 개발 프로세스를 따로 만들 필요가 없어 타임투마켓(time-to-market) 시간을 줄일 수 있었으며, 유지보수 비용도 절감되었다. CSC가 VP/MS Model Manager 프로젝트로부터 나온 플러그인(plugin)들을 Eclipse 핵심 플랫폼에 추가한 후, CSC는 그것들의 유지보수 업무에서 해방되었다. Eclipse 커뮤니티가 그 일을 대신 맡아 주었기 때문이다.

소프트웨어 개발에서 볼 수 있는 비용절감의 다른 사례는 미 해군이 대규모의 분산 및 실시간 컴퓨터 시스템의 운영환경을 관리할 수 있도록, CSC가 개발한 Insight 시스템이다. 개발 스케줄과 다른 제약들로 인해 솔루션을 처음부터 개발할 수 없는 상황에서, 개발팀은 컴포넌트 솔루션

을 찾기 위해 오픈소스들을 뒤지기 시작했다. 개발팀은 툴에서부터, 네트워크 서비스, 그리고 형상관리(configuration management)와 GUI에 이르는 약 십오만 라인에 이르는 오픈소스 코드를 통합하게 되었고, 결과적으로 소프트웨어 개발에서 수백만 달러의 비용 절감과 10개월이라는 개발기한도 지킬 수 있었다.

비용에 대한 면밀한 고찰

오픈소스 소프트웨어의 사용에 관한 결정을 내릴 때, 해당조직은 간과되거나 또는 잘 이해되지 않는 몇 가지 비용들을 이해할 필요가 있다. 즉, 전환 비용, 법적 비용, 오픈소스 커뮤니티에 제공해야 하는 리소스, 그리고 장기적 관점에서 오픈소스 프로젝트의 생존능력 등이 이에 포함된다.

귀사의 독점 소프트웨어 플랫폼(또는 소프트웨어 세트)을 오픈소스 소프트웨어로 전환시키는데 대체 비용이 얼마나 될까? 취득비용으로부터 비용 절감액을 결정한다면 아주 명확하지만, 하지만, 이전비용(migration cost), 작업중단, 새 플랫폼에 대한 트레이닝과 지속적인 기술지원 등을 계량화 하면 훨씬 복잡해진다. 세부적인 총소유비용(TCO) 분석이 필요하다. 사람이나 조직이 아무리 정직하다고 할지라도, TCO 가설은 낙관적이거나 비관적이 되기 쉽다. 실재, 객관적인 TCO는 얻기 힘들다. 40 ~43 페이지에 걸쳐 나와 있는 TCO 테이블이 가능한 객관적인 TCO 수행을 위한 기본 지침이 되어 줄 것이다.

오픈소스가 글로벌 커뮤니티에 기반한 다른 지원모델을 사용하기 때문에, 회사들은 오픈소스 커뮤니티에 대한 지원 담당자(들)를 지정할 필요가 있다고 결정할 수 있다.

마지막으로, "현재 제가 사용하는 오픈소스 소프트웨어의 장기적 관점에서의 생존가능성은 어떨습니까?" 라는 질문을 하는 회사가 많이 있다. 소프트웨어 제품 관련하여 한 벤더보다 자발적 커뮤니티에 의존하는 것에는 리스크가 있다. (벤더에게 의존하는 것도 동일한 리스크라고 주장하는 이도 있긴 하지만...). 소프트웨어 보수와 개선이 적절한 시간과 믿음직한 방식에서 이루어지지 않을 수도 있다. 오픈소스 소프트웨어를 보급하고 지원하기 위한 상업적 사업모델은 아직 태동하는 단계이다. 마찬가지로, SCO가 한 주장의 영향은 무엇이며, 우리가 아직 감지하지 못하고 있는 잠재적 지재권 위반 내용은 무엇인가?

39P

기관들은 오픈소스에서 가능한 비용절감의 기회가 어디에 있는지 자세히 연구할 필요가 있다. 그 달콤한 부분은 조직에 따라 틀릴 것이다. 하지만, 그 부분은 분명 존재한다. 관리자들도 오픈소스 아젠다에 관한 정보를 입수하고, 작업에 착수해야 한다. 그렇지 않을 경우, 돈을 날리는 위

험에 처하게 될 것이다.

그림설명 1

정보시스템의 총소유비용은, 단순히 취득비용이나 개발비용 뿐만 아니라, 기술지원, 트레이닝, 기술개발, 시스템 이전 동안의 작업 중단, 스태프 재배치 등을 수반하기 때문에, 복합적이다.

40P

총소유비용 분석은 기업에 따라 달라진다. 몇가지 TCO 모델에 기반한 본 테이블은, 오픈소스 소프트웨어를 평가할 때 고려해야 할 가장 중요한 TCO 컴포넌트들을 요약한 것이다. 기업들이 자신들의 환경에서 오픈소스의 사업적 가치를 결정하기 위해서는 TCO 및 기타 변수들(기술적 우위, 보안, 벤더에로의 종속, 소프트웨어의 장기적 생존능력, 기회비용 등)에서의 요소를 포괄적으로 볼 필요가 있다.

TCO 컴포넌트	오픈소스 소프트웨어(OSS) 영향	코멘트
----------	--------------------	-----

구매 가격 하드웨어 유지보수	OSS는 독점 하드웨어 (RISC 기반 유닉스) 대신에, 저렴한 범용 하드웨어(인텔 기반 리눅스)를 적극적으로 사용할 수 있는 계기를 마련했다. 어떤 작업량에 대해서, 이를 통해 하드웨어 비용을 상당히 절감할 수 있으며, 특히 수평적 확장이 이루어질 경우 절감효과는 더욱 커지게 된다. 범용 플랫폼으로 옮겨지는 상업 애플리케이션으로부터도 역시 비용이 절감된다. SAP같은 애플리케이션들이 독점 플랫폼으로부터 Intel/Linux로 옮겨질 경우 30 % 이상의 비용 절감 효과를 가질 수 있다.	SUN, IBM, 그리고 HP 모두가 인텔 기반 서버들과의 경쟁을 위해서 자신들의 범용 RISC/Unix 서버 제품 라인의 가격을 상당 폭 내렸다. 이러한 가격 하락은 리눅스와의 경쟁과, 작업량이 플랫폼으로부터 Intel/Linux로 옮겨가는 것을 벤더들이 목격하고 있다는 사실에 의해 촉진되었다. RISC 서버와 Intel 서버의 CPU 당 가격은 범용 세계에서는 매우 근접해있다. 따라서, CPU 카운트와 같은 단순한 시각으로는 하드웨어 비용 절감의 기회가 거의 없어지게 되었다. 이러한 상황은 바로 2년전과 비교해도 상황이 변한 것이다. 리눅스 초기 시대에도, RISC와 Intel 플랫폼 사이에는 상당한 가격 차이가 있었다. 하지만, 경쟁과 범용화로, 그 차이가 상당폭 좁아지게 된 것이다. Silicon Graphics과 같은 독점 하드웨어 벤더조차 범용 플랫폼을 선호하여 범용화 움직임을 촉진시켰다. 이외에도, Intel/Linux 서버와 더 노골적으로 경쟁하기 위해 더 낮은 비용의 서버 라인을 발표한 벤더들도 있었는데, Sun의 V 시리즈가 이에 해당된다. 그러나, 범용의 Intel CPU는 RISC CPU보다 훨씬 큰 폭으로 가격과 수행능력 면에서 계속 발전해갈 것이다. 따라서 비용 분석은 CPU 숫자보다는 동등한 처리 용량을 비교하는 것에서 시작해야 할 것이며, 이를 경우 균형추는 Intel/Linux 쪽으로 훨씬 더 기울어 지게 될 것이다.
--------------------------------	---	---

TCO 컴포넌트	오픈소스 소프트웨어(OSS) 영향	코멘트
직접 소프트웨어 비용	라이선스 비용은 OSS의 해당사항이 아니다. 그러나, 배포와 기술지원으로 인해 비용이 일어난다. 일반적으로, 독점소프트웨어는 높은 구매비용과 비교적 저렴한 유지보수 비용을 가진다. OSS의 경우 그 반대의 가격 정책을 가지며, 상업적 지원을 받는 보급판이 채택된다. 취득비용은 아예 없거나 아주 낮는데 반해, 지속적인 유지보수 비용은 상당히 높은 편이다. 기업들은 리눅스의 상업적 보급판(Red Hat 또는 Novell/SUSE)을 구매하라는 강력한 추천을 받기 때문에, Linux는 윈도우보다 훨씬 비쌀 것이다. 비영리 목적의 보급판을 사용할 경우 리눅스를 통한 비용절감을 보이는 것도 가능하다. 하지만 추가 비용이 다른 곳에서 나타날 것이다(기술 지원 등)	Linux를 채택하면, 즉, RISC 하드웨어 기반의 유닉스 기반 소프트웨어에서 Intel 기반 Linux로 전환하면, 여러 부분들에서 상당한 비용절감을 기대할 수 있다. 또한 시스템에 요구되는 CPU 숫자를 줄여서, CPU 당 부과되는 라이선스 비용도 줄일 수 있다. Intel의 작업수행 능력이 개선되면서 적어진 숫자의 CPU만으로도 같은 작업량을 처리할 수 있다. 그러나, 소프트웨어 벤더들은 그들의 가격 구조를 변화시켜, 약 30% 정도가 되는 이 이점을 최소화하거나 아예 제거 해 버릴 수도 있다. 리눅스 이외의 비용절감은 독점소프트웨어 대신에 다른 오픈소스 소프트웨어를 채택함으로써 성취할 수 있다. 이것은 주로 라이선스 비용에서 나온다. 기업들은 OSS를 다운로드 받거나, 상업적 보급판을 구매하는 경우도 있다. OSS는 일반적으로 기업이 사용하기에 충분히 안정되어 있다. 전체적으로 볼 때, OSS는 경쟁을 격화시켜, 독립솔루션벤더(ISV)와 소프트웨어 벤더들에게 가격 하락을 위한 압력을 가했다.

TCO 컴포넌트	오픈소스 소프트웨어(OSS) 영향	코멘트
간접 소프트웨어 비용 라이선스 관리	일반적으로, OSS의 경우 설치된 사이트의 합법성을 확인하기 위해 소프트웨어 라이선스의 번호를 추적하는 것은 필요하지 않다. 그러나, 보안과 패치의 목적에서 소프트웨어가 어디에 설치되었는지를 확인할 필요가 있다. 즉, OSS 덕분에, 소프트웨어가 설치되는 것을 가혹하게 감시해야 하는 부담으로부터 벗어나게 되었다. 어떤 컴포넌트를 조직에 사용하 지에 관한 전략을 수립하는 것은 중요하다. 이 전략은 OSS 컴포넌 트에 관한 숙련된 노하우를 지닌 전문인력들에 의해 정기적으로 평 가 받아야 한다.	소프트웨어 감사는 비용이 많이 들 수 있다. 사내의 어떤 이가 어떤 소프트웨어를 사용하는지를 정확하게 알아낼 수 있는 감사 프로세스를 제작하고 관리하는 것은 결코 작은 프로젝트가 아니다.
인력수급 비용 - 프로젝트 관리 - 시스템 엔지니어링/개발 - 벤더 관리 - 행정 (구매 등) - 트레이닝	인력수급 비용은 수요와 공급 상황에 따라 달라진다. 이전에, 리눅스 전문가들은 공급이 달리기 때문에, 윈도우 전문가보다 높은 급여를 받을 수 있었다. 하지만, 리눅스 및 리눅스 전문가들의 확산과 함께, 그들의 임금은 윈도우 전문가들과 비길만하게 되었다.	인력수급 비용은 사내 IT 인력의 숙련도에 따라 달라진다. 예를 들어, 한 회사가 장기간 전략 플랫폼으로 윈도우를 운영해왔고, 따라서 리눅스에 경험이 있는 인력이 없다면, 장기적으로는 효과가 있겠지만, 리눅스로 전환하는데 상당한 비용이 들 것이다. 한 회사의 가용한 인재 풀은 어떤 소프트웨어가 그 회사에 가장 적합한지를 결정하는데 중요한 역할을 한다.

TCO 컴포넌트	오픈소스 소프트웨어(OSS) 영향	코멘트
기술지원 비용 - 설치 및 구성 - 문제해결 - 지원 툴, 자원 - 패칭 - 동료지원 - 일상 습득 (매뉴얼, 시행착오 등) - 정규 트레이닝 - 애플리케이션 개발	일반적으로 리눅스보다는 윈도우가 악의적인 해커들의 주 타겟이 되며, 따라서 리눅스는 윈도우보다 보안 패치가 훨씬 적게, 하지만, 유닉스보다는 많이 요구된다. 리눅스의 상용 보급판은 출시 이전에 적절한 테스트 과정을 거치며, 자동 패칭 툴을 가지고 있다. 만일 사용하고 있는 것이 상용 리눅스 보급판이 아니라면, 최신 패치를 유지하기는 어렵다. 왜냐하면, 너무나 짧은 시간에 많은 변화가 일어나기 때문이다. 경험에 비춰볼 때, OSS의 경우 설치와 구성이 더 어려웠다. 하지만, 사용하기 쉬운 설치기 (installer)를 가진 주요 OSS 제품의 경우는 설치에 어려움이 없을 것이다. 문제해결의 경우, OSS에서 비용이 더 낮을 것이다. 트레이닝 비용은 문제가 되는 OSS에 따라 더 높을 수도 있다.	윈도우 구 버전에 비해, 리눅스환경에서는 문제해결과 보안패치 관련 작업이 더 용이하다. 특히 리눅스 상용 보급판에서 전문적이고 효율적으로 자동패치가 제공된다는 것을 고려할 때는 더욱 그러하다. MS는 윈도우 XP와 윈도우 서버 2003에서는 많이 좋아졌다고 말하지만, 이들 운영시스템에서도 벌써 많은 패치를 내놓았다. 자동 패칭을 위한 툴이 리눅스와 윈도우 모두에 존재한다. CSC는 현재 기업들에게 광범위하게 사용되는 이들 툴 모두를 사용한다. 오픈소스 환경에서 커뮤니티의 지원은 일반적으로 주요 OSS 제품에 도움이 된다. OSS 뉴스그룹은 종종 상용 벤더들보다 더 빠르게 문제에 대응하기도 한다. 물론 그들이 그렇게 즉각적인 반응을 보여야 한다는 계약에 의한 어떠한 요구도 없다. 너무나 광범위하게 확산되어 사실상의 업계 표준이 되어버린 독점 소프트웨어 제품들이 있다. (MS 오피스 등). 이들은 사람들 사이에 잘 알려져 있고, 따라서 업그레이드를 제외하고는, 기술지원 비용이 거의 들지 않는다. 그러나, 유사한 OSS 제품은 일반적으로 사용되는 제품을 교체할 경우 업무의 중단 상황을 일으킬 수 있다.

작업중단 시간 (Downtime)	리눅스의 모듈방식은 매우 간결한 시스템 설치를 가능하게 하며, 이에 따라 윈도우보다 더 안정되고 더 가용성이 있게 된다. (유닉스 환경 역시, 아주 안정되어 있다..)	예기치 않은 작업중단 시간은 총소유비용의 약 50%에 이를 수 있다. 그럼에도 불구하고, 중단시간은, 계량화의 어려움 때문에, TCO 분석에 포함되지 않는다.
-----------------------	---	--

44P

소프트웨어 혁명:

오픈소스가 개발을 가속화하고 새로운 아이디어를 창출한다.

44P

소프트웨어 개발의 새로운 규범은 협업과 활용이다. (오픈소스 커뮤니티의 개발자들과의 협업과 오픈소스 소프트웨어의 활용). 조직의 구속 내에서 일하고 처음부터 모든 것을 창조하기 보다는, 개발자들은 개발 프로세스를 앞당기기 위해 이미 사용가능한 오픈소스 툴과 컴포넌트를 사용하고, 기술지원과 영감 고취를 위해 오픈소스 커뮤니티를 적극 활용해야 한다.

오픈소스 커뮤니티는 새로운 아이디어와 혁신을 위한 인큐베이터와 같은 곳이다. 커뮤니티는

아이디어를 유포시켜, 청중들(개발자)을 대화와 개발에 참여시켜 특정 질문이나 문제를 해결하기 위한 개념 (코드)을 제작하게 한다. 그 원동력은 마치 상용 애플리케이션을 염두에 두고서, 기술적으로 난해한 또는 중요한 아이디어를 추구하는 소프트웨어 개발 연구소와 마찬가지로이다. 하지만, 오픈소스 환경에서, 회사의 예산이나 엔젤 투자자에 대한 부담이 없는 커뮤니티는 아이디어를 스스로 공개할 수 있다. 이것은 가히 혁명적이다. 커뮤니티 기반의 소프트웨어 개발은 개발 프로세스를 가속화시키고 혁신시킨다.

커뮤니티 기반 개발에는 네가지 모델이 있다.

- 소프트웨어 개발을 위한 오픈소스 활용
- 신 제품의 연관 개발
- 기존 제품의 확장
- 한 제품을 상용화를 위한 회사 설립

소프트웨어 개발을 위한 오픈소스 활용

활용모델은 당신의 개발노력을 최대한 이용하기 위해 기 개발된 오픈소스 소프트웨어를 사용하는 것에 관한 것이다. 오픈소스 소프트웨어를 핵심기능과 개발 툴로 사용한 CSC의 Proactive Service Management (PSM) 솔루션을 실재 사례로 들 수 있다. PSM 개발에 사용된 오픈소스 소프트웨어는 이전에 검증이 되었고 (문제를 해결했고), 그리고 사용하기 쉬웠기 (사람들은 사용방법을 이미 잘 알고 있거나 또는 신속하게 배울 수 있다) 때문에 선택되었다. 이외에도, 무료이기 때문이다. (구매비용이 들지 않음).

PSM을 설치한 회사들은 복잡한 네트워크로 연결된 원거리 장비들을 실시간으로 관리할 수 있게 되었다. PSM을 사용한 최초의 고객인, 미국의 한 초고속 인터넷 사업자는 90 %의 전화통화 감시 기능을 자동화시킴으로써 수백만 달러의 비용을 절감할 수 있었다. 뿐만 아니라, 기술인력 연구활동의 약 75 %를 자동화 시키고, 이 정보를, 고객이 전화했을 경우 즉시 문제를 해결할 수 있는, 전화서비스 인력들이 볼 수 있도록 함으로써, 평균복구시간 (Mean Time to Repair, MTTR)을 48시간에서 4시간으로 줄였다.

PSM 솔루션의 약 25%가 오픈소스 소프트웨어로 구성되었으며, 이들 중 주요한 것은 MySQL, PHP, 그리고 Expect이다.

MySQL은 리포팅 목적으로 데이터를 저장하기 위해 사용되었다. MySQL은 애초에 임시 솔루션으로 선택되었으나, 수행능력이 우수했기 때문에 살아남았다. 그리고 값비싼 독점 데이터베이스로 가면서 생기는 추가적 원가비용(오버헤드)도 용납할 수 없는 것이었다. PSM 프로그램의

Bob Solis 소장은 “이것은 MySQL이 ‘충분히 좋은’ 경우였습니다. MySQL은 데이터베이스에 대한 우리의 요구사항을 정확히 만족시켜주었습니다.” 라고 말한다. 데이터 저장 이외에도, MySQL는 PSM 세트의 여러 면의 통합, 애플리케이션의 확장, 그리고 로깅 등에 사용되었다.

그림설명

전통적 소프트웨어 개발(폐쇄적/성당)과 오픈소스 개발(개방/바자회)간의 경계가 애매해지면서, 소프트웨어 R&D의 성장과 혁신을 촉진하고 있다.

45P

PHP는 진단부의 의사결정 지원 시스템에 사용되어, 시스템의 모든 노드의 상태를 보여주는 수 페이지의 리포팅 정보를(회사 전체의 운영에 중요한 정보) 브라우저 기반으로 제공하였다.

개발자의 툴킷에서, Expect (오픈소스 스크립트 언어)는 다양한 전화 스위치(AT&T의 5ESS등)와 케이블 박스와의 상호작용을 자동화하는데 사용되었다. 이 장비들 중 많은 수가 표준 애플리케이션 프로그램 인터페이스가 없거나, 또는 있다 해도, 비용 때문에 사용이 거의 되지 않았다. 그래서 Expect가 사용되어, 그들의 코맨드 라인 인터페이스를 통하여 장비들과 상호작용하게 되었다. 이것은 중요한 극적 돌파구였다. 왜냐하면, 이를 통해 장비들을 자동적이고 적극적으로 모니터 및 통제할 수 있게 되었고, 나아가 예전의 수동조작을 없애고, 문제가 일어나기 전에 찾아낼 수 있게 되었기 때문이다.

Solis씨는 “여기에는 거의 사용할 수 없을 정도로 비싼 APIs를 사는 것 이외에는 도저히 일을 수행할 방법이 없었기 때문에 오픈소스가 사용되었습니다.” 라고 설명했다. 결과는 창조적이면서도 강력한 혁신이었다. 자동화된 전화통화 감시 기능이 설치된 이후, 고객센터는 크게 개선됐고, 사업비용 또한 극적으로 절감되었다.

개발자 툴킷의 다른 오픈소스 툴에는 Apache, Perl, Log4J, CVS, Samba, Jitterbug(결점 추적), DBTools(데이터 관리 프리웨어), Emacs (에디터 프리웨어) 및 GNU Compiler Collection (GCC)가 포함되어 있다. 소프트웨어 개발에서 또 하나의 오픈소스 활용 사례는, 한 기업 전체의 정보시스템 취약성을 평가하는 CSC의 보안 툴인 H.E.A.T.이다. H.E.A.T (Hydra Expert Assessment Technology)는 몇 개의 오픈소스 핵심 컴포넌트를 사용하여 개발되었으며, 이로 인해 최초 13개월의 개발기간 중 적어도 3개월을 단축할 수 있었다.

H.E.A.T 프로그램 매니저인 Jason Arnold씨는 “오픈소스 소프트웨어를 사용하지 않았더라도, H.E.A.T.는 개발될 수 있었을 것입니다. 하지만, 개발에 시간이 더 오래 걸렸을 것이고, 지금만

큼 강력하지도 않을 것입니다.”라고 진술한다.

H.E.A.T.는 보안 툴이기 때문에 원천적으로 안전한 코드를 가지는 것이 중요하다. 오픈소스 소프트웨어는 많은 사람들이 검토했기 때문에, 큰 도움이 되었다. (28 페이지 Security sidebar 참고). 독점소프트웨어 사용을 꺼리는 다른 이유는 의문스런 성능(소스코드를 볼 수 없으므로, 소프트웨어가 어떻게 작동되는지, 그리고 얼마나 잘 통합되어 있는지 알 수 없다), 라이선싱 문제(CSC는 H.E.A.T.를 자체 라이선스 하에서 판매하고 있다), 그리고 비용 최소화 등이다.

그림설명 1

원거리의 장비를 연결하는 복잡한 네트워크에 대한 다양한 기능을 제공하는 CSC의 Proactive Service Management(PSM) 솔루션은 핵심 기능을 위해 오픈소스를 활용하고 있다.

그림설명 2

H.E.A.T (Hydra Expert Assessment Technology)는 한 기업 전체의 정보시스템 취약성을 평가하는 보안 툴로서 CSC가 개발하였다. 많은 오픈소스 소프트웨어 컴포넌트를 사용하는 H.E.A.T.는 고도의 컴퓨터 범죄자들의 무자비한 기술을 보여주는 전략을 사용하여, 크고 복잡한 네트워크를 자세하고 빠르게 분석하는 기능을 수행한다. 개발의 관점에서 봤을 때, 오픈소스 소프트웨어를 사용함으로써 H.E.A.T.는 강력할 뿐만 아니라 안전한 솔루션이 되었다.

46P

H.E.A.T.에서의 핵심 오픈소스 소프트웨어에는 Apache, MySQL, OpenSSL, mod_SSL (OpenSSL에 대한 Apache의 인터페이스), “John the Ripper” (패스워드 크래킹), Blowfish (암호 알고리즘) 및 GD Graphics Library (리포트 그래픽스) 등이 포함되어 있다. 오픈소스 소프트웨어를 사용함으로써, 개발자는 H.E.A.T. 자체에 좀 더 집중할 수 있다. 예를 들어, Apache를 사용함으로써, 개발자들은 맞춤 웹서버를 제작하는데 시간을 보내는 것 대신에 제품의 핵심기능인 취약성 검사 코딩에 좀 더 집중할 수 있었다.

H.E.A.T.는 네트워크 상에서 운용되면서, 지정된 장비를 조사하면서 취약점을 찾아내고, 그 결과를 보고한다. 회사는 이후, H.E.A.T.가 발견한 취약점을 옮기거나 제거한다. H.E.A.T.가 한 기업에서, 서버에서 네트워크 스위치에 이르는 수 천 대의 장비를 검사하는 것은 아주 일반적이다. H.E.A.T.는 비교될만한 제품보다 훨씬 빨라서, 일반적인 취약성 평가의 경우 1주일 이면 완결된다.

SourceForge는 나에겐 친구와 같은 존재다. 개발자로서, 단지 다른 이가 그것을 어떻게 했는지에 관한 아이디어를 얻는 것일지라도 나는 SourceForge에 먼저 간다.

세계적인 화학회사인 Huntsman Corporation(유타 주 솔트레이크 시 소재)의 글로벌 IT 보안 매니저인 Ken Ferguson씨는 “H.E.A.T. 제품의 신뢰성 때문에, 사무실에 인력이 없어도, 6개 대륙과 거의 모든 타임 존에서 장비에 대한 평가를 시행할 수 있게 되었습니다. H.E.A.T. 설치 이후, 이 제품을 사용한 지난 2년의 기간 동안에 걸쳐, 시스템의 취약한 부분을 지속적으로 줄여올 수 있었습니다. 우리는 전 세계 각 지역에 퍼져 있는, 100여 개 사이트들에서의 진행상황을 비교할 수 있는, 객관적인 툴을 사용하여 성취한 성과로서, 안전한 환경에 대한 우리의 노력을 경영진에게 증명했습니다.” 라고 보고한다.

개발의 관점에서 봤을 때, Paul Gustafson LEF 소장은 “H.E.A.T. 와 PSM은 개발자가 오픈소스를 사용하지 않을 경우, 그들은 경쟁력에서 우위를 가지지 못했을 것이다” 라고 단언한다.

실제로, 오픈소스는 문의(enquiry)의 최전방에 있어야 한다. Arnold는 “SourceForge는 나에겐 친구와 같은 존재다. 개발자로서, 단지 다른 이가 그것을 어떻게 했는지에 관한 아이디어를 얻는 것일지라도 나는 SourceForge에 먼저 간다” 라고 재미있게 말한다.

오픈소스 소프트웨어는 타임투마켓(time-to-market) 시간을 단축시키고, 고품질의 결과물을 제작할 수 있도록 도와준다. Solis씨는 “오픈소스로서 가용한 툴을 가짐으로써, 우리는 당장 생산적이 되고 또 업무를 수행할 수 있게 된다” 라고 주장한다. 오픈소스는 언제든지 변경될 수 있고, 강력하며 안전하기 때문에, 맞춤 개발 프로젝트에서 핵심적인 요소이다.

신제품의 붓스트랩

기존의 오픈소스 소프트웨어는 새로운 아이디어의 원천이다.

수동적 처리반응시간 모니터인 PasTmon은 붓스트랩 모델의 좋은 예다. PasTmon은 오픈소스 유틸리티로서, CSC의 graham Bevan이 개발하였다. 기술적 난제를 해결하고 싶어 했던 Bevan은 (당시 TCP/IP 세계에선 네트워크 트래픽 분석기가 없었다), 그 솔루션의 핵심을, 오픈소스 계열의 네트워크 침입감지시스템인 Snort에서 발견하였다. Snort가 GPL 하에서 배포되었기 때문에, Snort의 여러 부분들을 통합하여 제작된 PasTmon도 오픈소스로서 GPL하에서 공급되었다.

Bevan씨는 “그때 나는 보안에 관한 실험을 하면서 Snort를 가지고 놀고 있었지요. 그때, 처리 감시기를 제작하는데, Snort를 활용할 수 있겠다라는 생각이 들었어요. 마찬가지로, 아이디어를 얻기 위해 더 큰 규모의 오픈소스 커뮤니티를 이용했었지요” 라고 회상한다.

PasTmon은, Bevan씨가 작업을 시작한지 4개월 만인, 2001년 3월에 SourceForge에 공개되었다. 유사한 독점 제품은 아마도 공개하기 까지 훨씬 더 많은 시간이 걸렸을 것이다. 현재, 비교될 만한 독점 제품이 나와 있긴 하지만, 서버당 연간 수 천 달러의 비용이 든다. PasTmon은 무료다.

기존 제품의 확장

오픈소스는 기존 제품을 연장시키는 데도 사용될 수 있다. 이것은 LEGO MINDSTORMS 만큼 재미있고, OpenCyc만큼 심각한 제품에서 일어났다.

자사의 MINDSTORM 로봇 키트를 역분석하려는 해커들에 대응하면서, 오픈소스 개발 툴을 제작하려던 장난감 제조업체 LEGO는, 제품 스펙을 완전히 공개하여 로봇 프로그램을 더 쉽게 하고, 고도의 프로그래머 키트 및 설명서를 발표함으로써, 오픈소스 커뮤니티를 지원하였다. LEGO는 자사 제품을 개발 및 확대하기 위해 오픈소스 커뮤니티를 활용하였고, 결국 새로운 기능을 제품에 추가시키고 또 활발한 개발 커뮤니티를 양성하였다. (Fun Factor 편 참고)

47P

OpenCyc는 좀 더 발전하기 위해 기존 제품인 Cyc가 오픈소스가 된 또 하나의 사례이다. Cyc는 인공지능 전문가인 Doug Lenat 주도로 MCC(Microelectronics and Computer Technology Corporation)에서 개발이 시작된 1984년 이래 축적되어 온 일반상식 관련 지식 베이스이다. Lenat씨와 Cyc 프로젝트는 MCC를 떠나서, 1994년 Cyc를 설립한다. Cyc는 처음부터 오픈소스가 되고자 했으나, 어느 정도의 개발이 먼저 이뤄져야 된다는데 동의하게 된다. Cyc는 다른 애플리케이션이 운용되는 플랫폼으로 위치하게 된다. 2002년, Cyc 시스템의 여러 부분들이 오픈소스로서 공개되었다.

오픈소스 소프트웨어 개발로의 변화에는 Cyc의 지식 베이스를 성장시키고, Cyc 기반에서 운용되는 새로운 애플리케이션을 제작하려는 두가지 측면이 있다. 근본적인 고려사항은 Cyc를 지식 대표, 지식관리, 그리고 지능 소프트웨어 애플리케이션의 표준으로 정립하기 위한 것이었다.

20년 간에 걸쳐서, 약 2백만 건의 주장들이 Cyc의 지식 베이스에 채워졌다. 개발자는 “Cyc가 정말로 탁월해지려면, 그 숫자가 약 2억이 되어야 하며, 이것이 바로 오픈소스 소프트웨어 개발이 필요한 이유이다” 라고 주장한다. (코드를 세계 커뮤니티에 공개하여 더 많은 주장들을 신속하게 추가하는 것.) 이것은 마치 인간 게놈 프로젝트와 같다. 일의 1%를 이루는데 수 십 년이 걸렸으나, 그 작업의 99 %를 이루는 데는 십 년이 더 걸릴 뿐이다. 이 놀라운 성취를 이루기 위해

서는 더 많은 사람과 더 좋은 품질의 툴을 필요로 하는데, 이것은, Cyc가 오픈소스가 되고, 사용자에게 편리한 (user-friendly) 툴을 더 많이 채택 하는 등, 지금까지 걸어온 경로와 일치한다.

Cyc를 위한 새로운 애플리케이션에 초점을 맞추고 있는 한 회사가 패키지 산업에 쓰일 소프트웨어를 개발하고 있다. 이 회사는 패키지 산업에 쓰일 온톨로지(ontology)를 개발하기 위해 OpenCyc와 공동작업을 하고 있으며, 완성되면, Cyc에서 운용할 것이다. (온톨로지는, 사람들이 예를 들어 “고객”이란 단어에 대해 같은 의미를 가질 수 있도록, 산업용어를 명확하고 지속적인 방식으로 정의한다.) 이 아이디어는 온톨로지를 오픈소스로 개방하여 산업계 전체가 사용토록 고취시키고, 그리고 온톨로지를 이용하여 자사가 개발중인 독점 비즈니스 애플리케이션에 대해서 요금을 청구하는 것이다.

전체적으로, 오픈소스가 되고, 더 광범위한 개발 커뮤니티에 접근함으로써, Cyc와 지식 애플리케이션은 성장 잠재력을 가진다.

제품을 상용화하기 위한 회사 설립 (Forming a Company to Commercialize a Product)

: 이 모델은 단지 하나의 신제품이나 제품확장이 아니라 오픈소스를 통해 전체 회사를 활성화시킨 것에 관한 이야기다. 이것은 Intalio사에 관한 이야기다.

젊고 야심만만한 소프트웨어 엔지니어로써, Isamel Ghalimi는 궁극적으로 Intalio사가 된 것에 대한 비전을 가지고 있었다. Ghalimi는 그의 꿈을 실현시키기 위해, 이미 오픈소스에 정통한 개발자로 구성된 커뮤니티를 구성하였다.

Ghalimi의 비전은 차세대 인터넷 기반의 기업용 애플리케이션 플랫폼이었다. 그는 북미와 유럽에 산재해 있던 개발자들에게, 함께 일 할 경우, 혼자일 때 보다 더 큰 영향을 업계의 거물인 IBM이나 MS에 대해 끼칠 수 있다고 설득했다. 작업 중인 오픈소스 소프트웨어를 기증한 사람들이 있었고, 모듈들은 합쳐져 몇 개의 병렬 프로젝트를 구축하게 되었다. 처음부터 목표는 커뮤니티가 상용 제품을 제작하는 방향으로 함께 일하는 것이었다. 이 제품은 궁극적으로 Intalio의 비즈니스 프로세스 관리시스템(BPMS, Business Process Management System)이 되었다. (CSC와 Intalio는 프로세스 관리(business process management) 표준개발을 위해 Business Process Management Initiative(BPMI.org)를 공동 설립하였다.)

Intalio가 상업용 소프트웨어 기업이지만, 자사 제품의 근원에 있는 오픈소스 소프트웨어는 ExoLab.org에서 구할 수 있다. ExoLab는 Intalio보다 먼저 설립된 오픈소스 커뮤니티이다. (Ghalimi는 ExoLab를 1999년에, Intalio를 2000년에 설립했다.) ExoLab는 Intalio에 두가지 측면에서 기여했다. Intalio에 대한 라이선스 하에서, ExoLab는 Tyrex Object Request Broker와 같은 기존 상업 소프트웨어의 오픈 소스 버전을 공급했다. 상업적으로 성공한 소프트웨어를 복

제하기 위해 오픈소스를 사용함으로써, Intalio는 생존가능한 사업을 가지기에 충분히 강력한 인프라를 개발할 수 있었다. 여기서, 오픈소스가 개발의 혁신을 위한 박차를 가했다. .

48P

ExoLab는 XML 자바 바인딩 소프트웨어인 Castor 역시 공급했다. 이것은 순수한 제품 혁신이었다. Castor는 많은 상용 제품의 핵심에 있다.

오늘날, Intalio의 BPMS는 새로운 분류의 소프트웨어로써, 최첨단으로 간주된다. 오픈소스 개발 프로세스로 시작함으로써, Ghalimi와 그의 커뮤니티는 혁신적 개념을 확인하고, 오픈소스가 아니었으면 구매했어야 했을 컴포넌트를 이용하고, 그리고 신속하게 신 제품을 개발 할 수 있었다.

오픈소스를 활용한 혁신

연구개발은 새로운 아이디어, 즉 혁신에 관한 것이다. 그러나, 오픈소스 세계에서, 혁신은 때때로 “me-too” 제품들에 의해 도전 받는다. 그 오픈소스 제품은 기존 제품의 다른 버전 아님니까? 우리는, 비록 가끔씩 덜 명확하고, 전통적이긴 하지만, 대부분의 오픈소스 소프트웨어는 혁신적이다 라고 주장한다.

사전적 의미에서의 혁신은 “새로운 어떤 것의 도입” 이다. 소프트웨어 개발 환경에서, 혁신은 상당한 변화, 또는 돌파구를 암시한다.

하지만, 혁신은 번덕스럽다. 어떤 환경에서 혁신은 다른 환경에선 평범한 것이다. 미용실에서 이메일은 혁신이겠지만, 대부분의 회사에선 평범한 것이다. 혁신은 최초의 Mosaic 브라우저 처럼 신 제품이 될 수도 있고, 오픈소스 Mozilla Firefox 브라우저의 tab이나 익스텐션(extension) 처럼 기존 제품의 특징이 될 수도 있다.

Firefox의 탭 브라우징으로 인해, 유저는 한 윈도우에 여러 페이지를 열수가 있다. Firefox는 익스텐션(extension) 장치 덕분에, 플러그-인과 유사한 작을 프로그램을 통해 추가적인 기능을 제공할 수 있다. Firefox가 시장에서 상승세를 타는 것은 이러한 독특한 특징에 힘 입은 것이 분명하다. (Firefox는 브라우저 순위에서 여전히 다크호스로 머물고 있긴 하다)

플러그-앤-플레이 확장 기능으로 인해 Firefox 사용자는 자신의 브라우저를 쉽고 신속하게 개인화할 수 있는 반면, 개발팀은 모든 새로운 기능을 공급해야 한다는 부담을 덜 수 있게 되었다. 대신에, 누구든 익스텐션을 추가할 수 있다. 현재, 150개 이상의 익스텐션이 있으며, 여기에는

Mouse Gesture(명령을 호출하기 위해 Mouse gesture를 개인화), QuickNote (브라우저에 노트 기능), 그리고 SearchThis! (E-bay, Wikipedia 그리고 Dictionary.com 같은 웹사이트를 right-click 메뉴에 추가) 등이 포함되어 있다.

익스텐션 이외에도, 익스텐션 장치 자체가 혁신을 자극하기 때문에, 혁신적이다.

한 눈에 보기에, 모방(me-too)로 보이는 오픈소스 제품은 Jboss J2EE 애플리케이션 서버다. 대충 봐도, J2EE 애플리케이션 서버를 개발한 것은 혁신적인 것은 아니다. 기능은 모든 벤더가 반드시 이행해야 하는 J2EE 스펙에 정해져 있다. Jboss는 전형적인 “me-too” 프로젝트로 시작하여, 이미 자사 제품을 시장에 확립시켜 놓았던 BEA(WebLogic)과 IBM (WebSphere)과의 극심한 경쟁에 직면해 있다.

그럼에도 불구하고, Jboss는 2001년 당시, 대단한 혁신적이었던 JMX (java Management Extension)에 기반한 초소형 커널 주변에서, 애플리케이션 서버를 느슨하게 결합된 세트의 서비스들로 수행함으로써, 기본적으로 다른 아키텍처를 사용했다. 이것은, 개발자들이 더 쉽게 새로운 기술을 변화시키고, 채택할 수 있다는 것을 의미한다. 이것이, 신속한 개발을 위한 오픈소스 개발 모델의 민첩성과 결합되어, JBoss 오픈소스 프로젝트의 성공으로 이어졌다.

Jboss는 2003년에 출시된 Jboss의 차기 버전의 기반으로, J2EE를 aspect-oriented programming (AOP)와 결합시켜, 새로운 영역에 진입했다. 다시, 이 조치는 기업 컴퓨팅에 대한 기본적으로 새로운 시각을 도입시켰고, J2EE 세계에서 주목을 끌게 되었다. 비록 Jboss가 그러한 테크놀로지와 처음으로 연합된 경우인지 아닌지 간에 JBoss는 처음으로 개선하고 마케팅하는데 성공하였다. 전형적으로 시장으로 처음으로 진입하는 BEA같은 메이저는 스위트를 따라갔다.

49P

문자메세지(Jabber는 이 독점 영역을 열고 표준에 관한 개선작업을 하고 있다), 애플리케이션 개발(에클립스는 InfoWorld의 최고의 애플리케이션개발툴부분에서 2003년 이해의 테크놀러지상을 수상), 그리고 OS를 포함한 많은 영역에서 공개소프트웨어는 수많은 영역에서 공개소프트웨어는 기술혁신을 계속하고 있다. 리눅스는 IT의 경제와, 이어서 대형비즈니스와 정부 시스템을 가동시키는 기업을 변화시키고 있다.(미션크리티컬참조) 수많은 애플리케이션, 휴대용품 그리고 소비자용 장치 등에서 이들 제품의 기능을 풍부하지만 가격은 적절하게 하는 방법을 발견하고 있다.(투명인간 참조)

2002년으로 돌아가서, 리눅스에 10억불을 투자한 IBM사의 대변인은 NewsFactor에 말하기를 “

저희는 리눅스가 미래 운영환경이 될 것이라고 믿습니다. 왜냐하면 리눅스는 그 어느 것보다 다르게 혁신의 연료를 제공하기 때문입니다.”¹⁷

공개운동은 전반적으로 혁신에 연료를 제공하되 소프트웨어뿐만 아니라 커뮤니티가 참여되는 모든 훈련에까지 어려운 과학에서는 이를 출판하도록 연료를 제공한다고 많은 사람들이 믿고 있다.(새 영역 참조) 아이디어는 자유롭기를 원하며 오픈소스는 아이디어를 꺼내어 사람들이 함께 일하고 학습하도록 한다.

오픈소스제품이 궁극적으로 사용되도록 선택되지않더라도 오픈소스는 또한 경쟁을 촉진하기도 한다. 오픈소스라는 대체물이 현존하는 것 자체가 독점 소프트웨어의 가격을 낮추도록 하고 조직들에게 새로운 협상 톨을 제공하는 것이다(시장의 위력)

그러나 점점 갈수록 오픈소스는 선택 옵션이 되어가고 있다. 많은 경우 오픈소스는 이전에는 존재하지않았던 곳에(개발도상국이나 제품개발에) 컴퓨팅을 들여오고 있다. 클레이튼 크리스틴슨 (Clayton Christensen) 과 마이클 레이너(Michael Raynor)의 “혁신가들의 솔루션(The Innovator’s Solution)”에서 이 경우 공개소프트웨어는 파괴적인 혁신이라고 한다. 공개소프트웨어는 비소비와 경쟁하여 감당할만한 기능성을 제공하는 것이다.

공개소프트웨어개발은 민첩해서 사람들로 하여금 여기저기 살펴보게 하고 방해되지않는 새로운 아이디어를 산출해 내도록 한다. 이러한 민첩함으로 오픈소스는 소프트웨어 개발 혁명을 주도하고 혁신을 가속화시킨다.

당신을 위한 서비스로: 서비스는 새로운 비즈니스 기회를 연다

공개소프트웨어의 취약점은 기술지원이라고 보편적으로 인식되고 있다. 그럼에도 불구하고IT 서비스 및 솔루션 제공사, 소프트웨어벤더, 애플리케이션 서비스 제공자 등 공개소스분야에 풍요로운 비즈니스 기회를 제공하는 것이 기술지원이기도 하다.

커뮤니티 기반의 지원

기술지원은 공개소프트웨어에 관한 비평거리 중 하나이다. 공개소프트웨어는 어떻게 작동하는가? 누가 책임지는가? 문제발생 시 소유권은 누구에게 있는가? 기술지원이란 문제의 해결, 교육(트레이닝), 매뉴얼, 패키징, 통합 및 기타 전문적인 서비스 등을 망라하는 것이다.

전통적인 애플리케이션 개발환경에는 두 가지 지원 시나리오가 있다. 하나는 구매자가 지원하는 것(스스로 지원하기) 또 다른 하나는 벤더가 지원하는 것(전형적인 고객-벤더 관계)

경우에 따라서는 세 번째 시나리오로 구매자와 벤더가 지원에 따르는 ‘고통’을 공유하기도 한다. 이 시나리오에서는 양쪽이 소스 코드에 주목하는 까닭에 이 시나리오를 통해 공개소스의 세상과 연결되게 된다.

‘고통-공유하기’ 접근의 예는 CSC의 가상구매자개발(virtual customer development)이 있다. 재무 서비스 애플리케이션의 대표적인 벤더인 CSC는 보험업 애플리케이션의 소스코드를 구매자들과 공유하며 구매자들은 이 소스코드를 변경할 수 있으나 CSC의 도움을 받을 경우에 가능하다. CSC는 이 소프트웨어의 다음 버전에 맞춤주문을 포함시킴으로써 구매자들의 부담을 경감시키고 본 제품라인을 강화시키고 있다. (다른 분야와는 달리 보험업계에서는 요구되는 맞춤주문의 수준이 높은 까닭에 애플리케이션은 소스코드형태로 배송된다. 이러한 핵심 애플리케이션은 보험사에서 ‘제조되는 라인’이며 이것이 바로 시스템에 고객맞춤화한 제품이다.

오픈소스분야의 새로운 지원 시나리오는 구매자, 벤더, 서비스제공자 그리고 오픈소스커뮤니티가 혼합되어진 것이다. 구매자-커뮤니티 시나리오는 구매자가 스스로 지원하며 필요 시 커뮤니티에 정보를 입력한다. 벤더-커뮤니티 시나리오는 벤더가 지원을 제공하고 오픈소스커뮤니티에 정보를 입력한다(Novell/SUSE, RedHat 혹은 TiVo와 같이 오픈소스 임베디드 소프트웨어 OEM).

서비스제공자-커뮤니티 시나리오는 서비스 제공자가 하나의 솔루션에서 공개소프트웨어를 사용하고 전체 솔루션에 대해 지원하고 공개소스커뮤니티에 정보를 입력한다. 이것은 공개소프트웨어를 포함한 CSC사의 보안 취약성 진단 제품인 H.E.A.T에서 잘 나타난다.(소프트웨어 혁명 참조) CSC는 클라이언트를 앞단에서 지원한다.

보다 많은 협력이 이뤄지는 시나리오에서는, 서비스 제공자가 공개소프트웨어에 대해 책임을 지고 클라이언트와 협력하는데 이 협력관계는 커뮤니티와 연계된다. 상업적인 공개소스배포판이 관여되는 경우 공개소스벤더가 이들 협력시나리오에 참여하기도 한다.

커뮤니티는 구매자, 벤더 혹은 서비스 제공자간에 이뤄지는 지원상호작용을 증폭시켜준다. 커뮤니티가 많은 종류의 기능을 수행할지라도(예를 들면 초기구동, 혁신, 동지애), 커뮤니티의 다양한 기능 가운데 지원역할은 제품 기능성에 상당히 중요하며 궁극적으로는 구매자 만족도와 직결된다.

신뢰받는 중개자로서의 서비스 제공자

아직도 많은 조직들은 공개소스커뮤니티와 직접적으로 접촉하는 것을 불편해 하거나 혹은 그다

지 관심이 없다. 이들이 원하는 것은 안정된 영리 단체에 대한 기대, 신뢰, 구조인 것이다.

51p

구매자 가치: 공개소스는 어떤 점에서 다른가

레드햇은 공개소스가 구매자들에서 제공하는 특별한 가치를 4 가지로 제시한다.

구매자들이 참여가능하고 활동적인 대규모의 개발자 커뮤니티	공개소스커뮤니티 내의 개발프로젝트들은 25만에서 75만개 가량으로 추정된다. 가장 규모가 큰 공개소스프로젝트인 리눅스는 3천만 라인을 넘는 코드가 특징이며 R&D비용면에서 관례적으로 드는 1억불이상 효과를 갖는다. 이러한 개발 커뮤니티의 가치는 진행중인 소프트웨어 개발과 로드맵에 접근, 참여하여 구매자들에게 도달하는데, 이 방식은 독점 소프트웨어 개발 모델로는 불가능한 것이다.
---------------------------------	--

전 세계 유저들로 구성된 대규모 커뮤니티	리눅스는 서버와 클라이언트부분에서 마켓쉐어를 넓혀간다. 규모와 전 세계적이라는 공개소스의 특성은 공개소프트웨어기술 혁신을 가속화 시키며 기업용 소프트웨어와 하드웨어를 취급하는 대형벤더들사이에서 공개소프트웨어에 대한 대대적인 지원을 이끌어내고 있다.
------------------------	--

고용, 교육, 혁신 및 코드 품질에 기여하는 이러한 커뮤니티의 연대 네트워크 효과	대규모 커뮤니티에서 개발하고 사용함에 따라, 숙련된 공개소프트웨어 개발자와 IT기술지원 인력풀이 마련되었다. 공개소프트웨어와 연계된 개발자와 유저 규모가 클수록 혁신되는 정도도 커지며, 코드 품질 역시 공개소스커뮤니티와 연계된 규모가 커짐에 따라 더욱더 향상된다
---	--

코드 소유권과 코드 접근의 유연성	공개소프트웨어는 소스코드에 적절하게 접근가능하고 업그레이드와 보안패치와 소스코드개선이 지속적으로 가능하다. 공개소프트웨어는 지속적으로 패치를 배치할 수 있어 이제까지 가장 급진적으로 발전되고 있는 소프트웨어이다.
--------------------	--

52p

공개소프트웨어와 소스코드가 무료로 가능해짐으로써 제 3자들에게도 공개소스지원에 관한 책임을 지게 되었다. 가트너는 이러한 동향에 대해서 예견하기를 “가장 대중적인 공개소프트웨어

최소한 100개에 대한 보증과 추가 지원이 상업적인 소프트웨어 벤더나 서비스 제공자 혹은 보험사에 의해 2005년까지 제공될 것이다”

LEF의 VP인 빌 코프(Bill Koff)에 의하면 “공개소프트웨어를 사용하는 조직은 신뢰할만한 중개자를 찾을 것이다. 왜냐하면 책임질 자가 필요하기 때문이다” 해결할 수 없는 문제가 발생 시 책임을 추궁할 누군가가 필요하다는 것은 관리자 관점에서 합리적이다. 전통적으로 애플리케이션 개발환경에는 위험을 경감시키는 제대로 수립된 프로젝트 관리 업무흐름이 있다. 이러한 프로세스는 단계적 확대(에스칼레이션), 책임소재, 벌금(패널티), 법적 분규등의 이슈를 제대로 정의하고 있으며 프로젝트 관리자와 책임자는 이러한 프로세스에 숙달된 사람들이다.

공개소스 기술지원 책임, 구매자와 커뮤니티
간에 중재는 서비스 제공자의 새로운 가치
제안이 되고 있다

그러나, 이러한 위험경감 프로세스가 공개소프트웨어에는 없다. 그 결과 구매자에 대한 기술지원이 불가능하게 인식되기 때문에 구매자들은 공개소스에 대해 불안감을 갖게 된다. 이러한 책임공백을 어떻게 설명해야 할 것인가?

앞단에서 공개소스 기술지원을 제공하여 책임 공백부분을 메꾸는 서비스 제공의 역할은 과거 제품 벤더들에게 기술지원을 꺼려왔던 서비스제공자에게 부여된 새로운 역할이다. 이제 서비스 제공자는 제품에 대한 서비스부분에서 보다 직접적인 역할을 하며 구매자와 공개소스커뮤니티 간에 중개자 역할을 하게 된다. 예를 들면, CSC는 대규모 아웃소싱계약을 할 때 앞단에서 공개 SW 기술지원을 하는 것이다. 공개소스기술지원에 대해 책임 지고, 구매자와 커뮤니티를 중개하는 것 등은 서비스 제공자에게 부여된 새로운 가치 포지션이다. 아웃소싱하는 클라이언트는 각별히 이러한 역할을 기대할 것이다. 예를 들면 호주 남부정부가 2003년 1조억불로 추정하는 아웃소싱계약을 연장하려할 때 공개소스가 계약 협상의 핵심 부분이었다.

IT서비스와 기술지원이 서비스 제공자의 주요 비즈니스이기 때문에 서비스 제공자는 공개소스 기술지원을 떠맡는 것으로 제대로 포지션되었다. 공개소프트웨어 기술지원이 이들 비즈니스를 확대시켰으나 여기에는 주목해야 할 점이 있다. 서비스제공자는 독점 벤더에게 일방적으로 문제를 넘겨주기보다는 반드시 공개소스커뮤니티와 협력해야 하는 것이다.

서비스제공자들에게 이 부분은 중요한 이슈이지만 많은 사례들을 보면 이 이슈는 구매자들에게는 아주 투명하다. 구매자는 솔루션이 공개소스이든 아니든 서비스 제공자가 솔루션을 제공하고 지원하기를 기대한다. CSC의 경험에 따르면 구매자 관점에서 공개소스 기술이슈는 존재하지 않는다는 것이다.

CSC에 따르면 공개소프트웨어에 대한 기술지원은 일반적으로 추정되는 것 이상으로 뛰어나다는 것이다. 공개소스제품 솔루션을 주로 사용하는 CSC의 덴마크 e-솔루션수석인 한스 자야티사에 따르면, “버그가 발견되면 우리 내부에서 정정하기보다는 공개소스 프로젝트팀에 주로 보고하게 된다. 대개 공개소스프로젝트팀은 24시간에서 48시간 내에 문제를 해결할 뿐아니라 새로운 방안까지 제공할 수 있기때문이다. 이러한 수준의 기술지원은 상업적인 벤더들에게서 기대하는 지원이상이다.”

Role for pure Play and Software vendors too

구매자 입장에서 뿐아니라(수요) 공개소스가 실용가능한 서비스 비즈니스라는 경향을 확신하고 있는 공급자의 입장에서 서비스 제공자가 공개소스를 수용하는 것은 일리가 있다. RedHat(리눅스), Novell(수세 리눅스), JBoss(애플리케이션 서버), MySQL(데이터베이스), Jope(콘텐츠 관리), VA 소프트웨어(소스포지)등의 공개소스회사들은 개선된 소프트웨어와 기술지원, 컨설팅, 통합, 교육 및 매뉴얼(다큐멘테이션)등 일련의 수수료 기반의 서비스를 제공하는 성공적인 비즈니스 추진하고 있다.

53P

순수 공개소스회사 중 가장 앞선 레드햇은 기업의 독특한 수요에 겨냥하여 성공적인 비즈니스를 수행하고 있다. 레드햇기업리눅스제품라인은 공개소스개발모델에서 제공되는 기술혁신의 효과를 톡톡히 본다. 기업하드웨어와 소프트웨어벤더들은 레드햇을 통해 그들의 제품을 인증해주는 표준플랫폼을 갖게 되었다. 레드햇은 공개소프트웨어에 필요한 계측가능성과 보안을 제공함으로써 미션 크리티칼한 리눅스 개발을 가능케했다. 레드햇은 기업을 개선시키는 레드햇리눅스의 배치, 통합, 업데이트, 관리, 교육 및 기술지원을 도왔다.

레드햇은 버그수정, 유지보수 업데이트, 새로운 기능, 시스템관리 기술 및 기술지원의 책임을 완수한다. 레드햇은 공개소스커뮤니티의 기술혁신에 다리를 제공한 것이다. 레드햇은 공개소스와 리눅스의 가치를 물려받아서 구매자들이 리눅스와 공개소스의 유익을 이해할 수 있는 편리하고 책임감있는 실체를 만들어 낸 것이다.

그림.

레드햇은 공개소스커뮤니티의 코드로부터 출발하여 서비스, 테스트, 개선을 추가하고 개별 소프트웨어벤더(ISVs),와 개별 하드웨어벤더(HVs) 인증을 추가하여 리눅스를 ‘제품화’시켰다. 이것이 바로 레드햇기업리눅스이다. 18개월마다 새 버전이 출시되고 끊임없이 버그수정이 이뤄지고

레드햇네트워크를 통해 보안 정오표가 출시된다. 버그수정과 보안 정오표를 진행하는 업데이트는 3개월에서 6개월마다 출시된다. 모든 버전은 모든 업데이트를 통하여 이진 호환성을 binary compatibility 유지보수된다. 모든 버전은 5년간 지원된다(새로운 하드웨어 플랫폼은 첫 3년동안 보증된다). ISVs와 IHVs는 새 버전이 나올때마다 보증된다.

페도라프로젝트는 레드햇이 스폰서하고 커뮤니티가 지원하는 공개소스프로젝트이다. 이 프로젝트는 결과적으로 레드햇제품을 가능케하는 신 기술을 위한 기초를 제공한다. 페도라프로젝트는 레드햇이 대가없이 출시하는 공개소스코드이며 4개월 혹은 6개월마다 새 버전이 출시된다. 페도라프로젝트는 공개소스개발에 첨단을 견고자하는 엔드유저, 개발자, 연구원들에 의해 사용된다.

54P

“이 페도라프로젝트는 공개SW가 가진 실제 비즈니스의 문제점들을 해결하고 시장에 등장하게 하였을 뿐아니라 공개소스운동의 네트워크 효과를 인식할 수 있도록 하였다”라고 레드햇의 CTO 마이클 티만은 말한다. “네트워크의 가치는 연계된 유저 혹은 도구의 수효를 제공함으로 증대한다는 Metcalfe의 법칙은 공개소스에도 적용된다. 참여하는 개발자가 많을수록 새로운 아이디어 생성을 계속하는 공개소스커뮤니티의 위력을 더욱 강화시킬 것이며 궁극적으로는 모든 이가 윈-윈하게 되는 것이다”라고 덧붙인다.

결과는 성공적이었다. 2004년 2월 말 보도에 의하면, 경쟁사가 동일한 기간에 3800개(subscription)의 매출을 낸 반면 레드햇은 87,000개의 레드햇리눅스 매출을 올렸다고 보도되었다. 이로인하여 레드햇은 ISVs 및 하드웨어 OEM사들에게 영향력을 끼칠 수 있도록 했다. 기업 하드웨어와 소프트웨어벤더들에게 그들의 제품을 인증할 표준플랫폼을 제공함으로 레드햇의 소프트웨어는 900여개 이상의 인증된 ISV 애플리케이션에서 사용되고 기업에서의 소프트웨어의 신뢰도와 사용을 강화시켰다.

노벨은 공개소스 비즈니스 사례를 구축한 또 하나의 벤더이다. 침체기를 겪었지만, 공개소스로 재활성화되어 가장 앞선 기업 리눅스 배포회사인 수세리눅스와 합병하게 된다. 전형적인 독점 기업인 노벨은 공개소스로 회사에 생기를 불어넣었다. 공개소스는 진정한 서비스비즈니스라고 표명한다(10년간의 'Opensoft' 움직임이 될 것이다)

가장 최근의 움직임은, 2004년 6월 HP가 JBoss와 MySQL에 기술지원을 제공하기 시작한 것으로 이것은 안정적인 독점 벤더들이 공개소스 서비스제공자의 역할로 제대로 포지셔닝되고 있음을 보여준다. HP, Sun, Novell, 그 외 벤더들은 리눅스 구매자들을 위해 SCO 법적 소송의 가능성에 대응하여 보상을 제공하며 이것은 다양한 서비스와 신뢰할 수 있는 제공자로서의 벤더의 역할을 제시한다.(법과 비즈니스 이슈 참조)

New Opportunities for ASPs

ASPs를 위한 새로운 기회

애플리케이션 서비스 제공자들 역시 공개소스 영역으로 들어오고 있다. ASPs는 이를테면 서비스 비즈니스를 활성화시킬 수 있는 무료 소프트웨어로 된 애플리케이션과 같이 실용적인 비즈니스 기회에 주력하고 있다.

공개소스이 ASP에 제안하는 것은 무료 라이선스비용 비즈니스이다. 공개소스애플리케이션을 호스팅함으로써 ASP는 라이선스 비용을 지불할 필요가 없는 것이다. 이것은 ASP의 시장진입장벽을 낮추어, 소규모 업체들과 현금은 부족하나 경력은 풍부한 신생기업들이 시장에 진입할 수 있게 된다.

예를 들면, 신생기업 Workspot은 ASP로서 리눅스 데스크탑 플랫폼과 애플리케이션을 제공하였다. 유저들은 월 9.95달러를 지불하고 자신들의 브라우저를 통해 workspot에 의해 호스팅되는 이 플랫폼에 접근할 수 있게 되었다. Workspot의 비전은 개인을 위한 무한한 기계, 어느 곳에서든지 가능한 영원한 데스크탑을 제공하는 것이다. Workspot이 5년 후(2003년에)는 해체되었지만, ASP 전략으로서는 아주 앞선 사고였다.

신생회사들에 더불어, 전통적인 ISPs 역시 공개소프트웨어를 활용하여 ASP 영역으로 확장할 수 있도록 할 수 있었다. 탄탄한 기술 인프라를 지닌 ISPs는 이 분야에서 제대로 포지셔닝한다. 필요한 것은 ASP시장을 뜨겁게 할 교스크, 휴대용품, 전체 애플리케이션에 접근이 가능한 브라우저 등 보다 가벼운 클라이언트 환경이다.

확실히 공개소스가 소프트웨어 인프라와 애플리케이션을 편리하게 공개소스의 비즈니스 기회는 점점 증가하게 된다. 서비스 기회는 애플리케이션 배송에 관한 전문적인 서비스에서부터 엔드유저에게 애플리케이션과 서비스를 직접 제공하는 것으로까지 확대된다. 공개소스의 지도자인 팀 오레일리(Tim O'Reilly)는 후자를 일컬어 공개소스를 위한 '감춰진 서비스 비즈니스 모델'의 예시라고 한다.

오레일리에 의하면, 감춰진 서비스 비즈니스 모델로 몇 가지가 더 있다고 하며 ISPs가 되는 것도 이에 해당한다고 한다. 예를 들면, 오레일리는 현재까지 가장 성공적인 공개소스 비즈니스는 레드햇이 아닌 UUNet이라고 주장한다. UUNet(MCI의 인터넷 유닛이 됨)은 공개소프트웨어로 수익분의 성공적인 ISP비즈니스를 이뤄냈는데, 1억달러로 추정되는 레드햇의 기세를 꺾는 것이다.

서비스가 왕이다

무료로 공개소프트웨어를 사용하고 기술지원에는 비용을 지불하는 구매자를 확보하는 것이 공

개소스개념의 비즈니스의 첫 단계이다. 그 다음 단계는 기술지원과 더불어 ASP에 의해 제공되는 서비스로써 공개소프트웨어 혹은 애플리케이션에 구매자가 비용을 지불하는 것이다.

어떠한 방식이던지, 서비스는 최고여야한다. 조직체는 공개소프트웨어를 최상으로 사용하기 위해 전문 서비스와 기술지원에 비용을 지불할 것이다. 이는 법을 최상으로 활용할 수 있도록 변호사 서비스를 필요로 하는 것과 같은 맥락이다. 서비스 제공자를 기업화시키는 것은 공개소스가 ‘당신의 서비스’가 되는 것이다.

Invisible Man: 투명인간

공개소스는 우리를 둘러싸고 있다.

TiVo로 좋아하는 영화를 감상하고 테라스에서 무선으로 인터넷을 하고 운전하면서 MP3 음악을 듣는 것 등 이러한 모든 기술은 ‘under hood’로 공개소프트웨어를 사용한다.

임베디드 공개소프트웨어는 리치 마켓으로 공개소프트웨어 사용을 가속화시킨다. 네트워크 장치, 전자제품 및 전자장치에서 군대용 휴대전화 장치들에까지 리눅스와 내장된 공개소프트웨어로 작동되는 제품의 수는 계속 증가하고 있다.

모든 곳에서 공개소프트웨어 자체로 임베디드됨에 따라 경제학자 브라이언 아더의 비교에 의하면 철도와 같은 정보기술의 성장을 주도하는 엔진을 목격하게 된다. 아더에 따르면, 경제적으로 기술이 그 자체로 깊이있게 내장될 때 이것은 경제성장을 주도하는 엔진이 된다는 것이다. 이렇게 ‘성립된’ 긴 기술은 기술과함께 초기붐과 파산 이후 나타나며(닷컴 버블과 burst를 회상하라) 수십년간 지속할 수 있다²⁰. 성립되는 동안 기술은 삶이라는 섬유의 일부가 되어 보이지조차 않고 그저 작동한다.

근원에서부터

리누스 토발즈가 리눅스 개발을 시작했을 때 리눅스 386PC 이외 다른 것에서 작동될 것이라고는 전혀 기대하지않았다. 그러나 공개소스 커뮤니티의 역동성과 파워는 그의 기대가 잘못되었음을 입증했다. 몇 가지 다양한 프로젝트 이후 최초의 리눅스 커널은 대규모 서버와 웹스테이션에 포팅되기 시작했고 몇몇 개발자들은 이와는 상반되는 방향으로 목표를 지향했는데 이것은 리눅스를 작은 규모의 하드웨어에 적용하는 것이다. ELKS(Embedded Linux Kernel Subset)와 uClinux(micro-controller Linux)와 같은 초기 성공적인 프로젝트에 이어 수많은 무료 임베디드 리눅스 시스템과 상업적인 배포판들이 시장에 등장했다.

오늘날 임베디드 리눅스는 수많은 소모품을 위한 소프트웨어기반으로 제공되며 제조사들의 소

프트웨어개발에 따른 부담을 덜어줬고 신뢰할 수 있는 기능을 제공하며 새로운 차원의 강건함과 개선의 가능성을 더하고 있으며 비용감소에 기여하고 있다. 특별히 소비제와 네트워크제품에 기여하는 바가 크다.

...네트워크 렉에는

firewall, router, modem, 무선 local area network access points, 그 외 네트워크 제품들 덕분에 오늘날 컴퓨터 인프라는 어느 곳에서나 가능케되었다. 눈에 띄지않는 대부분의 블랙박스는 매일매일 운영될 수 있도록 **under hood**된 섬세한 소프트웨어시스템이 필요하다. 따라서 리눅스가 아시아에 대체로 분포하고있는 네트워크 하드웨어 제조사들의 임베디드 운영체제로 왜 리눅스를 가장 선호하는지 이해할만하다. 2003년 10월 EE times에서 실시한 조사에 따르면(아시아 지역 대상으로 가트너실시) 지역에 따라 벤더의 40~50%는 펌웨어 기반으로 임베디드 리눅스를 사용한다고 보고했다²¹.

리눅스가 임베디드소프트웨어 시장에 가져온 대대적인 충격은 임베디드소프트웨어개발을 위한 툴과 플랫폼을 제공하는 선두 제공사인 윈드리버시스템에 의해 충실하게 리눅스를 지원하고 Eclipse에 오픈 IDE를 만들어낼수 있도록 하기까지 강화되었다. 올해 초 윈드리버의 발표에 따르면 레드햇과 공동작업하여 리눅스와 윈드리버 개발툴 기반으로 통합된 임베디드솔루션을 개발하였다.

리눅스를 구성하는 강력한 프레임워크는 네트워크 제조사들이 공개소스 운영체제에 의존하는 유일한 이유는 아니다. 리눅스는 온전한 필터패킷full-fledged packet filter와 firewall시스템과 같이 그 자체가 정교한 네트워킹 능력을 가지고 있기 때문에 네트워크 제품을 구성하기위한 이상적인 기반이라는 것이다.

최종적으로 컴퓨터와 TV를 한곳으로 모으는데 공개소프트웨어는 파격적으로 필요할 것이다.

네트워크제품과 비즈니스폰의 혁신적인 조합은 줄티스(Zultys Technologies)사가 개발한 ZIP 4X4 IP 텔레폰이다. 한눈에보기에 Zip 4x4는 평범한 아날로그 전화이나 여기에 Voice over IP functionality(인터넷폰)이 추가되고 네트워크 스위치, firewall, 인터넷라우터가 내장되어 있는 것이다. 다른 네트워크제품처럼 줄티스사 제품은 임베디드 리눅스기반이다. 아날로그폰과 디지털인터넷의 간격을 연결하였기 때문에 이 제품은 공개소스를 내재시켜 소비제품을 만드는 주요한 일보가 된다. 리눅스와 공개소스를 사용하는 미래의 표준 비즈니스 인프라인 폰분야에 오픈 플랫폼은 기여한다.

...거실 안으로

임베디드운영체제로 리눅스가 사용된 가장 최초 소비제품 중 하나인 TiVo는 TV프로그램을 디지

털로 녹화하는 셋톱박스이다. 1999년에 출시된 TiVo는 최초로 상품화된 디지털 비디오 리코더(DVR)이었다.

그림: Zip4x4는 IP폰을 가진 비즈니스폰이며 리눅스를 포함한 공개기술에 전적으로 기반한다. 이 폰은 SIP(Session Initiation Protocol)을 사용하는 IP전화시스템과 호환되며 이더넷 스위칭, 음성부호화보이스 엔크립션, 4인 동시 통화와 4개의 이더넷포트 등의 첨단 기능을 지원한다.

TiVo사의 창립자 짐 바톤은 상품에 공개소프트웨어를 활용하는 아이디어를 선도하여 GPL라이센스때문에 리눅스가 독점 개발을 허용하지않을 것이라는 라이선스에 대한 두려움을 극복하게 하였다. 2003년 7월 큐(Queue) 매거진의 기사를 통해 바톤은 “GPL를 숙지한 결과 저는 이러한 두려움은 근거없다는 것과 리눅스는 우리의 지적재산권을 보호하는 동시에 강력한 개발 유익을 준다는 점을 발견했다” 회고했다. GPL를 충족시키기위해서 TiVo사는 웹사이트 상에서 변경된 리눅스 커널의 전체 소스코드를 공개했다.

오늘날 TiVo사는 엄청난 성공을 거두어 백삼십만의 사용자를 가진 ‘TiVoolution’을 불붙였다. TiVo 커뮤니티는 인터넷상에서 형성되어 6만 회원과 백8십만을 넘게 포스팅하는 TiVo 커뮤니티 포럼을 만들었다. 이외 알려지지않은 수많은 TiVo 사용자들이 리눅스를 사용한다.

사실, TiVo사는 ‘블랙박스’로 소비자들에게 나타나서 공개소스가 내부에서 작동한다는 것을 은폐하였다. 이와는 상반된 접근방식은 독일 주재 드림멀티미디어TV (Dream-Multimedia-TV GmbH)의 드림박스 DM7000 (Dreambox DM7000)이라는 DVR의 경우이다. 이 업체는 공개소스기반 제품이 소비자뿐만아니라 테키들 말하자면 TV 기기 해커들과 리눅스 프로그래머들 대상으로 하는 잠재적 시장기회를 발견였다.

그림: Dreambox DM7000, 디지털 비디오 리코더로써 리눅스로 운영되며 커뮤니티개발자들을 위해 모두에게 공개된다. 제품이 공개소스접근이기 때문에 공개소스 웹서버와 이메일클라이언트를 포함하여 추가적인 소프트웨어가 가능하다.

“기본적으로, 이 셋톱박스는 모든 이들이 어떤 용도로 이용하든지 자유이나 단 많은 외부 애플리케이션을 사용하지않는 TV 중심 애플리케이션이라는 우리의 표준 내에서 가능하다”라고 LinuxDevices.com의 한 아티클을 통해 DM7000의 개발자 펠릭스 돔케는 말한다. 이러한 방식으로 제품을 공개하는 것은 외부 개발자들이 기능성을 확대할 수 있는 능력을 심각하게 제한시키는 독점 운영시스템에서는 불가능한 것이다. 그러나 오픈플랫폼덕분에 공개소스웹서버와 이메일클라이언트를 포함하여 추가적인 소프트웨어가 드림박스에서는 가능한 것이다. 공개소프트웨어는 결국에 컴퓨팅과 TV를 통합을 위한 돌파구가 될것이다.

...이동하면서 **on the road**

사람들이 음악을 즐길 수 있는 곳이 어디든지 디지털 오디오 파일을 수요가 급증하는 가운데, 1999년에 창립된 팻노이즈사 (PhatNois) 자동차용 디지털 주크박스를 개발하게 된다. 팻노이즈 카오디오시스템 (PhatNois car audio system)은 CD 체인저와는 유사하나 400여개의 음악 CD 분량에 해당하는 상당한 분량의 음악이 가능한 일종의 MP3 플레이어이다. 음악은 PC에서 팻노이즈시스템으로 전환되어 차내 스테레오로 연주된다. 직관적인 파일 관리기능있으며 소비자의 기호에 맞추어 다양하게 연주리스트를 생성할 수 있는 팻노이즈시스템은 리눅스 기반이다.

리눅스를 이용하는 주요 이유 두 가지는 라이선스비용이 없다는 것과 개발이 빠르다는 것이다. 리눅스로 시작하고 사용하는 것은 기획에서부터 전체 솔루션을 새로 개발하는 것에 비해 비용 효과는 엄청나게 크다. “현재 내장된 장치들은 다양한 소프트웨어가 구비되어야 하는데 전통적으로 임베디드 OS는 대부분 충분히 빌트인 되어있지 않다” 고 팻노이즈사의 공동 창립자 겸 CTO인 댄 벤야민은 말한다. “리눅스가 데스크탑 OS로 탑재될 때 보다 풍성한 애플리케이션과 기능 풀로 시작하여 필요한 만큼 구비할 수 있다”

... 당신의 포켓 속에

가정에서나 이동 중에 개인 정보에 접근할 수 있는 것들은 개선되어질 여지가 많다. 작은 장치에 강력한 네트워킹 기능의 탄탄한 OS를 탑재 가능하면서 벤더들이 개인정보접근을 위해 값싼 서버를 구축하게 된다.

D링크시스템사(D-Link System)의 중앙홈작동 (Central Home Drive)는 모뎀사이즈로 5000MP3 파일 혹은 고화질 JPEG 파일 2만개에 이르는 홈네트워크 스토리지 수용량을 지원한다. 인텔의 혁신적인 프로토타입 개인 서버로 지갑사이즈의 이 장치는 무선으로 다른 컴퓨터에 연결되어 파일 접근이 가능하고 유저들은 파일과 데이터를 지갑에 넣어서 어디든지 갈 수 있다. 노트북을 가지고 다닐 필요가 없어진다.

그림: 리눅스는 팻노이즈카오디오시스템이 혁신되고 보다 빠르게 시장에 접근할 수 있도록 기여하여 자동차 스테레오로 수백 장 분량의 음악CD를 비축할 수 있게 했다. 초기 기획부터 전체 솔루션 개발하는 것보다 리눅스로 시작하여 리눅스를 채용하는 것이 확실하게 비용효과가 있다

리눅스에 더불어, 두 제품은 Samba(파일쉐어서버)와 HTTP(파일쉐어프로토콜)을 사용하며 둘다 공개소스이다. 인텔의 개인서버는 배포되는 파일관리를 위해 WebDAV서비스를 개선하기 위하여 아파치를 변형시킨 버전을 사용한다.

PDA 스마트폰에서는, 리눅스가 팜원사 (PalmOne)의 PalmOS, MS의 윈도우 CE, Symbian OS의 대체한다. PDA시장에서 리눅스의 약진은 2001년 샤프사 (Sharp)의 자우러스 (Zaurus) SL5000D에서 공표되는데, 이 제품은 초기 OS로 리눅스를 사용했던 주요 소비자전자제품 제조

사가 개발한 첫 PDA 였다.

2003년에 전 세계 두번째 휴대폰개발사인 모토롤라는 임베디드 리눅스와 Sun의 자바에 기반한 저렴한 모델의 미래형 휴대폰, 스마트폰이 등장한다. 같은 해 모토롤라는 세계 최초 공개소스 스마트폰, 모토롤라 A760을 아태지역에 소개한다. 이 제품은 개인정보관리, 비디오플레이어, 음악 플레이어와 문자메세지 기능을 포함한 고성능 스마트폰이다. 최근 모토롤라에서 전 세계시장을 대상으로 출시한 모토롤라E680™ 은 미국시장에 모토롤라가 출시한 최초의 리눅스기반의 휴대폰이다.

기술혁신과 신속한 개발은 핵심 기술로써 리눅스와 자바 지원으로의 움직임에 박차를 가하였다. 리눅스와 자바 기술을 주도하는 모토롤라는 개발자들이 게임에서부터 생산물에 이르기까지 새로운 애플리케이션을 자유롭게 개발해낼수 있도록 한다. 독점 구도에서보다 협력하는 공개소스 커뮤니티가 더 빠르게 개발을 진행할 수 있는 것은 기존 공개소스코드를 활용할 수 있기 때문이다.

비용 역시 한 요소였다. 라이선스비가 없다는 것은 소비제품의 대량 생산에 상당히 매력적인 것이다. 실제로 독점 라이선스비는 전체 비용에서 압도적으로 큰 포션을 차지한다.

모토롤라에 이어 주요 장치제조사인 삼성, NEC은 리눅스기반 휴대폰을 계획 중이다. 장치 제조사와 휴대폰 사용자들에게 애플리케이션과 인터페이스 소프트웨어를 제공하며 오늘날 모든 데이터관리가 되는 휴대폰의 50%이상에 사용되는 소프트웨어를 생산하는 오픈웨이브시스템사 역시 리눅스 버전의 소프트웨어를 발표했다.

...전장에서

포켓 속이나 거실에서 사용되는 장치 내부에는 리눅스가 내장있을 것이나 군용품에서 역시 앞단에서 등장한다. Land warrior로 불리는 미군의 정교한 착용가능한 컴퓨터에는 리눅스가 내장되어 안전한 무선 통신과 land warrior를 이용하는 군인들의 기타 활동을 지원하고 있다.

일반적으로 군대, 현장 및 이동작업자들을 위해 전문화된 애플리케이션 등 착용품과 휴대용품 시장은 성장일로에 있다. 의복 소프트웨어는 평형상태가 중요한 대목이다. 채택가능성, 모듈화 방식, 전력 효율성과 시장진입시간은 휴대용 애플리케이션에서 임베디드 리눅스를 사용하는 이유들이다.

미군에서는 보안과 그 외 여타 이유로 인하여 Land Warrior System를 개발하는 도중에 윈도우 플랫폼에서 리눅스로 전향하였다. Land Warrior System의 주계약자인 General Dynamics사와 작업을 위해 CSC는 맞춤형 OS와 유저 애플리케이션 배송을 담당한다.

...누가 어디인지를 아는가?

미디어분야에서 공개소스의 뛰어난 경력은 궁극적으로 공개소스 소비화에서 최소한 한가지 신기한 특징을 주도한다. 월마트 (Walmart) 독일지사에서는 'micro&soft'라는 섬유유연제와 더불어 'Linux'라는 새로운 세제를 마케팅하고 있다. 둘 다 스위스의 Rosch사 제품이다. 이 두 제품은 상당히 호환성이 높다.

소프트웨어분야로 다시 돌아와서, 'under hood' 공개소스전략은 제품개발을 위한 견고한 상품의 플랫폼과 소프트웨어를 자유롭게 보완하고 확장할 수 있게 하며 시장진입 시간을 단축하여 전체 생산비용을 경감한다.

상품 플랫폼은 혁신적이고 차별화된 애플리케이션을 개발하게 한다. 영화, 음악, 게임, 문자서비스 등 바로 소비자를 끌어잡을 수 있게 하는 것이다. 소비자에게는 공개소스가 'under hood'되는 것은 그리 중요한 것이 아니라 제품의 기능이 중요한 것이다.

시장의 위력:

공개소스는 경쟁을 증가시키며 기존 시장의 힘에 도전한다.

‘오픈소스는 단순한 소스코드가 아닌 시장의 위력이다’-forrester research ²⁴

오픈소스는 소프트웨어시장의 경쟁을 촉진한다. 독점제품과의 직접 경쟁, 코페티션(co-opetition, 협력-경쟁), 협회, 정부주도, 상품화, 심지어는 오픈소스제품간에서의 경쟁 등 여러 형태이다. 결과는, 오픈소스는 경쟁을 일으키고 자신들의 제품을 방어하려는 기존 시장세력들에 도전하게 된다. 이 이상 아무것도 아니라면 상업적인 소프트웨어라이선스비를 낮추기 위한 협상 톨로 기능한다.

과거에는, 기존 업체들과 기성 시장세력을 흔들어보려는 신생업체간의 경쟁이었다. 오늘날 오픈소스커뮤니티는 강력하지만않다면 전통적인 경쟁자와 동등하게 되는 실용적인 경쟁의 소스를 보여준다. 근육을 유연하게 함으로 오픈소스는 경쟁구도에서 모든 종류의 트위스와 뒤집기를 일으킨다.

MS 역시 오픈소스에 대해 탐색하기 시작했다. MS는 Window Template library의 소스코드를 오픈소스처럼 공개했다. 이 소스코드는 윈도우 애플리케이션과 유저 인터페이스 컴포넌트 개발하는데 사용되는 것이며 윈도우 인스톨러 XML의 코드는 공개된 후 몇 주동안 8만회이상 다운로드되었다. MS사의 이러한 움직임은 소프트웨어 거대비즈니스가 비공개되는 소스라는 패러다임에 주목할 만한 것이다.

Head to Head

머리 대 머리로

MS윈도우를 OS로 선택하는 것에 리눅스가 정면 도전할 것이라는 것은 더 이상 비밀이 아니다. 리눅스의 도전은 서버에서 시작하여 이제는 데스크탑으로 옮겨가고 있다. 리눅스는 전세계 PC 시장의 90%이상을 차지하는 윈도우에 압력을 가하고 있다(그를 이어MacOS는 대략 2%) 이것은 다윗가 골리앗 이야기이다. 그러나 다윗은 점점 더 강력해지고 있으며 리눅스가 2004년에는 제 2위를 차지할 것으로 분석가들은 전망한다.

데스크탑OS와 오피스스위트 양 부분에서 MS를 대체할 오픈소스는 경쟁자 MS를 향해 경주를 선포하고 있다. IBM, HP, Sun, Red Hat, Novell 등의 업체들은 데스크탑 리눅스를 공식적으로 지지하기 시작했다. SuSE Linux를 인수한 Novell은 리눅스의 시장위력을 선언하고 MS에게서 돌아서는 전사적인 움직임으로 전향했다.

Sun의 스타오피스에 기반한 오픈오피스는 MS오피스와 경합을 하고 있다. 2000년에 첫 출시된 오픈오피스는 대규모 기업오피스시스템시장에서 14%의 마켓쉐어를 갖는다. Forrester Research사에 따르면 오픈오피스는 전체 오피스 시장의 94% 마켓쉐어를 보유하고 10여년동안 시장을 점유하고 전 세계적으로 3억명의 유저를 확보한 MS오피스를 실질적으로 대체해가고 있다는 것이다. 오픈소스방식을 시큰둥하게 여겼던 MS이지만 위협을 인식하기시작했다. MS사는 오픈오피스에 비교하여 MS오피스의 세일즈 포스를 위한 경쟁력있는 가이드를 출판하였다(이것은 웹사이트에서 바로 제거되었다) 경합이든 혹은 실제 포팅이든지 간에 IBM이 MS오피스에서 리눅스로 전향한 내용조차 언급되었다.

브라우저 전쟁 역시 오픈소스로의 전략적 이동에 불을 붙였다. 1998년 넷스케이프는 MS의 인터넷 익스플로러에 대한 마지막 움직임으로 오픈소스 모질라로 브라우저를 출시했다. 넷스케이프는 소스코드를 전 세계 개발커뮤니티에게 공개하여 한때 깃발을 날리던 브라우저에 대한 지원을 얻기를 기대했다. 비록 이러한 시도가 넷스케이프를 유지시키는데 그다지 기여하지는 않았지만, 결과적으로 2002년 새로운 오픈소스 브라우저가 생산되었다. 최근, 모질라 화이어폭스로 명명되는 브라우저는 혁신적인 기능을 구비하고 있는 경쟁력있는 브라우저로서 추진력을 얻는다. 흥미로운 움직임으로, 노키아는 휴대폰 브라우저를 개발하기 위하여 모질라 파운데이션을 설립한 것이다. 이로 인해 화이어폭스의 명성은 높아졌고 이런 추세라면 브라우저 세계에서 힘의 균형은 이동할 것이다.

코페티션 (협력-경쟁)

오픈소스는 협력이라는 천성에 따라 경쟁자를 동일한 방향으로 끌어당긴다. 오픈어답터(Openadaptor)는 협력-경쟁의 사례로써 값비싼 독점 소프트웨어를 대체할 소프트웨어 개발에 경쟁자들이 공통 코드로 사용하여 작업하는 것이다. 오픈어답터 오픈소스프로젝트는 드레스

덴 은행의 투자 파트너 Dresdner Kleinwort Wasserstein(DrKW), 그 외 동일한 계열의 독일 은행들 등 재정단체들의 지원을 받는다. 고도로 경쟁력있는 은행산업에서 주목할만한 일이다. 이 은행들은 고가의 기업 애플리케이션 통합 툴을 오픈소스로 대체하기 위하여 오픈어댑터와 작업을 추진하고 있다.

DrKW는 코드를 개발했고 2001년 오픈소스형태로 출시했다. 그 외 은행들은 그 코드를 채용하고 확장하였고 오픈소스커뮤니티에 다시 공개했다. 그리하여 경쟁자들은 본질적으로 다른 시스템 통합을 위한 더 좋고 더 싼 소프트웨어 툴을 제공하기 위하여 나란히 같이 작업하고 있다.

DrKW가 오픈소스로 전향하게 된 동기는 경쟁사를 포함하여 보다 큰 커뮤니티를 끌어들이 기술 혁신을 증대시키자는 것이다. 애플리케이션 통합은 라이선스비용만 백만달러를 넘는다. 오픈소스 채용은 라이선스 비용이 없다는 것, 폭넓은 커뮤니티의 전문기술, 특정 벤더에 묶어두려하지 않고, 궁극적으로는 오픈어댑터에의해 제공된 통합 덕분에 전자적으로 비즈니스를 보다 쉽게 하여 비즈니스를 증대시켰다.

비록 오픈어댑터가 상업적인 EAI 제공의 모든 기능을 제시하지는 않았으나 어떤 조직들에게는 대안이 제시된 것이고 이는 협력-경쟁을 위한 촉매가 되었다.

콘소시엄

IBM이 발족하였으나 현재는 독립적인 에클립스콘소시엄은 애플리케이션 개발을 위해 사용되는 통합된 개발환경인 에클립스 플랫폼을 위한 제품과 표준을 개발한다. 오픈소스 프로젝트의 하나로써 에클립스는 기성 IDE 벤더들(JBuilder IDE와 Borland Software)에게 가공할만한 도전을 주기위해 50개의 벤더들을 포함시켰다.

에클립스는 자바를 위한 VisualAge의 다음 버전으로써 IBM에의해 시작되었다. IBM은 2001년에 오픈소스로 코드를 출시하였고 수년 동안 에클립스 콘소시엄을 주도하였으며 오픈소스프로젝트로 발아시키기위해 4천만달러를 투자하였다. 콘소시엄에 대한 IBM의 영향은 의문을 가질 만하지만 -올해 IBM에의해 콘소시엄은 decoupled되었다- 에클립스 플랫폼은 IDEs를 위한 표준 설정을 대중적으로 대체하였다. 플랫폼은 대략 매일 3만회의 다운로드 요청을 받았으며 6백 개 이상의 오픈소스 혹은 프리웨어 플러그인을 자랑하고 있다.

그러나 에클립스는 단지 오픈소스 IDE가 아니라, Sun에의해 오픈소스가 된 넷(NetBeans)과 결합한다. (Sun은 Sun ONE 개발스튜디오로써 넷빈의 상업적 버전을 마케팅한다)

넷빈은 에클립스에게 기반을 빼앗겼고 양자간의 경쟁구도를 강화시켰다. 최근, Sun은 에클립스 프로젝트에 조인할 것을 요청을 받았으나 거절하였다. 이유는 아마도 Sun의 상품 기반이 되는

넷빈을 포기하고싶지않거나 합병하기를 원치않았던 까닭일 것이다.

62P

넷빈은 독립적인 플러그인 세트를 가진 핵심 시스템의 매우 성공적인 개발 모델을 개발한 최초의 자바 IDE이다. 이 방식은 오픈소스로 배포된 제대로 개발된 모델이며 이후 에클립스 프로젝트를 위해 IBM이 채택했다.

에클립스 콘소시엄은 IDE 환경에서 중요한 역할을 한다. 비록 산업의 강력한 설립자가 있기는 했지만 시장에서 표준을 설정하고 강력한 세력이 되는 오픈소스 산업 콘소시엄의 위력을 보여주었다. 에클립스 플랫폼은 IBM의 웹스피어스튜디오애플리케이션개발자 (WebSphere Studio Application Developer)와 SAP의 넷위버개발자스튜디오 (NetWeaver Developer Studio)를 포함하여 새로운 경쟁 제품들을 등장을 촉발하였다.

부분적으로 에클립스와 넷빈과의 경합을 겨냥하면서 공개소프트웨어로된 웹로지(WebLogic) 프로그램 프레임워크 BEA가 등장한다. BEA가 기대하는 것은, 자바개발자 수요를 증대시켜서 BEA플랫폼만이 아닌 어떠한 플랫폼이든지 애플리케이션 탑재가 가능한 오픈소스 운영 환경 가운데 모든 자바 플랫폼을 위한 제품을 구축할 수 있도록 독려하는 것이다. 이러한 움직임은 자바 애플리케이션 서버 시장에서 하강추세인 BEA 포지션을 지지하려는 것이다. 자바 작동을 용이하게 하는 것이라면 어떤 것이든지 자바 에코시스템이 성장의 열쇠가 된다(그리고 MS에 대응하고)

정부주도

선정과정에서 공개소프트웨어를 검토하도록 정책화 시도 등 공개소프트웨어에 관해 다양한 정부주도 노력이 있다.

정부시스템에는 반드시 공개소프트웨어를 사용해야만 하고 만일 오픈소스를 사용하지않을 경우 공개소프트웨어가 시스템에 맞지않는 이유를 설명해야한다는 매사추세츠 주정부의 지시는 독점SW벤더들에게 새로운 압력을 가했다. 벤더들은 제품가격을 언급하기보다는 제품을 정당화하고 가능한 오픈소스를 지원해야만 한다.

주 보건당국의 경우, 시스템이 Jboss기반으로 개발되었으나 제품은 독점제품 환경에 포팅되었다. 이러한 접근을 검토하기위해 당국이 소집한 세개의 벤더들에의하면 JBoss는 생산대기만이 아니라 개발대기로 간주되었다. 벤더들은 JBoss사의 실적과 어느 정도까지 검증되었는지에 관해 의문을 제기했다. 당국은 생산 환경에서는 JBoss를 사용하지말라는 벤더들의 권유에 동의했다.

이 결과, 생산을 위한 독점적 플랫폼에 포트되는 것은 힘들게 되었고 개발측면에서 얻어진 것들도 수포로 돌아갔다. 이런 유사한 상황에 빠진 조직들은 결과적으로 JBoss를 생산에서도 사용하게 되었다.

벤더들은 오픈소스 불가항력을 무시할 수 없을뿐 아니라 오픈소스와 조화를 이루어야 할 필요가 있는 것이다. 오픈소스를 고집하는 프레임워크 하이버네이트(Hibernate)를 살펴보자. Jbaoss는 하이버네이트와 제대로 통합되는 반면 다른 벤더 플랫폼과는 그렇지 않다. 대신에 다른 플랫폼은 관행적인 코드를 요구한다. 이것이 바로 보건당국이 당면하고 있는 포팅이 난이하게 되는 한 요소이다.

정부주도의 오픈소스 추진으로, 벤더들은 그들 제품이 얼마나 차별화되었는지, 조직이 그들의 제품의 특별한 기능을 왜 요구하는지에 대해 설명하도록 요청받고 있다. 따라서 소프트웨어 진단은 더욱더 세분화되고 현재 요구되는 특정 기능에 기초하여 이뤄지고 있다. 향후 필요하게 될 기능에 대한 세일즈 피치는 이전만큼 영향을 줄 것으로 보이지 않는다. 벤더들은 오픈소스가 제안하는 것과 요구에 대해 대결을 벌이기 위해서 새로운 대안을 찾아야 할 필요가 있다.

상품화

공개소프트웨어는 소프트웨어스택에 상품화를 가져왔는데 이는 가장 낮은 단계에서 출발하는 것이다. 이론적으로 시간이 지남에 따라 경쟁자들은 정리된다(이러한 방식으로 IBM PC의 오픈 하드웨어플랫폼은 Data General, Digital, Prime등의 독점 하드웨어벤더들을 정리했다) 공개 소프트웨어제품들은 상용품이 되어 독점소프트웨어가 경쟁을 벌일 수 없는 무가시장을 형성하였다.

3가지의 오픈소스 유틸리티로는, Log4J, Tomcat, Struts가 있다. 이들은 대형 기업 애플리케이션에서 사용되고 있으며 실제 상업적인 경쟁관계는 아니다. 이들 유틸리티에 우선하여 개발자들은 맞춤코드를 개발함으로써 시장을 점유할 수 있게 되었다. (Log4J는 로깅 서비스이고 Tomcat은 Java Servlet과 Java 서버페이지를 위한 Java '컨테이너' 혹은 프로세서이며, Struts는 Java 웹애플리케이션 구축을 위한 프레임워크이다) 이 모두는 기업맞춤형자바애플리케이션 개발을 위한 오픈소스프로젝트인 아파치 자카르타 프로젝트의 일부이다.

상용 소프트웨어인 Log4J, tomcat, Struts는 마켓진입에서부터 개연성있는 경쟁자들을 대체하였고 유저들은 필요한 소프트웨어를 얻었다. 이들 오픈소스제품들은 유저들에게는 너무나 만족스러운 것들이다.

기반으로서의 상용소프트웨어와 더불어 차별화는 이 분야를 풍성하게 하였다. 에클립스 플랫폼은 필수적으로 상용 플랫폼이고 이를 바탕으로 벤더들은 제품을 구축한다. CSC는 새로운 금융

및 보험상품을 개발하기 위하여 금융서비스산업에 맞춤형 IDE인 VP / MS Model Manager를 개발했다. VP / MS Model Manager는 CSC의 비주얼 제품 모델링 시스템인 VP / MS의 일부이다. VP / MS는 CSC의 수많은 금융 및 보험서비스제품 속에 통합되었다.

VP / MS와 VP / MS Model Manager를 사용하여 독일의 시그널이두나 (Signal Iduna) 등의 보험사들은 이전에 비교하여 놀랍도록 짧은 시간에 새 제품 기획부터 마케팅까지 실현해 낼 수 있게 되었다.

모델 매니저용으로 상용 오픈소스 플랫폼을 사용하면서, CSC는 바퀴를 새로 개발하기보다는 맞춤개선에 주력할 수 있게 되었다. “오픈소스를 사용함으로 고객들은 보다 양질의 제품을 가질 수 있게 되었다”고 독일 콜로뉴의 CSC 금융서비스그룹의 소프트웨어 설계자인 롤프 윌름스 (Rolf Wilms)는 말한다. “에클립스사가 제공한 확장가능한 플러그인 아키텍처로 말미암아 VP / MS 제품군을 위한 기술지원을 증대키시킬 수 있게 되었다”

VP / MS Model Manager로 CSC는 오픈소스커뮤니티로부터 단지 얻기만할 뿐아니라 커뮤니티에 돌려줄 수도 있게 되었다. CSC는 개발된 모듈 관리(configuration management)와 Editor(detecting conflicts before they occur)등 두 개의 모듈을 에클립스 커뮤니티에게 제공하였다.

“우리는 에클립스없이 VP / MS Model Manager를 개발해 낼 수 없었다. 아마도 우리는 아주 비싼 비용을 들여서 전체 IDE부터 개발했어야만 했을 것이다. 우리가 이들 커뮤니티에게 뭔가를 돌려주는 것은 당연하다.” 라고 말하며 설명하기를 “우리는 CSC에 대해 경쟁 효과로서 아닌 커뮤니티에게 기술적 가치를 기부하고 싶었다”

상용 IDE는 CSC 클라이언트와 공개소스커뮤니티 모두를 위한 기술혁신에 박차를 가했다.

이것은 시장을 위해서도 좋은 소식이지만 상품화(와 경쟁적 효과)는 하룻밤새 이뤄진 것이 아니라는 점을 기억해야 한다. 예들 들어, 상품화는 데이터베이스 분야를 위협하지만 이 분야를 차지한 것은 아니다. MySQL이 2003년 후반에 30%이상 성장하여 시장을 잠식하기 시작했고 MaxDB는 SAP가 MySQL과 연대하여 오픈소스로 데이터관리시스템을 출시하려는 노력의 결과이다. 오픈소스데이타의 현실은 상용소프트웨어에 있어서 데이터베이스 혁명을 가르킨다.

그럼에도 불구하고, 대부분의 기업들이 한동안 독점적 데이터베이스(오라클, IBM)사용을 지속할 것이기 때문에 오픈소스데이타베이스의 세일즈는 힘들다고 볼 수 있다. 기업들은 DB에 투자된 그들만의 비즈니스가 있으며 아직은 상용소프트웨어로 간주하지는 않는다. 전환비용은 데이터 전환면에서 상당히 높고 여러 기업들은 데이터베이스 라이선스비용에 만족해하고 있고 무료 오

오픈소스라이선스 비용의 효과를 인정하지않고 있다. 그리하여, DB분야는 상품화로 이동하는 경향을 발견하게 되고 고성능 DB에서 오픈소스를 채택하는 것은 다른 OS와 개발 툴보다 진행 속도가 더딜 것이다.

오픈소스DB는 기업 끝단의 리포팅 업무에는 상당히 매력적이다. 여기서는 기술이 '좋으면 그만이다'이기 때문이다. 오픈소스가 끝단과 아랫단에 보다 많이 탑재될수록 오픈소스DB는 핵심 비즈니스로 부상할 것이다. 어떻게 진행될 지 지켜보자.

표준

표준은 경쟁의 본질을 변화시킨다. 표준은 벤더들이 가장 큰 시장을 사로잡기위해 반드시 확인해야만 하는 것이다.(비록 비평가들이 표준이 혁신을 가로막는다고 논쟁할 것이지만) 가치는 표준의 제일 위에 올려지는 것이다. 한번 더 언급하면 상품화될수록 차별화는 상품을 다양하게 한다.

오픈소스커뮤니티는 표준개발을 가속화시킬 수 있다. 세상이 점점 빨라질수록, 전 세계적으로 협력네트워크된 오픈소스커뮤니티는 전통적인 표준작업조직보다 신속하게 표준관련 이슈들을 다룰 수 있는 잠재력을 보유한다.

표준을 만들어내는 오픈소스커뮤니티의 예는 하이버네이트(Hibernate)이다. 하이버네이트 프로젝트는 가빈 킹이라는 호주 학생이 2001년에 착수한 것으로 중/소 애플리케이션에서 객체관련매핑툴(object-relational mapping tool)을 위한 사실상의 표준이 되고 있다. 또 다른 예로 JDO(Java Data Object)는 그 당시 개발 중이었으나 하이버네이트 앞에서 가까스로 버티고 있다.

하이버네이트는 우수하고 가벼운 툴이다. 단도직입적으로 말해서, 사용이 쉽고 뛰어난 수행력을 나타낸다. 처음 개발되었을 때 이것은 표준은 아니었으나 사용 인구가 많은 까닭에 중소조직을 위한 사실상의 표준이 되었다.

이에 비해 JDO는 사용이 복잡하다. 비록 대형 벤더들이 JDO기반제품들을 개발하고 있지만 대중적으로 사용되지 않으며 종종 이 제품은 단지 부분적으로 표준을 개선한다. 수많은 사람들이 JDO개발에 참여하였으며 3년에 걸쳐 진행되어 2002년에 첫 제품이 출시된다. 그 당시 표준은 미완결된 상태였고 아직도 완성되지못했다. 하이버네이트가 잘 나가는 반면 JDO는 고전을 면치 못했다. 그러는 와중에도 JDO 2.0 출시의 기미가 보이다가 제법 기반을 닦고 있다. 오라클의 탑링크(TopLink)와 하이버네이트는 이를 지원한다. 사실, 가빈 킹은 JDO표준body 소속이다.

최근 기업자바빈(Enterprise Java Beans)(EJB)사양(EJB는 J2EE사양의 대부분 혹은 일부를 구성)

의 다음 버전을 구성 중인 EJB3.0 Expert Group은 현재까지 고집해온 모델로 JDO쪽보다는 하이버네이트쪽에 기반한 새 모델로 전환할 것을 결정했다. 이것이 말해주는 것은 오픈소스프로젝트의 민첩하고 실용적인 접근방식은 제도권적 표준정의보다 강력하다는 것이다.

오픈소스커뮤니티는 21세기의 표준에 도달하는 새로운 방안인가? 시간이 설명해 줄 것이다. IBM은 오픈소스자바에 관하여 Sun을 압박하고 있으며 이것은 독립커뮤니티가 산업표준이 될 기술을 한층 더 개발하도록 하기위함이다. 최소한, 오픈소스커뮤니티는 표준개발의 대안적인 접근이고 경쟁력있는 결과들이 나올 것이다.

생존

오픈소스의 경쟁력은 투명한 생존이다.

공개소프트웨어에 과장된 내용에는 모든 공개소프트웨어는 우수하다는 것이다. 상업적 소프트웨어에서처럼 단지 선택된 몇몇 제품만이 정말 우수하고 널리 사용된다. 소스포즈(Sourceforge)에는 8만2천개가 넘는 오픈소스 프로젝트가 있으나 이중 80~90%는 저품질이거나 사용성이 낮다.

오픈소스프로젝트는 품질과 기능에 기반하지 마케팅 과장에 기반하여 생존하는 것이 아니다. 리눅스와 같이 최고의 제품은 생존하나 MySQL같이 ‘충분히 우수한’ 제품 역시 생존한다.

오픈소스분야 내에서 생존은 서로 먹고 먹히는 것이지도 않다. 오픈소스 내 최대의 라이벌인 데스크탑환경의 KDE와 GNOME 이 둘은 공생하는 것 같다.

오픈소스는 경쟁이 힘이고 오픈소스분야 내 힘의 균형을 재분배하고 보다 중요한 점은 소프트웨어시장을 하나로 본다. 오픈소스커뮤니티는 놀라운 경쟁자이고 자원자의 협력체이나 이러한 움직임 뒤에는 품질과 열정이 대형마켓주자들과 맞서고 있다. 공공분야 관리를 위한 스웨덴 예 이전시의 2003년 연구에 의하면 ‘자유 그리고 공개소프트웨어는 미봉책적인 현상이 아니라 기존의 독점 제품과 솔루션에 확실히 적절하고 의존적인 경쟁자이다’ 오픈소스는 소프트웨어 소비자이든 제품이든지 조직들이 인식하고 활성화시킬 필요가 있는 경쟁력이다.

새로운 영역

공개소스는 수많은 영역 안에 살아있다

인류의 모든 두뇌 능력에 접근하고 싶어하는 것은 아주 오래된 염원 중의 하나이다. 1937년 영국 작가 H.G.Wells는 ‘영구적인 세계 백과사전’에 관한 한 아티클에서 ‘엄청나게 많은 사람들이 인간 지식목록을 완성하고 업데이트하는데 매달릴 것이다’²⁶ 예시했다. 오늘날 정확하게 위키페

이아(Wikipedia)에 관한 것처럼 웰스의 비전은 예언적이었다.

위키페디아는 누구나 기여할 수 있는 웹상의 전세계 공개소스 백과사전이다. “무료 백과사전”으로 위키페디아는 2001년 시작되어 오늘날 2십3만 아티클들이 영문버전으로 실려있다. 오픈소스적인 접근방식은 위키페디아에서 가장 정확하고 최신 정보를 보유할수 있도록한다. 어느 누구나 정보를 제공할 수 있고 어느 누구나 정보에 접근할 수 있다.

2004년 디지털 커뮤니티 대상 Prix Ars Electronica 의 최고상을 수상한 위키페디아는 오픈소스적 접근이 소프트웨어개발만이 아니라 수많은 영역에서 살아있고 ‘소스’에관해 보다 넓은 의미를 제공하고 있다.

오픈소스운동의 일반적인 특징은 1)무료로 가능한 정보 2) 협력하여 문제 해결이라는 것이다. 대체로 여기서의 문제는 복잡하고 공공의 천성을 말한다. 종종 오픈소스적 접근은 공급(자발적인 전문가 커뮤니티)와 수요(해결되어야 할 문제)를 연결하는 방법이다.

물론 커뮤니티는 질병 해결이나, 제품을 발명하거나, 범죄자를 검거하거나 오랜 기간동안 문제를 해결해왔다. 오늘날 오픈소스운동을 차별화시키는 점은 인터넷으로 놀랍도록 쉽고 빠르게 커뮤니티에 연결되고 문제에 연관된 핵심 정보에 접근할 수 있다는 것이다.

인간게놈프로젝트는 생물학이라는 다른 영역에서 오픈소스를 대표하는 잘 알려진 예이다. 전 세계의 연구자들은 인간 DNA 해석하기위해 함께 작업하고 그 결과는 모두가 자유롭게 이용할 수 있다. 덜알려진 예로는 오픈로(Openlaw)가 있으며 이것은 오픈소스스타일의 공공 포럼 내에서 법적 논쟁을 창출하는 하버드 로스쿨의 실험이다. 이것은 리갈브리프온라인(legal brief online)에서 편집과 커멘트가 가능하다. 오픈소스의 장점은 하드웨어와 상품 디자인, 농업, 과학, 음악과 협력 지식등을 포함하여 소프트웨어 개발과는 동떨어진 영역에서 작업이 가능하게 한다.

하드웨어와 제품 디자인

씽크사이클(ThinkCycle)은 under-served 커뮤니티와 환경을 위한 제품과 서비스 설계에 관한 협력적 주도체이다. 오픈소스식 접근에 기반하여 설립된 Thinkcycle은 보건, 에너지, 교육 및 지속적인 living(농업, 물, 주택) 분야의 중요한 문제들을 다루기위한 지식 공유와 최신 기술 개발목 적의 세계적인 커뮤니티를 만들어내었다. 이 프로젝트는 저가 안경, 저가 콜레라처치장치, 동력 생산(батери대신에), 미숙아를 위한 특수 인큐베이터, 식수처리시스템등을 포함한다.

MIT Lab이 개발 운영하는 ThinkCycle Initiative는 대학, NGO, 기업, 지역 커뮤니티, 공공분양의 설계사, 엔지니어, 전문가들이 참여한다. 이러한 그룹들이 기여하고 접근할 수 있는 문제들과 솔

루션을 위한 데이터베이스를 제공하려는 목적을 가지고 있다.

컴퓨터시장에서, OpenCores.org는 오픈소스하드웨어프로젝트를 위한 포털로 소프트웨어프로젝트인을 위한 SourceForge와 유사하다. OpenCores는 공개적으로 개발된 AISCs과 연관된 프로젝트에 집중한다. 엔지니어들의 협력을 통해 재사용하는 핵심 설계는 ASIC설계를 가속화하는 등 ASIC 설계부분에 상당한 노력을 기울인다.

OpenCores의 미션은 하드웨어커뮤니티가 소프트웨어에 대한 Lesser General Public License에 기반한 하드웨어모델을 위해 라이선스를 가진 핵심 설계를 개발하고 출시할 수 있도록 지원하는 것이다.(법과 비즈니스 이슈 참조) 무료이며 자유롭게 사용, 재사용할 수 있는 핵심 설계를 하는 것이 기본 의도이다.

시장에서 대한 혜택은 기술혁신으로 어떤 것이던지 크리에이티브를 자극하는 것(핵심 설계의 재사용과 같은)이 열쇠다. IBM은 Chip 설계를 공개하기 시작했는데, 칩설계와 제조기술에서 선두를 달리는 이 기업의 이러한 움직임은 상당히 의미가 깊다. 미래기술혁신이 가속화되기보다는 고객맞춤에서 이뤄진다고 기대되는 분야에서 IBM은 칩기술을 광범위한 커뮤니티에 조금씩 공개함으로써 기술혁신의 씨를 뿌릴수 있기를 희망한다.

IBM과 리눅스에서 얻은 교훈에 이어 IBM은 마이크로프로세서의 power family에 관한 기술정보를 보다 많이 공유하며 칩설계와 테스트를 위한 무료소프트웨어 툴을 배포하고 구매자들이 자신들의 맞춤 칩 개발을 지원해줄 설계센터를 전세계에 설치하고자 한다.

인텔 역시 칩설계분야에서 앞서기위해 오픈소스를 찾고 있다. 인텔의 오픈소스머신 러닝 라이브러리(OpenML)를 통해 인텔은 실시간 머신러닝의 신생분야에 대해서 배우기위해 대규모 커뮤니티로부터 정보를 모을 수 있게 된다. 인텔이 머신러닝과 이것이 어떻게 사용되는지에 대해서 대해 보다 더 잘 이해할수록 이 머신을 용이하게 할 칩개발에 더욱 도움이 된다. 오픈소스는 연구기반을 확장하고 머신러닝에서의 협력과 혁신을 조장하여 궁극적으로 칩설계 개선에 기여한다.

기술혁신을 위한 오픈소스 주제는 대형빌딩에까지 뻗친다. MIT의 오픈소스빌딩연대는 더 나은 건축자재, 기술, 애플리케이션 서비스를 주도할 새로운 전략을 개발과 테스트하는데 주력하고 있다. 초기부터 이 프로젝트는 싱글 거주자와 대가족 거주자를 고려하고 있다.

빌딩 분야에 공개SW테크닉을 채택하기위해 의식적으로 노력해왔다. OSBA 안내서에 따르면 오픈소스전략은 품질, 가치, 전자에서 자동차설계까지 제품의 다양성등을 극적으로 개선시켰왔다. 대조적으로, 대부분의 새집, 아파트 등은 범용, 낮은 등급에 고가이다. OSBA는 설계, 데이터, 전

자, 소프트웨어, 물리적 컴포넌트 연결 등이 모듈로 구성된 산업이 관계된 공개소스웨어 창의적 활동을 폭주시켜 고효율 실적, 비용효과환경을 이끌어낼 것이라고 제안한다. 이러한 접근법은 기술혁신의 장애요인을 제거하는데 필요하며, 에너지보존, 적극적으로 가정내 예방건강케어, 새로운 형식의 작업, 학습, 오락, 그리고 고도로 개인적인 주거환경의 대량 맞춤등에 연관된 흥미로운 기회들을 발굴해낼 것이다.

OSBA는 아이디어 생성, 기술평가와 공공의 코멘트를 위한 웹사이트와 더불어 오픈소스조직을 운영하고 있다. 이 조직은 상업적 환경에서 제공될만한 스트 베드, 데몬스트레이션 환경을 조성할 것이다.

현재 고려중인 컨셉은 모듈식 부분, 벽과 전원연결 그리고 환경감지 와 활동 인식 및 커뮤니케이션 대체할 기술을 대체할 설정 가능한 컴포넌트로 이뤄진다.

농업

주거에 이어, 공개소스기술은 또다른 기본적인 니즈, 음식에 적용될 수 있다. Cambia(Center for the Application of Molecular Biology to International Agriculture)는 sustainable 농업, 강력한 종자개발 그리고 식물재배자와 전 세계 과학자들에게 관습적인 식물, 동물 사육에 새로운 방향을 제시해줄 정책 모음에 관한 연구를 진행하고 있다. Cambia의 목표는 지역 커뮤니티 자체의 농업 및 환경 문제들을 자체적으로 해결하되 현대 유전학의 충분한 가능성을 좌절시키는 지적 재산의 봉쇄를 풀어내는 것이 필요하다.

호주 캔버라에 근거지를 둔 캬비아는 농부에서부터 연구자들에 이르기까지 모든 사람들이 최신 BT 툴에 접근할 수 있어야한다고 믿는다. BT분야가 대기업에 의해 좌지우지되어선 절대 안되는 것이다. “유전학의 툴들과 현대 생물학은 컴퓨터프로그램 툴들이 공개SW운동에서 공유되는 것처럼 자유롭게 이용가능해야 한다.”라고 캬비아의 설립자 Richard Jefferson은 ABC Science Online에서 이야기한다.

캬비아는 열린사회를 위한 생물학적 혁신이라는 새로운 프로그램을 창안했다. 이것은 전형적으로 제한적인 특허를 수반하는 전세계 BT 커뮤니티들이 자유롭게 접근할 수 있도록 지원하는데 목적을 둔다. 제퍼슨은 “우리는 농작물을 개선하여 세계 인구를 먹여 살릴 현대 유전학의 잠재력을 이제 겨우 활용하기 시작했다. 아이로니컬하게도 열쇠는 인간의 창의력과 사용 기술 그리고 모드 사람들 안에 내재된 문제 해결 능력을 양성시킬 현명한 정책이라는 것이다.”이라고 보도를 통해 밝혔다.

오픈소스는 캬비아 미션의 중심이고 Open-access Toolkits공개-접근 툴킷은 BT산업에서 공개운동을 싹틔운 핵심 추동으로서 리눅스에 견주된다. 캬비아는 공개소스의 모든 측면에서 영감

을 얻는다. 제퍼슨이 호주 BT뉴스(Australian Biotechnology News)에서 말한 것처럼 “ GPL같이 규범으로 구속한 메커니즘을 개발함으로써, 우리는 오픈소스개념을 BT커뮤니티를 하나의 공통 방법론으로 결속시킬 수 있을 것이다”

과학

과학에서는 새로운 아이디어와 발견은 선행작업에서 나온다. 따라서 가장 최신 과학적 발견과 연구에 접하는 것이 중요하다. 과학적이고 학문적인 출판물을 더욱 공개하려는 움직임 덕분에 출판된 정보 접근은 전통적인 스타일의 저널과 데이터베이스 서치를 통한 것보다 신속하고 널리 퍼진다.

무료 다운로드를 통해 동료간 검토하고, 아직 저널에 제출되었으나 아직 출판되기전의 아티클에 신속하게 접근하는 방식인 ‘preprint servers’는 1990년대초에 등장했다(그리하여 ‘preprint’로 명명). 1990년도 말에 그 다음 스텝인 프리프린트 움직임은 오픈 아카이브 이니시에이티브의 소산으로써, 독립적인 프리프린트 서버의 노력이 어떻게 학문적인 출판물의 공개를 확장될 수 있도록 협력했는지 보여줬다. 오픈 아카이브 이니시에이티브는 효율적인 콘텐츠 배포를 위한 상호운영가능한 표준을 개발하고 활성화하였다.

오픈아카이브이니시에이티브가 과학출판물에 뿌리를 두고 있지만 사회과학과 인문학을 포함한 광범위한 정보에 자유롭게 접근할 수 있는 표준을 개발하고 있다.

시기적절하고 접근가능한 과학 내용에 대한 도전을 주도한 또 다른 것으로는 the Public Library of Science이다. PLoS는 과학과 의학문헌을 공공 자원(누구나 어디서든지 무료로 이용가능한)으로 만들려는 과학자들과 물리학자들의 NGO이다. “과학적 아이디어와 방법, 결과와 결론에 즉각적이고 제한없이 접근할 수 있는 것은 과학과 의학의 진보를 가속화시킬것이다”라고 PLoS 웹사이트는 표명한다³¹.

이 목적을 위하여 PLoS는 PLoS Biology와 PLoS Medicine이라는 두개의 저널을 출판하고 온라인으로 보급했다(PLoS Medicine은 2004년 가을에 출시되었다). 저널들은 출판된 연구 보고서들의 전문과 데이터를 실고있다. PLoS는 2004년 Wired magazine의 과학분야 Wired Rave상을 수상했다. 과학출판물에 대한 무료접근의 모델을 통해 “과학 연대(cartel)을 깨뜨렸다”고 PLoS를 호평하였다³².

열대질병을 극복할 약을 발명하기위해 오픈소스접근이 요청되기도 했다. 웹사이트는 특정 질병분야의 전문성을 자원하는 생물학자와 화학자들에 의해 사용될 것이다. 과학자들은 데이터베이스 주석을 달고, 실험을 실시하고 채팅룸 등에서 결과를 공개하여 공유하는 등 협력하여 작업할

것이다. 궁극적으로 의약 개발은 경쟁 입찰을 통해 실험실에 수여될 것이지만 제품은 이를 생산할 범용 제작사들을 위해 공공 분야에서 출시되어 가장 저렴한 가격으로 광범위한 대중에게 도달할 것이다.

음악

출판에서의 오픈소스가 대체로 무료로 가능한 정보를 일컫는 반면 음악분야에서는 오픈소스의 두 개의 핵심적인 특징으로 돌아가야 한다. 무료로 가능한 정보와 협력적인 창작이다. Open Music Registry는 공개소스음악을 활성화한다. (음악은 자유롭게 공유되고 청취되고 편집되고 배포된다.) 음악을 사랑하는 이들에게 이것은 강력한 툴이다. 이 사이트가 비상업적 음악에 기여하지만 음악작품의 예술자 보상과 저작권(copyright)을 지지한다. 이 사이트에서 음악은 일렉트로닉 프론티어 화운데이션(Electronic Frontier Foundation)에 창립된 오픈 오디오 라이선스(Open Audio License)하에 라이선스된다. 공개소프트웨어원칙에 근거하여 만들어진 라이선스는 음악 저작권의 고유목적, 즉 “우리가 즐길 수 있으며 작곡가가 작곡할 수 있는 공공 공통의 생동하는 음악의 창작”을 회복시키는 것을 목표로 한다.(오픈오디오라이선스가 2001년에 등장한 오픈뮤직레지스트리를 뒷받침하는 원동력이지만 오픈오디오라이선스를 대신할 유사한 라이선스들이 있다) 오픈뮤직레지스트리는 2003년부터 2004년까지 5만6천회 이상의 무료 음악의 다운로드호출은 하루평균 120회 이상의 다운로드가 진행되고 있다.

물론 음악은 자체를 오픈소스에 차용시켜 디지털 형태(가단성있고 공유가능한 소프트웨어처럼)로 그리고 자유롭게 동료에서 동료로 파일 공유되어 테스트되고 있다. 음악이 오픈소스와 잘 맞는 곳은 창의성과 상업성의 균형을 흥미있게 테스트할 수 있을 것이다.

반면, 음악이 소스라면 음악을 전달하는 공기진동은 무엇인가? 오픈공개소스라디오스펙트럼에 대한 아이디어는 TCP/IP의 발명가이며 그룹형식의 네트워크에대한 Reed의 법칙을 만든 데이비드 리드에 의해 옹호되고 있다.(“The Architecture revolution”이라는 LEF 리포트 참조) Reed는 라디오스펙트럼의 일부를 공개하기를 원하며 개인권리를 보존하는 반면에 공공 상품창작에 비영리적으로 기여하는 창의적인 사람들의 단체(the Creative Commons organization)을 통해 라디오스펙트럼을 보호하기를 원한다.

Reed의 주장은, 스펙트럼은 희소하지않고 풍부하며, 어느 누구나 자유롭게 사용해야만 한다. 사용자들은 분리된 채널이 아닌 협력적인 네트워크로 조직화되어야 한다 등이다. 라디오 장치로 정의된 소프트웨어를 탑재함으로 네트워크 용량을 감소시키기보다는 증가시켰다. 이것은 인터넷에 마디하나를 추가함으로 그 용량을 증대시키는 것과 같다. 기술혁신은 네트워크의 가장 자리, 유저들에게서 일어난다. 좋은 아이디어를 가진 사람이라면 중앙통제 혹은 소유자의 허가 없이 방송술할 수 있다. Reed의 비전에서 오픈스펙트럼은 공공제품 서비스에 혁신을 가져왔다.

협력적 지식

오픈소스방법을 통해 지식을 추수한다는 개념은 위키페디아에서 잘 나타난다. 오픈콘텐츠백과 사전에 더불어 공개소스 협력지식의 또 다른 예로는 대학코스와 심지어는 요리책에서도 나타난다.

오픈텍스트북프로젝트(Open textbook Project)는 오픈소스방법을 사용하여 텍스트북을 발간하는 것이다. 도서는 협력하여 개발되고 미국 및 어디서든지, 이상적으로는 서비스가 충분하지 않는 시장에서도 가능하다. 협력모델은 소재가 최신화되고 지역 독자들에게 맞춰질 수 있도록 한다. 또다른 프로젝트가 위키북(Wikibooks)으로 공개된 콘텐츠 텍스트북, 매뉴얼, 기타 텍스트를 제작하고 배포하는 것으로 현재까지 80개 이상의 도서가 개발되었다.

지리적 여건이나 수입에 관계없이 지식에 접근하는 이러한 정신으로 MIT의 OpenCourseWare Initiative는 대학교재를 멀리있는 학습자들에게 웹을 통해 배포하고자 한다. 공식적으로 2003년에 착수된 이 이니시에이티브의 아이디어는 공개소스개발 모델에서 출발한 것이다. 현재 700개 코스 이상의 교재가 가능하고 MIT는 대학들의 정보공유와 교육 방식을 혁신하기위하여 2008년까지 모든 과정에 이러한 방식을 도입할 계획이다.

오픈소스가 사람을 더 현명하게 만들수 있다면 간단한 요점으로 더 나은 요리사가 될 수도 있을 것이다. 몇몇 공개소스 요리책들인 웹상에 등재해있어서 모든 종류의 요리사들이 요리법들이 사용가능하다. 컴퓨터광을 위한 식품, 국제 요리법 온라인, 카이저라우턴대학 유닉스그룹의 요리책(독일어), IT전문출판사 O'Reilly&Associate의 프로그래밍 요리책 시리즈인 커뮤니티 요리책(예,아파치 요리책) 등을 포함하여 오픈소스요리책들이 꾸준히 개발되고 있다.

요리법에서 콜레라처치를 위한 crops에 이르기까지 오픈소스는 수 많은 영역에서 그 혁신성을 빛내고 있다. 공개성 이데올로기에서 출발하여 전세계적에서 접근가능하고 지식을 개선하고 제품 개선을 위한 문제가 해결되고 있다. 인터넷이 우리들의 개인적인 삶속, 소프트웨어 영역에 관련없는 보다 많은 사람들의 모든 영역에 침투함으로 공개소스는 알려지고 많은 새 영역에서 그 잠재력을 발견하게 될 것이다.

재미요소:

공개소스는 당신의 창의력을 흘러넘치게 한다.

오픈소스는 응용과학에 비견되어진다. 과학자같은 개발자들은 흥미롭고 재미있는 새 아이디어를 탐색한다. 자신들이 직접 코드를 가지고 실험하고 조작해보기를 좋아한다., 호기심이 많고 학습욕과 다른이들과 자신이 발견한 것을 공유하는 것과 커뮤니티에 기여코자한다.

개발자들은(혹은 과학자들) 문제해결이 즐겁고 이속에서 자신감을 얻게된다. 재미있기 때문에 놀랍도록 창의적이고 헌신적으로 고도로 독창적이거나 유용한 도구를 프로그래밍하는데 많은 시간을 보낸다.

개발자들에게 더 이상 재미를 발견할 수 없다면 이러한 창의적인 의식은 프로그래밍을 멈출 것이다. 조직들은 재미요소가 공개소스개발을 위한 명백한 요소이고 프로젝트 성공을 위한 영양소라는 것을 깨달은 필요가 있다. “파티얼기”와 “재미있기”와 혼동해서는 안된다. 이것은 전문적인 열정과 즐거움이 개인적인 성과에 기여한다는 것이다. 자신의 일을 재미있게 하는 동기유발된 직원은 아닌 직원들보다 생산적인 경향을 보인다.

행동에 관한 것이지 나이가 아니다

결과를 개발하는 사람들과 명성을 개발하는 사람들이 있다. 도전적인 문제를 해결하는 것은 재미있다. 유용하거나 근사한 소프트웨어를 개발하는 것과 짧은 시간 안에 그것을 사용할 수 있는 것(힘든 생산공정이 필요없는)는 재미있다. 알려지는 것과 동료들 및 유저들에게 존경받는 것은 재미있다.

“시스템을 깨고” 자본주의적 속박으로부터 자유롭게 코드를 개발하는 것도 재미이다.(소프트웨어 혁명 참조) 공개소프트웨어를 개발하는 것은 멋진 일이다. 당신이 대기업보다 현명하다는 것과 불법적 행위를 원치않는다는 것을 증명하고 싶다면 오픈소스개발을 하라. 이것은 시대에 관한 것이 아닌 태도에 관한 것이다.

물론 재미는 주관적이어서 누군가에게 재미있는 것이 다른이에게는 지루하고 재미없는 것일 수 있다. 개발자들의 관심사가 다양한 것에 일종의 경계와 같은 것으로 커뮤니티가 있다. 그 커뮤니티의 누군가는 특별한 어떤 것 혹은 문제에서 재미를 발견하고 계속 매달릴 것이다.- 비록 엄격한 일정을 따르지 않을지라도 그러나 이것이 바로 공개소스개발의 문제가 될 수 있다(뒤로 갈수록 심해질 것이다)

재미와 게임

오픈소스개발은 개발자들이 그들의 재능을 발휘할 수 있는 기회를 제공한다. 그들은 학교나 직업과는 별도로 흥미있는 것에 관해 작업할수 있다. 그리하여 다양한 취미를 위한 공개소스애플리케이션을 볼 수 있다. 사이클리스트를 위한 길찾기(BBBike.de), 집에서 숙성시킨 맥주사이트(CyberBrau.org), e-북 매니저(eBookCollectors.sourceforge.net), 가계도(vjet.demon.co.uk/ftree)등이 그 예이다.

대규모프로젝트들은 음악 혹은 영화와같이 거의 모든이들이 관심있는 취미에 관한 것이다. Ogg vorbis 오픈소스 오디오 포맷(a la MP3)으로 된 오그멀티미디어프로젝트(Ogg

multimedia project)와 오픈소스 클라이언트 서버 비디오 스트리밍 솔루션 VideoLAN 프로젝트, 이 두개 프로젝트 모두 무료 라이선스의 스트리밍 오디오 혹은 비디오를 위한 고품질 컴프레션테크닉을 제안한다.

비디오세계에서 역시 DivXNetworks는 오픈소스 비디오 코덱 (비디오화일을 작동시키기위해서 요구되는 컴프레션/디컴프레션 소프트웨어) 작업을 하였는데, 이것은 오픈소스비디오코덱프로젝트의 고향인 Mayo프로젝트의 일부이기도 하다. 비록 이 제품은 공개소프트웨어와 같이 출시되지않았으나(이 업체는 다른 아이디어를 tap하기위해 공개소스모델을 사용했다는 논쟁을 불러일으킴) 프로젝트 Mayo, XviD를 통해 나온 다른 제품이 궁극적으로 공개소스로 출시되었다. XviD는 뛰어난 압축기능과 앞선 특징을 갖춘 고품질의 MPEG-4코덱이다. 오픈소스이기 때문에 라이선스 무료요 고객맞춤이 가능하고 데스크탑OS간에 뛰어난 호환성을 가지고 있다.

음악과 비디오는 오픈소스프로젝트만큼 재미가 가득한 것이나 확실히 최고로 재미있는 소프트웨어는 컴퓨터게임이다. 흥미롭게도 대부분의 공개소스게임은 화면디자인, 저품질 애니메이션 등 기술적인 면에서 뒤떨어진다. 그러나 최근 정교한 것들이 증가하는 변화를 보여준다.

이중 하나가 PlaneShift로 3D로 된 다중 플레이어를 위한 비주얼 환타지세계이다. PlaneShift는 미국, 유럽, 캐나다, 뉴질랜드의 20인의 핵심 개발자들과 몇 개의 부수적인 프로젝트로 된 아주 역동적인 것이다. 오픈소스란 개발자로서의 유저들이 자신들의 집 혹은 궁전처럼 요소들을 창조할 수 있으며 누구든지 그것들을 사용할 수 있고 그것 위에서 새로운 것을 만들어낼 수있다는 것을 의미한다. PlaneShift 웹사이트에서 언급하는것처럼, 오픈소스개발이기 때문에 “게임은 앞으로 계속 확장해 나갈 것이다.”

PlaneShift는 오픈소스와 성공적이고 신나는 게임들을 강력하게 신뢰한다. 이런 이유로 이 회사는 3개의 라이선스를 유지한다. 모든 소스코드에는 GPL을 사용한다. 단 게임 진행법칙은 예외이다. 게임의 비코드 자산부분, 아트워크와 텍스트에는 두 개의 자산 라이선스를 사용한다. 이 전략을 정당화하면서 PlaneShift 웹사이트는 아래와같이 주장한다

“ 우리는 너무나 많은 게임들이 실패하는 것을 보아왔으며 그 게임들에 얼마나 많은 고생들이 실려있는지 알기 때문에 참으로 그러한 것을 목격하는 것은 고통스럽다. 우리는 개별 개발자들을 모두 모아들일 방법을 찾기를 원하며 게임 플레이어들뿐아니라 동일한 소망을 가진 개발자들을 위해서도 성공적인 프로젝트를 개발하기를 원한다³⁵⁾”

PlaneShift는, 프로젝트가 여러 개로 갈라지는 것을 피하고 한 프로젝트 하에 통일된 자원을 유지하기 그리고 게임의 독특함으로 성공을 확보하기라는 두 가지 목표를 지지하기 위하여 복합적인 라이선스 계획을 가진다. PlaneShift의 라이선스는 참여하는 아티스트와 디자이너의 독특

한 창의성을 유지하는 반면 개발커뮤니티간에 창의성을 용이하게 한다.

도전해 볼만하지만 그래픽상으로 그리 정교하지않은 오픈소스게임 NetHack은 싱글 플레이어 를 위한 던전(지하토굴)탐험 게임이다. 이 게임의 저급 기술 ASCII 인터페이스에도 불구하고 로직, 특수 레벨, 캐릭터, 아이템과 컨셉면에서 특성이 풍부하다. NetHack는 1980년대에 처음으로 출시되었으며 강력하여 거의 컬트에 가까운 후속편과 연재만화의 특성을 보인다. NetHack 커뮤니티가 제공하는 다양한 플레이 시나리오와 인터페이스는 던전에 대한 재미와 탐험과 드래곤에서 영감받은 게임의 소스들이 꾸준히 등장한다.

진지한 비즈니스

새로운 기술적 기교를 상상하고 시도할 수 있는 놀이터인 게임은 재미있다. 그러나 게임은 역시 진지한 비즈니스이기도 한 것이다. 게임과玩具제작사들에게 오픈소스는 비즈니스 전략 속으로 긍정적인 결과를 드러내고 있다.

컴퓨터게임 시조 중 하나인 Doom을 고려해보자. 1993년 Doom이 출시되었을 때, Doom은 획기적인 기술과 앞선 게임플레이를 보였다. 게임의 엔진 혹은 플랫폼의 소스코드를 지켰기 때문에 이러한 독점은 게임 개발사인 id소프트웨어의 경제적 안정을 위한 선택이었다.

그러나 유사한 엔진이 등장하면서 Doom 소스코드의 가치는 하락했다. 1997년 id Software사는 GPL하에 소스코드를 공개했다. 이 회사는 새 엔진을 보유했기 때문에 그 당시 엔진과 오픈소스를 지원하는 회사로서의 지위를 포기했다. 다른 게임 엔진들, 특히 id Software사의 Quake는 이후 GPL하에 공개되었다.

Neverwinter Nights 개발사인 Bio Ware사는 다른 방식으로 공개 전략을 따랐다. 소스코드를 공개하는 대신 유저가 그들 자신만의 세계와 캐릭터를 창조할 수 있고 다른 이들과 공유할 수 있도록 툴셋트를 판매했다. 이것은 2002년 그 게임이 출시되던 그 당시 혁명적인 시도였고 이것은 오픈소스에 영감받은 것이었다.

“우리는 오픈소스운동을 면밀히 관찰했고 공개 협력의 눈에 띄는 장점을 발견했다.”고 Neverwinter Night의 제작사 Trent Oster가 회고했다. “우리가 편집 툴에 관해 이야기하기 시작했을 때 여러분만의 모듈을 개발하는 것은 너무나 많은 작업이 되지않을까 염려했다. 우리의 소망은 공개소스패션으로 게임팬들이 작업과 전문지식을 공유하는 것이었다. 게임이 선적된 후 곧 커뮤니티의 힘을 목격할 수 있었다. 이들은 숙련자들간에 작업을 배분하고 다양한 그룹의 개발노력을 지원하기 시작한 것이다. 현재까지 이 커뮤니티는 이러한 협력을 통해 정말 놀라운 일들을 해내고 있다.”

Neverwinter Night는 1백7십만명의 등록과 하루 3만이라는 유저들이 로그인하는 대대적인 성공을 이루었다. 인터넷을 통해서 2천7백개 이상의 자체 개발된 모듈들이 무료로 가능하다. 이들 개발물은 게임의 수명을 연장시키고 있으며 오늘도 잘 판매되고 있다. Neverwinter Night은 2004년 FireSquade의 PC게임 top 10에 든다.

오픈소스가 비즈니스 전략의 일부가 된 또 다른 예는 토이제작사 LEGO이다. 이 회사의 프로그래밍이 가능한 로봇장난감 MINDSTORM은 1998년에 소개되었는데 선의의 해커에 의해 역설계된다. 이 장난감의 개념도가 인터넷에 올려지고 오픈소스 개발 툴이 등장하기 시작했을 때 이 회사는 공개움직임을 포용하기로 결정했고 이 제품의 사양을 공개했고 오픈소스프로그래머를 위한 지원을 제공했다. 레고는 제품 프로그래밍을 용이하게 했고 고급 프로그래머 키트와 매뉴얼을 인터넷에서 무료로 공개했다.

그 결과는 인상적이었다. Software Development Times에서 “소란스런 커뮤니티가 Mindstorm을 통해 성장했다. C, 자바 그리고 Forth에 기반한 오픈소스개발시스템이 등장했다. 프로그래머들은 수 십개의 뉴스서버와 웹기반커뮤니티사이트를 통해 전문지식과 C++, Perl, Tcl, Visual Basic, Delphi와 Scheme으로된 샘플코드를 포함한 소스코드를 공유했다. 프로그래머들의 기업화된 그룹은 로봇을 위한 오픈소스OS를 개발하기조차했으며 개발자들이 임의적이고 복잡한 방식으로 장치를 조절할 수 있도록 허용했다.

레고의 오픈소스로의 움직임은 이 회사의 슬로건인 “여기서 놀아라”와 제대로 맞아떨어졌다. 레고사의 프로파일에 나타나듯이 이 슬로건은 “공개된 놀이에 초점을 맞추었고 한계도 걸론없이 지속할 수 있다” 이것은 오픈소스개발과 딱 들어 맞는 것이다. 아이들이 놀면서 배워가듯이 오픈소스개발자들은 재미를 통해 학습한다.

놀이는 정부와 비즈니스 세계에도 영감을 주었다. 컴퓨터게임과 군대간의 연결은 훈련과 시뮬레이션 연습을 목적으로 가까워졌다. 비즈니스에서의 시나리오 기획은 모델링과 시뮬레이션과 더불어 게임에서 나온 것이다. Doom은 보다 직감적인 환경을 제공하기위한 인터페이스로서 시스템 행정툴로 간주되기조차한다. (잘못된 과정은 제거하라!)

교훈

게임이 재미에 관한 것이고 표면상 놀이인 반면, 게임들은 이전보다 심도있어졌다. 조직들은 그들의 IT전략에 오픈소스게임에서 얻은 교훈을 적용할 수 있게 되어 효율적인 프로젝트 관리에 창의성을 통합시켰다. 조직은 기성 제품 혹은 플랫폼을 차별화할 수 있도록 확장성에 초점을 맞추기 위해 공개소스커뮤니티를 독려할 수 있다. 그리하여 제품의 수명을 연장할 수 있게 된다. 이 조직은 핵심 개발의 허브가 되고 창의적이기위해 커뮤니티가 요구하는 툴을 제공하게 된다.

급속하게 진전하는 게임산업이 환기시켜주는 것은 오픈소스개발이 게임산업과는 대조적으로 그리 빠른 진전을 보이지않는다는 것이다. 실제로 대부분의 오픈소스프로젝트들은 시간순으로 진행되지않는다. 프로젝트 매니저를 위한 교훈은 이것이다. 프로젝트 매니저의 프로젝트가 공개소프트웨어를 사용한다면 이미 있는 특성만을 사용하라는 것이다. 왜냐하면 필요한 특성이 제 때 구축되지않을 것이기 때문이다.

또한 조직들은 항상 IT 재능에 결사적으로 매달린다. 현명한 조직들은 창의적인 기술 마인드를 위해 오픈소스와 비오픈소스 양쪽 다 거절할 것이다.

노는 것이 배우는 것임을 기억하라. 놀이는 과업과 비즈니스가 어떻게 될 수 있을지, 빛나는 창의성과 동기유발에 영향을 끼친다. 공개소스가 부여하는 재미요소는 비즈니스를 위하여 인식되어져야만 하며 활용되어져야만 한다.

법과 비즈니스 이슈들

오픈소스의 법과 비즈니스 이슈들은 진지하게 취급되어야 한다. 소스코드가 공유되지않는 비공개세계에서와는 달리 위험에 처해있다. 조직들은 공개SW사용에 관해 규제될 필요가 있다. 소프트웨어에 라이선싱 규제가 부착된 것은 새로운 일이 아니지만 새로운 것이라면 오픈소스운동에 의해 소개된 광범위하게 나열되는 라이선스 의무 혹은 요구사항이다. 조직들은 공개SW 사용을 착수할 때 수많은 법적, 상업적, 정책 이슈들을 반드시 고려해야만 한다.

법적 고려사항들

Treasure Chest는 공개소프트웨어에서 비즈니스 가치를 얻을 수 있는 기회는 무한함을 보여준다. 그러나, 조직이 공개소프트웨어를 사용하기 전에 소스코드가 비록 무료일지라도 의무마저 없는 것은 아니라는 것을 반드시 이해해야 한다. 의무 불순응은 그 소프트웨어를 사용할 조직의 권리가 취소될 수 있고 불순응은 계약위반을 의미할 것이고 조직은 잠재적으로 재정적인 손실을 초래할 것이다.

공개소프트웨어 사용에 대해서 이해해야 할 두 가지 키포인트가 있다.

- 그 소프트웨어가 **공공분야에 사용된다면**(이것은 저자가 작품에 대한 카피라이트를 포함한 소유권의 모든 권리를 일반 공중에게 이양한 것), 공개소프트웨어에 대한 접근은 소유권자에 의해 결정된 사용조건에 결정된다는 것이다. 이러한 조건은 라이선스 조건이다.
- 라이선스 조건은 유사한 소프트웨어 라이선스 조건과 차이가 날 수 있고, 어떤 경우 그 공개소스라이선스 조건은 공개소프트웨어를 처음 사용하고자 한 이유를 무효화시킬 수 있다.

만일 리눅스가 가장 잘 알려진 공개소프트웨어의 예라는 전제로 출발한다면, 자유소프트웨어재단(Free Software Foundation)의 창립자 리처드 스톨만이 만든 라이선스조약을 사용하는 리눅스 토발즈가 커뮤니티를 위해 리눅스를 개발했다는 것을 이해해야 한다. 이 라이선스가 the General Public License이다.

The GPL-Free Speech, Not Free Beer

GPL은 사용 기간과 조건에 관한 것 이상이다. 또한 소프트웨어를 협력하여 개발하여 모든 이가 자유롭게 사용할 수 있도록 하는 널리 퍼진 소프트웨어 커뮤니티를 위한 비전이 그려진 정치적 선포이기도 하다.

GPL은 공개SW 사용 권리를 교환하기위한 초기 벤치마크를 설정했다.

스톨만을 자유소프트웨어를 안치시키는 4가지 타입의 자유를 언급했다.

1. 어떠한 목적이든 프로그램을 운영할 자유
2. 프로그램 자동법과 당신의 필요에 적용할 방법을 연구할 자유(소스코드 접근은 여기서는 선결조건)
3. 당신의 이웃을 도울 수 있도록 카피를 재배포할 자유
4. 전체 커뮤니티에 유익할 수 있도록 프로그램을 개선하고 개선된 것을 공중에 공개할 자유(여기서도, 소스코드 접근은 선결조건)

스톨만의 이 네 가지 자유와 자유소프트웨어의 개념에 대해 유명한 비유를 남겼다. 여기서 자유는 연설의 자유이지 맥주가 공짜가 아니다라는 것이다. 자유이념을 뜻하는 것이지 가격을 의미하지않는 것이다.

The catch

이 네 가지 자유가 GPL에 반영되어있지만 GPL에는 많은 제약 또한 있다. 특히, GPL에 있는 단서로는, 전체적으로 혹은 부분적으로 GPL하에 라이선스된 공개소프트웨어를 포함한 어떠한 새로운 작업도 반드시 GPL하에 라이선스되어야만 한다.

따라서 자신 소유의 소프트웨어를 GPL하에 라이선스된 소프트웨어와 결합시켜서 제3자에게 배포 혹은 판매 목적의 파생 제품을 개발한 소프트웨어 개발자는 GPL에 입각한 그 자신 자산 소프트웨어로 라이선스되어야 한다. 이 결과는 “바이러스적” 혹은 “감염적”이라는 본성으로 기술된다. 왜냐하면 GPL은 최초 프로그램에 자기복제적이라고 자칭하기때문이다. (GPL의 지지자들은 전혀 바이러스적이지 아니라 상호관계에 기반한다고 말한다) 따라서 그 소프트웨어개발자는 전체 소스코드 공개를 요청받을 것이다. 단지 GPL 코드의 몇 줄이 개발자의 독점 소프트웨

어에 포함될 지라도 이것은 적용된다.

이로 인하여 독점적으로 보이고 따라서 비공개된 소스코드를 개발하기 위해 R&D에 수백만달러를 들인 개발조직을 힘빠지게 하는 것이다.

예를 들어, 하나의 코드블럭속에 공개소스와 독점소스를 포함한 프로그램은 GPL조건에 따르기 위해서 독점소스를 공개하도록 요구받을 것이다.

이것이 바로 “copyleft”개념의 기본이다. 카피레프트된 라이선스는 전통적인 라이선스 개념을 뒤집는 것으로 라이선스받는 쪽에 제한보다 많은 자유를 제공하는 것이다.(카피라이트 소유권자가 독점권을 가진 경우) 따라서 카피라이트는 모든이가 프로그램의 코드 혹은 이 코드에서 파생된 프로그램을 동일한 조건으로 사용하고, 변경하고 재배포할 권리를 가지고 있음을 주장한다. (이것은 카피라이트법을 다르게 사용한 것이고 논쟁하는 변호사가 있을 수 있다. 라이선스 사용에 대해 x달러 지불을 요구하는 대신에 소유권자는 개발자의 소스코드의 카피를 제공해야 할 것이다)

이것이 이론적이거나 학문적인 것이라고 생각된다면, 다시 한번 생각해보라. Sun사가 Solaris OS의 소스코드를 오픈소스 커뮤니티에게 공개할 것을 약속한지 1주도 지나지않아서, SCO는 언론을 통해서 GPL로 솔라리스를 제공하는 Sun을 저지한다고 밝혔다. 왜냐하면 솔라리스는 다른 종류의 라이선스하에 있는 SCO의 코드를 포함하고 있기때문이다. (SCO대IBM케이스는 차후 논의된다)

만일 Sundl 솔라리스 오픈소스코드를 GPL로 공개한다면 이론적으로 볼 때 내부에서 개발된 솔라리스코드 일부 역시 공개해야하기 때문이다.

이러한 바이러스적 효과는 GPL라이센스작업으로 독점적 작업을 수행하는 것과는 별개로 GPL 라이선스에 기반하여 파생된 작업에만 적용된다. GPL라이선스 작업의 예로 리눅스를 사용하는 후자의 경우에, 리눅스를 운영하는 독점 프로그램이 리눅스를 포함하지않거나 리눅스로부터 파생되지도 않았다면 바이러스 효과는 적용되지않는다.

허용

바이러스 효과의 잠재력을 인식하고, 자유소프트웨어재단은 결과적으로 GNU Lesser General Public License(LGPL, 덜일반적인 공공라이선스)를 소개했다. LGPL은 LGPL라이선스된 소프트웨어 라이브러리와 비LGPL라이선스 소프트웨어가 덜 제한적인 방식으로 연결될 수 있도록 허용한다. LGPL은 GPL(혹은 LGPL)에 따라 독점소프트웨어를 요구하지않고 프로그래밍 라이브러리 사용을 통해 독점 소프트웨어가 GPL소프트웨어와 연결될 수 있도록 허용한다. 이러한 허

용은 다음과 같이 만들어졌다. “결국에는 사실상의 표준이 될 특정 라이브러리를 가장 폭넓게 사용가능할 수 있도록 독려되기위해서 특수한 필요가 있을 수 있다.”

아래 두개의 예시는 LGPL이 어떻게 작용하는지를 보여준다.

- LGPL 라이브러리 X는 Z이라는 BSD스타일라이센스된 소프트웨어와 링크될 수 있고 그 결과인 패키지 XZ은 Z가 LGPL하에 라이선스되어질 필요없이 하나의 전체로서 배포될 수 있다.
- GPL하에 소스코드를 가능하게 하는 OpenOffice.org는 라이브러리를 라이선스하고 LGPL 하의 OpenSource.org 소스코드의 요소기능성에 라이선스를 준다. OpenSource.org는 또한 Sun Industry Standard Source License를 상업적으로 사용할 수 있는 소프트웨어버전을 개발함으로써 이중 라이선스 모델을 지원한다.(이중 라이선스 모델은 차후 논의된다)

상기 분석은 GPL의 특별한 요구내용에 관련하여 주의를 모으려는 의도를 보인다. 자신의 코드가 자유롭게 사용되어지기를 희망하는 **소프트웨어 작가(author)들과**, 자신의 코드를 사용하는 다른 유저들이 기여하기를 희망하는 소프트웨어 작가들을 위하여, GPL은 아마도 이상적인 것이다. 만일 작가가 타인의 노동의 결실을 갈취하듯이 찾은 이 작가의 작품을 통해 어느 누구도 수익을 취해서는 안된다는 시각을 갖고 있다면 이것이 바로 이 경우에 해당한다.

그러나 자유소프트웨어 개념에 일관되게 그들의 소스코드를 공개하도록 누군가 압력을 가하고 있는가? 작자로서, 당신은 아마도 다른 유저들이 당신의 코드에 하듯이 할 다른 유저들을 원할 것이다. 그러나, 역사적으로 비공개소스코드를 배포해온 소프트웨어개발조직들을 폐쇄시킴으로써 **shutting out**, 유저커뮤니티는 긍정적인 기여를 잃어버릴 것이다-혹은 조직은 자체(non-infringing, 위법되지않고) 버전을 개발한다

덧붙이자면, 만일 전통적인 비공개소스 조직들이 자신들의 상품 속에 공개소스코드를 포함시키려했다면, 공개소스커뮤니티는 실질적인 혜택을 받았을 것이다. 왜냐하면 이 조직의 구매자들은 독점상품에서 공개소스상품으로 바꾸기만 하면 되는 쉬운 일이기 때문이다. 언젠가 그들 상품이 공개소스코드를 포함한다면 그 조직 내부의 공개소프트웨어개발자들을 위한 취업 기회는 넓어질 것이다. 사고를 위한 음식

기준**Criteria** – 과다한 라이선스

GPL은 천성적으로 제한적이며, 자유소프트웨어재단에 의해 적용되었던 단어 “자유”가 함축하는 의미로 인하여, 1990년대 후반에 공개소스이니시에이티브가 설립되었고 “공개소스”가 공식적인 명칭으로 사용되기 시작했다.

“자유소프트웨어”와 “공개소프트웨어”는 종종 번갈아가며 사용되어진다. 구분하는 것이 많은 이들에게는 아마도 학문적인 것일 수도 있지만, 자유소프트웨어재단의 지지자들과 공개소스 이니시에이티브에게 이 구분은 뜨거운 화두이다. 말하자면 순수주의자 대 실용적 관점. 이 챗터에서는 ‘공개소프트웨어’를 더 자주 사용할 것이다.

오픈소스이니시에이티브의 리더들(공개소스주창자 에릭 레이몬드를 포함)은 “OSI 인증”되기 위해 포함시키도록 요구되는 오픈소스라이센스의 표준 기준세트인 오픈소스정의(Open Source Definition)을 개발했다. 여기 기준은 아래의 내용을 포함한다.

- 자유 재배포(Free Redistribution) – 유저에게는 소프트웨어 저작권을 제3자에게 무료로 수여할 수 있는 무제한적인 권리가 있다.
- 소스코드 – 소스코드는 적절히 변경할 수 있는 형태로 반드시 가능해야만 한다.
- 파생작품 – 파생작품과 변경내용은 반드시 허용되어야하며 최초 소프트웨어의 라이선스와 동일한 조건하에 배포를 허용해야만 한다.
- 작가의 소스코드의 보존 – 라이선스가 소스코드와 함께 패치의 배포를 허용한다면 소스코드가 변경된 형태로 배포되는 것을 라이선스는 제한할 것이다. 이리하여 패치는 손쉽게 기본 소스코드와 구별될 것이다.
- 라이선스의 배포 – 라이선스 권리는 자동적으로 소프트웨어를 배포받는 모든 이들에게 전달된다.(즉, 소프트웨어는原作者의 허가없이도 제3자에게 재배포될 수 있다)
- 비차별 – 사용성은 어떠한 사람, 그룹 혹은 노력분야에도 거절되어질 수 없다.

이러한 기준은 공개소프트웨어의 폭넓은 의미를 포용할 조건들과 함께 사용될 라이선스의 과다를 초래했다. 접근 가능한 소스코드를 아직도 요구하는 반면 수 많은 라이선스는 특별히 GPL의 바이러스적 본성을 거부한다. 이런 라이선스들은 오픈소스프로젝트에 상업적 조직이 지속적으로 개입하는 것을 보장하려는 욕심에서 등장했다.

또한 이들 라이선스들은 종종 상업성 조직이 공중에게 공개될 코드에 대한 모든 권리를 소유하지 않을 것이라는 점을 깨닫는다. 예를 들어, GPL 조건하에서 출시를 허용하지 않는 라이선스 조건에 따라 기존 코드는 임베디드 소프트웨어 혹은 기타 제3자 소프트웨어를 포함한다.

더불어, 오픈소스정의의 모든 요구사항과 일치되지 않기 때문에 오픈소스를 다루지만 OSI에서 승인되지 않은 라이선스들이 많이 있다. 일례로, Sun Community Source License는 지적재산권보호와 단속이 합리적이라고 Sun사가 평가하는 국가들에 대해서만 가능하다.

다양한 오픈소스라이센스는 다른 이의 작업 혜택을 취하던지 혹은 기여를 하던지 혹은 단지 코드를 차용하든지에 상관없이 오픈소스운동에 참여코자하는 조직들에게 많은 기회를 제공한다. 오픈소스라이센스비교표가 78p~79p에 있다. 중요한 두 개의 아이템은 아래와 같다.

- BSD라이선스 - 캘리포니아 버클리대학에서 개발된 이 라이선스는 개발자들이 적절한 크레딧을 수령하는 반면 오픈소스코드의 변경이 독점, 비공개소스코드로 변환되는 것을 허용한다. 따라서 BSD는 GPL 스타일 라이선스에서 발견되는 카피레프트를 포함하지 않는다.
- Mozilla Public License - 오픈소스파일("covered"보도되는)파일로 언급)은 보도되는 파일이 변경되지 않는다면 상업적 조직 소유의 파일들과 함께 사용될 수 있다. 만일 보도되는 파일들이 변경된다면, 변경된 파일은 MPL파일로 배포되어야만 한다. MPL은 독점소프트웨어와 공개소프트웨어가 함께 작업할 수 있도록 한다.

다양한 오픈소스라이선스의 특징들과 복잡한 지식을 비교해주는 아티클은 그리 흔하지 않다. 즉, 상업적 조직은 공개소프트웨어를 변경하고 라이선스를 작가에게 돌려주고 그러나 독점 애플리케이션에서 변경된 코드를 사용하는 권리는 유지할 수 있다. 더욱이, 법의 관점에서 볼 때 어느 작품이 파생된 것인지 아니면 변경된 것인지가 항상 명확한 것은 아니다.

변경물에 라이선스하는 것에 덧붙여서, 상업적인 개입을 독려시키기위한 또 다른 접근 방식은 MySQL, Trollech AS 그리고 Sleepycat에 의해 전형화된 이중 라이선스모델이다. 이 모델에 해당하는 벤더들은 자신들의 제품에 오픈소스라이선스와 상업적 라이선스 두 라이선스 적용을 제안한다. 이것은 오픈소스프로젝트가 소프트웨어를 무료로 사용할 수 있도록 하는 것이며(따라서 광범위한 사용과 테스트를 지지), 반면 상품의 일부로 소프트웨어를 재배포하고자 하는 조직들은 상업적 라이선스를 구매하여 그렇게 진행한다. 조직은 소스코드를 공개하도록 요청받지 않으며 혹은 요청받는 경우 소스코드를 공개한다. 이중라이선스모델에 따라 한 상품은 동시에 다른 조건으로 다른 유저들에게 다른 목적을 위해 라이선스된다 (예. GPL하에 그리고 상업적 라이선스하에)

라이선스 관점에서 보면, 다양한 공개소프트웨어 라이선스의 영향을 설명하는 또 다른 방법은 페이지 80에 있는 뒤집힌 삼각형에서 잘 나타난다. GPL스타일라이선스와 독점라이선스 두 타입의 라이선스가 그들의 코드의 소유권 및 공개가능성이라는 이슈에서 삼각형의 반대쪽에 각각은 위치하고 있음에도 불구하고 양측이 각각의 조건에 얼마나 철저하게 순응하도록 요구하는지를 보여준다.

보증은 어떻게 되나?

소프트웨어 제품의 유저들은 라이선스권자로부터 여러가지를(비록 한정되지만) 보증받기를 기대하기 때문에 라이선스권자들은 유저들에게 일정 상황에 의지하는 형식으로 편안함을 준다.

그러나, 한 개인이 공개소프트웨어를 사용하는 것에 대한 통제를 최소화시켜서 보장측면이결여

된 라이선스 조건이라는 것이 이상한 것은 아니다. 아래서 상세하게 설명된다.

- 소프트웨어는 “현재형”을 기본으로 제공된다.
- 사용과 수행에 연관된 모든 보증은 포기된다.
- 라이선스권자에 의한 어떠한 채무로 제외된다(혹은 채무는 심각하게 제한된다)
- 구매자를 대상으로 제3자가 일으킨 지적재산 위반을 요구함에 대해 라이선스권자는 면책되지않는다.

주요 오픈소스라이선스

78P

	GNU General Public License(GPL) Version2	GNU Lesser General Public License(LGPL)V.2.1	BSD License July 1999 version
라이선스사용	라이선스는 어떠한 소프트웨어에도 사용될 수 있으나, 변경될 수 없다	라이선스는 어떠한 소프트웨어에도 사용될 수 있으나, 우선적으로 코드 라이브러리를 통해 사용되도록 설계된다. 라이선스는 변경될 수 없다	어떠한 소프트웨어에도 사용가능하다
비용	라이선스 받은이는 구매자에게 카피를 제공하는 물리적인 행위에 대해 비용을 물리며, 추가적으로 제안되는 보장보호에대한 비용도 물게된다.	라이선스받은이는 구매자에게 카피를 제공하는 물리적인 행위에 대해 비용을 물리며, 추가적으로 제안되는 보장보호에대한 비용도 물게된다.	언급없음
재배포	재배포(어떠한 변경도 포함하여)는 반드시 GPL 조건위에서 되어야 한다.	재배포(어떠한 변경도 포함하여)는 반드시 GPL 혹은 LGPL 조건에서 되어야 한다.	BSD 라이선스가 제공하는 조건에 따른 재배포는 가능하다

소스코드사용 가능성	어떠한 재배포라도 반드시 소스코드가 병행되어야하거나, 라이선스 허가받은자는 최소 3년간 소스코드를 요청이 있을 때 반드시 제공해야한다	어떠한 재배포라도 반드시 소스코드가 병행되어야하거나, 소스코드는 대상코드와같이 동일한 장소에서 접근이 가능해야 한다.	소스코드 사용하기 위해 필수요건이 없다
다른소프트웨어에 대한 애플리케이션	전체 혹은 부분적으로 포함하는 혹은 파생된 작업이든지, 라이선스된 소프트웨어는 반드시 GPL하에 배포되어야만 한다 그러나 프로그램에서 파생되지않은 독립적이고 분리된 작업은 그렇게 라이선스될 필요가 없다. 동일한 배포매체 위에 각각의 소프트웨어를 단순히 번들링한 것은 GPL하에서는 분리된 소프트웨어	편집되거나 링크되어 ‘라이브러리’를 사용하는 프로그램’으로서 라이브러리의 일부에서 파생되지않은 프로그램은 이 라이선스에 포함되지않는다. 그러나 편집되거나 링크된 결과로서 그 라이브러리를 포함하는 소프트웨어는 구매자가 변경하여 사용하는 것을 허용하고 이러한 목적을위해 역설계할 수 있도록하는 라이선스 조건 하에 배포될 수 있다.	언급없음
보장 및 보상	보장없고 법에 의해 허용된 범위까지 보상이 완전히 배제됨. 라이선스허가받은이는 작성한다면 그 자신의 구매자에게 보장을 제안할 수 있다.	무보장 및 법이 허용하는 범위까지 보상 배제된다. 라이선스 허가받은자는 자신의 구매자에게 보장을 제안할 수 있다.	무보장 및 보상 배제
관할법	없음	없음	없음
종료	라이선스는 자동적으로 라이선스받은자가 그 조건을 위반할때 종료됨	라이선스 허가받은 이가 조건 위반치 자동적으로 종료된다	종료 조항언급없음

	Mozilla General Public License(MPL) Version 1.1	IBM Public License Version 1.0	Apache License Version 2.0
라이선스사용	라이선스명을 개명된다면 라이선스가 변경되어 다른 소프트웨어에 적용될 수있다. 그렇지않을 경우 새 라이선스의 조건이 다르다는 것을 명시해야한다.	라이선스는 오직 라이선스된 IBM소프트웨어와 기여물에만 가능하다	어떠한 소프트웨어에도 사용가능하다
비용	라이선스허가받은 자는 추가되는 보장, 지원, 보상 혹은 제안되는 책임의무에대해 비용을 지불할 것이다.	조항 없음	언급없음
재배포	변경될 경우는 반드시 MPL 하에서 라이선스 받아야한다. 소스코드로 공개되지않은 비공개코드(최초 라이선스된 소프트웨어와 이의 변형)는 MPL과 편집된 라이선스가 가능하다면 다른 라이선스 조건으로 배포될 수 있다.		BSD 라이선스가 제공하는 조건에 따른 재배포는 가능하다

소스코드사용 가능성	모든 변형은 소스코드가 같은 매체에서이든 혹은 따로 분리되어 사용가능 해야만 한다. 후자의 경 우 변형은 최소 12개월 동안 가능하거나 그 다 음 버전이 나온 후 6개 월동안 가능해야만 한다.		소스코드 사용하기 위해 필수요건이 없 다
다른소프트웨 어에 대한 애플 리케이션	라이선스 허가받은 이는 다른 프로그램과 그 소프트웨어를 조합할 수 있다. 이 경우 다른 프로 그램은 MPL의 영향을 받지않는다. 그러나 라 이센스는 다뤄지는 모든 코드에 대해 여전히 MPL를 따라야 한다.(최 초 라이선스된 소프트웨 어와 그 변형들)	편집되거나 링크되어 ‘라이브 러리를 사용하는 프로그램’으 로서 라이브러리의 일부에서 파생되지않은 프로그램은 이 라이선스에 포함되지않는다. 그러나 편집되거나 링크된 결 과로서 그 라이브러리를 포함 하는 소프트웨어는 구매자가 변경하여 사용하는 것을 허용 하고 이러한 목적을위해 역설 계할 수 있도록하는 라이선스 조건 하에 배포될 수 있다.	언급없음
보장 및 보상	무보장. 보상배제, 단 제한을 가하는 법을 무 시하여 초래된 죽음 내 지는 개인적인 상해에 대한 보상은 가능하다.	무보장 및 법이 허용하는 범위 까지 보상 배제된다. 라이센 스 허가받은자는 자신의 구매 자에게 보장을 제안할 수 있다.	무보장 및 보상 배 제
관할법	캘리포니아	없음	없음

종료	라이선스가 위반되고 라이선서가 이 위반을 인식하고 30일이내에 처리하는데 실패한다면 라이선스는 자동적으로 종료된다(부차적인 라이선스가 수여된 경우는 종료되지않는다) 라이선시가 기여자에 대해 특허위반클레임을 한 경우 종료된다.	라이선스 허가받은 이가 조건 위반치 자동적으로 종료된다	종료 조항언급없음
----	---	--------------------------------	-----------

소프트웨어 라이선스 오버뷰

여기 소프트웨어 라이선스 오버뷰에서, 삼각형의 세 꼭지들은 주요 라이선스 타입인, GPL, 상업적 라이선스, 공공영역의 라이선스를 대표한다. 꼭지부분에서 이동하면서 변형된 라이선스들을 보게 된다. GPL과 상업적 라이선스는 대부분의 라이선스 의무를 수반하나 소스코드 소유권과 사용가능성에 관해서는 독점적 척도의 가장 반대쪽 끝에 있다. 공공영역 소프트웨어는 라이선스가 없다.

유저커뮤니티의 크기는 변호사들이 “타이틀의 연쇄”로 언급하는 것이 무엇인지 규명하기 불가능하기 때문에 이러한 상황이 등장한다(투명한 역사는 소프트웨어 제품의 소유권을 결정한다.) 유저들에게 제품에대한 라이선스 권리를 가진 licensor로부터의 보증과 제3자로부터 유저에 대한 클레임으로부터 유저를 보호할 보상도 없이, 그 소프트웨어 사용에 대한 소송(잘못된 소송)으로 말미암아 유저가 치뤄야 할 손해를 상환할 법적 장치도 없다.

덧붙여서, 유저는 제3자가 지적재산권의 침해를 청구하는 것에 대해서 스스로 자신을 보호해야 한다.

마지막으로 유저는 그 소프트웨어사용이 유저의 필요나 기대에 미치지 못했을때를 위한 대안이 없다.

제3자의 청구에 대한 아무런 보증과 보장 부재는 보다 전통적인 소프트웨어라이선스모델과 의미있게 다른 점을 부각시킨다. 이것은 애매한 공개소프트웨어와 보다 중심에 있는 공개소프트

웨어 양쪽에 똑같이 적용된다.(예를 들어, 79P의 테이블에 있는 IBM Public License Version 1.0을참조)

다양한 공개소프트웨어 라이선스에는 표현된 의무와 책임의 상당부분이 배제됨은 라이선스 상의 동의부분이 강제적이 아님을 의미하는 것에 주목해야 한다.

예를 들어, 오픈소스라이선스는 계약이고 강제적이기를 ‘고려하도록’하는 요청이라는 것이다. 그러나, 무엇인가를 수행하기로 계약을 하지만 계약내용 수행에 실패 시에 대한 그렇게 하는데 대한 보장이 배제된다는 것은 사실상 아무것도 약속하지않는 것이다(따라서 실제적인 고려됨이 없다) 만일 이것이 정확히 바라본 것이라면 계약은 전혀 효과를 내지않을 것이고 licensor는 배제조항의 혜택을 받지못할 것이다.

GPL스타일 라이선스에는 원칙적으로 적용되는 또다른 이슈는, 배포자가 GPL 조건을 알고 있는지 혹은 작은 규모의 오픈소스코드가 상품에 임베디드되든지 상관없이 GPL소스코드를 담은 코드를 배포하는 자에게 자동적으로 적용되는 라이선스 조건을 주장하기 때문이다. 이 컨셉은 다양한 국가에서의 기본적인 계약법 원칙과 상충되고 있다.

SCO 케이스

SCO대IBM케이스는 반드시 고려해야 할 케이스는 아니다. 왜냐하면 이 케이스는 공개소프트웨어를 개발하기위해 코드를 사용하는 것과 관련되기때문이다. (이 경우, SCO는 SCO의 Unix 코드의 일부가 IBM에의해 카피되었으며 리눅스에 포함되었다고 주장한다) 이 케이스는 SCO의 지적재산권이 IBM에의해 침해 혹은 IBM이 기밀정보 훼손에 관한 계약위반을 케이스이다.

IBM은 아이로니컬하게 SCO가 GPL를 위반했으며 IBM 특허의 일정부분을 위반했다고 맞대응하였다. 이결과는 GPL의 조건이 미 카피라이트 법에 맞지않기 때문에 미법아래에서 GPL은 위헌이라고 SCO는 추가적인 맞소송을 요청하였다.

SCO는 엔드유저인 Daimler-Chrysler와 AutoZone에 대한 법적 행위를 시작하면서 분담금을 올렸는데 SCO의 주장에 의하면 이들은 SCO가 IBM을 대상으로 한 클레임의 내용인 Unix 코드를 이들이 사용하고 있다는 것이다.

SCO 케이스는 법정에서 합리적이고 강제적인 일련의 용어와 조건이 결정되듯이 GPL의 타당을 입증해야한다.

그러나 이 케이스는 저작권의 위반이든 특허권의 위반이든지간에 이것에 기반한 제3자의 위반이 가질 수 있는 리스크를 클레임한 것으로 주목된다. 공개소프트웨어라이선스는 특히 GPL이

지적재산권 위반의 책임을 부인하는 것 같아서, 개발자는 기존 독점코드를 복제하고(그리하여 카피라이트를 위반하는) 공개소스솔루션의 기초로 사용하는 것은 전적으로 개연성이 있다. 이것은 결국이 코드는 시장전체에 배포되게 된다.

사실, 미국 기업인 오픈소스위기관리회사 (open source risk management LLC)는 이와 유사한 상황들에 대한 보험제안을 위해 설립되었다. SCO경우와 유사한 소송들은 리눅스 밴더와 유저들을 대상으로 진행될 것이다.

법정에서의 GPL

최근 GPL에 대한 첫 법정 판결은 아마도 뮌헨지방법원에서 2004년 4월에 Sitecom Germany GmbH사에 대해 진행된 것이다. Sitecom사는 GPL라이센스되었으며 netfilter/iptables프로젝트에서 개발된 소프트웨어 기반의 무선엑세스라우터 제품을 제공하고 있었다.

법원의 명령에 따르면, Sitecom사는 netfilter/iptables 소프트웨어에 적용된 GPL에 나타나는 의무사항을 충실히 진행하지않았다는 것이다. Sitecom사는 유저 커뮤니티에게 소스코드를 공개하지않았으며 GPL조건 하에 유저들에게 그 제품이 제공하지않았기 때문이다.

법원의 명령은 Sitecom사가 GPL에서 명기한 의무사항을 따르지않는다면 그 제품을 배포할 수 없도록하는 것이다. 이 글을 쓰는 시점에 sitecom사의 대응이 어떤 것인지는 명확하지않다.

상업적 고려사항들

공개소프트웨어를 포함하는 솔루션을 사용하는 것이 안전하지않다는 것을 SCO케이스가 의미하는 것인가? 절대 그렇지않다. SCO케이스가 말해주는 것은 소스코드에대한 카피라이트를 가진 자를 주변에는 종종 복잡한 일들이 발생한다는 것이다.

공개소프트웨어를 위한 보장 계획

SCO의 실제 클레임과 리눅스 라이선스에 대한 두려움에 기술하면서, HP사와 같은 디스트리뷰터들은 그들의 리눅스를 구매하는 고객들을 보장할 것이라고 공포하였다. 단 구매자들이 리눅스의 소스코드를 허가없이 변경하지않는 경우에 한해서 이다. 더불어,레드햇, 노벨/수세 등 리눅스 판매사들에 의해 OS를 업데이트한다는 것에 동의해야만 한다. HP는 그들의 구매자들이 리눅스 개발이 진전되는 것에 안도할 수 있도록 보장하는 것이 필요함을 발견한 것이다.

유사한 예로는, Sun은 리눅스기반 자바 데스크탑 시스템과 솔라리스 OS를 사용하는 구매자들을 보장하는 데에 동의했다.

SCO소송이 성공하지않을 것이라고 생각하는 것 같다. 이것이 적합한 예가 아니라면, HP와 Sun

에 미친 재정적인 충격은 고려할 만한 것임에 분명하다.

보장을 제한하는 독점 라이선스

독점밴더들의 최근 경향은 그들의 표준 라이선스 조건을 다양하게 하여 그들 제품에 포함되게 될 어떠한 임베디드 소프트웨어에 대한 보장도 제한하는 것이다. 이것은 그 임베디드 소프트웨어가 근본적으로 공개소프트웨어에서 출발했든지 아니든지에 상관없이다.

예를 들어, OEM 선두주자는 소프트웨어 라이선스 약관에 이렇게 명시했다. 제3자가 지적재산권을 위반했다고 주장하는 클레임에 대해서 라이선스를 보장할 의무가 없는데 여기서의 클레임은 “오픈소스에 한정하지않는 제3자의 코드”를 말한다.

라이선스 허가받은자가 공개소프트웨어 라이선스에 대해서 안다면 이러한 예외란 가능할 것이다. 그러나 라이선스 소유자는 라이선스 허가받은 자가 라이선스 소유자의 제품에 임베디드 소프트웨어가 있거나 혹은 없거나에 경우의 모든 위험을 가정할 수 있도록 하는 것이 상식적으로 보이지않고 이것은 반드시 라이선스 허가받은 자에의해 거절되어야한다. 그렇지않을 경우 그 소프트웨어의 콘텐츠를 보다 잘 아는 자에게서 라이선스 허가받은 자에게 제공되어야만 하는 기본적인 보호 내용 중 하나를 상실할 것이다.

책임

Treasure Chest에 명기되었듯이, 구매자는 풍부한 기능성, 저렴한 가격 그리고 곧바로 사용할 수 있는 혁신적인 솔루션을 원한다. 이것은 소프트웨어개발사 혹은 서비스제공사에게 상당한 압력으로 작용한다.

구매자들 중에는 개발자 메일링리스트, 뉴스그룹 혹은 동료 유저그룹에 의지하는 지원은 수용하지않으려는 경향을 보인다. 이러한 구매자는 공급자가 믿을만하고 배송준비가 되어있기를 기대한다. 그리하여 비록 공개소프트웨어가 주로 ‘as is’ 기본으로 제공되지만, 공개소프트웨어가 포함된 솔루션을 판매하는 소프트웨어회사나 서비스제공사는 구매자가 판매되는 솔루션의 성능을 커버하는 품질보증뿐만아니라 지속적인 기술지원보장과 제3자에대한 보상까지 기대한다.

공개소프트웨어를 개발자를 보호하기위해 오픈소스커뮤니티로부터 “back-to-back” 품질보장 혹은 보상없이 소프트웨어회사 혹은 서비스제공사는 적절하게 교정할 수 없는 것에 대해 잠재적인 노출을 추정하고 수용할 필요가 있다.

정부관점

정부차원에서 공개표준을 채택하고 특히 공개솔루션을 사용하는 하도록 권고하는 것은 전 세계적인 관심사이다. 이러한 관심은 정부가 밴더가 독립적이기를 희망하는 기초에서 정당화된다.

최근 예로는 윈헨시에서 리눅스와 기타 공개소스데스크탑 애플리케이션을 공공행정 컴퓨터 1400대에 설치한 것이다.

미국방성은 DoD의 프로젝트를 위해 공개소프트웨어솔루션을 선택했다. 그러나 2003년 5월 28일자 메모에 따르면 국방 Assistant Secretary이자 DoD의 Chief Information Officer인 John Stenbit는 언급하기를 GPL라이센스의 공개소프트웨어는 여전히 정부 정책에 맞출 필요가 있는 제한들을 포함하고 있다는 것이다. 여기서 암시하는 것은 GPL코드는 DoD 요구사항과 상충될 가능성이 존재한다는 것이다. 따라서, Stenbit의 메모에서 명확하게 들어난 것은 밴더는 어떠한 공개소프트웨어솔루션이든지 합법적인 라이선스 요구내용을 따르는지를 확신해야만 한다는 것이다.

라이선스 조항이 복잡할 수 있기 때문에 특정 라이선스의 법적 의미를 충분히 이해되었는지 그들의 법률 자문들이 DoD Components를 검토하도록 강력하게 요구되고 있다.

정책적 고려사항들

공개소스가 사용된다면 소프트웨어개발조직은 확실하게 문서화하고 소스에대한 지식을 보유하고 있어야하는 것이 명확하다.

왜 이러한 것들이 필요한가? 왜냐하면 소프트웨어개발조직의 모든 고용인들이,배포되어 사용되는 코드의 근원을 정밀하게 추적할 수 있다고 단순히 믿기때문이다. 기존 코드가 재사용될 수 있을때 왜 코드를 개발하는가? 많은 공개소스에 이리도 쉽게 접근이 가능하기때문에 문자적으로 키보드의 한 키를 터치하듯이, 공개소스코드가 독점코드가 될 가능성은 아주 힘들다

비즈니스와 법적 대표의 서명을 따라 기술적 품질적 보장을 포함한 내부 승인과정이 필요하다. 이러한 승인 프로세스는 다음과 같은 사항이 필요하다.

- 사용을 위해 제안된 공개소프트웨어를 포착하고 어떤 것인지를 밝히는 능력을 가져야 한다
- 왜 특정 공개소프트웨어가 사용되어야만 하는지를 고려해야만 한다
- 그렇게 사용하는 것이 조직의 비즈니스 전략과 일치하는지를 결정해야만 한다.
- 그 조직을 위해 코드를 쓰는 직원 혹은 계약직원의 노력의 제품이 그 조직에 소유권이 있음에 동의하도록 해야만 한다.
- 기술적 사양과 라이선스 조건 측면에서 그 코드를 내부에서 사용하는 것에 대해 승인을 확정해야만 한다
- 적용가능한 라이선스 조건과 지원과 품질관련 이슈들을 지속적으로 모니터 할 사람을

확정해야만 한다.

더불어, 만일 그 코드가 아직 승인되지않았다면 다음의 사항에 대해 신중해야 한다.

- 법률 자문가가 라이선스 조건을 검토하기
- 입찰 혹은 배포관리자가 솔루션의 일부로서 제안된 코드의 사용과 관련된 비즈니스적 이슈들을 검토하기
- 품질 보증, 배포 보증 혹은 대응그룹은 적절하기만 하다면 지원 비용을 결정하게 되고 기술적 이슈에 관해 서명하기

아마도 역사적으로 지적재산권 분야 내 소프트웨어솔루션에 사용된 코드에 이렇게도 많은 관심이 쏟아진 경우가 없었을 것이다. 그러나 공개소스에 대한 접근 용이함과 최근 SCO소고송으로 조직은 소프트웨어 솔루션 내의 코드에 적용되어질 수 있는 코드의 근원과 사용조건에 점점 더 주의를 기울이고 있다.

독점 소스와 오픈소스를 추적하고 관리하기위한 잠재력있는 감수 및 관리 툴이 만들어져서 조직 내 공개소스 사용의 유지관리 및 품질통제를 지원하는 것은 그리 놀라운 일이 아니다.

블랙덕소프트웨어사는 “제품이 아직도 개발중임에도 불구하고 개발프로젝트 내의 공개소스코드를 확인할 수 있고 잠재적인 라이선스 문제를 인식할 수 있는... (그리하여) 당신은 관리상의 도전과 IP(지적재산)위험을 경감시키는 반면 공개소프트웨어의 혜택을 즐길 수 있게된다” 툴을 구비할 것을 요청한다.³⁹

전체 소스트리를 해독하고 특정 가지는, 말하자면 Common Public License계열의 에클립스 프로젝트 출신과 다른 가지는 GPL 계열이라고 결정지을 수 있을까? 시간이 말해줄 것이지만, 오픈소스커뮤니티 자체가 그 해결책을 발견할 것이라고 혹자는 기대할 것이다.

공개소프트웨어의 존재는 IT산업에 익숙하지만, 공개소프트웨어사용과 병행하는 의무내용은 그렇지 않다. 특히, GPL은(비록 많은 소프트웨어혁신의 근원이지만) 독점소프트웨어 라이선싱 비즈니스의 상업 조직에 그리 유해한 영향을 끼칠 것이다. 이런 경우에 해당하는 기업은 공개소프트웨어를 어떠한 의도로 사용하던지 정밀하게 조사하는 적절한 프로세스를 이해해야만 한다.

첨부하여, 소프트웨어기업과 서비스 제공사들이 반드시 깨달아야만 하는 것은 궁극적으로 공개소프트웨어가 포함된 소프트웨어 솔루션을 제공하는 글들은 어떠한 구매자에게든지 보장을 제공해야 하는 것과 관련된 책임을 기업이 떠안아야한다는 것이다. 구매자들은 구매자들은 자유 내지는 공개소프트의 이타주의적 관념에는 관심이 없고 단지 그 소프트웨어가 작동하는지만을

생각하기 때문이다.

SCO소송에도 불구하고 오픈소스솔루션 사용은 안전하다. 산업계나 정부 조직은 IT비용을 지속적으로 감축하고 보다 풍부한 기능을 구비한 새로운 애플리케이션을 찾기 때문에, 공개소스가 가진 기회는 실제적으로 무한하다고 볼 수 있다. 모든 잠재적인 공개소스구매자들은 공개소프트웨어와 자신들이 사용할 라이선스를 반드시 숙지해야만 한다. 특히, 사내에서 개발을 진행할 경우는 더욱 그러하다. 이들은 신뢰할 수 있는 서비스 제공사들에게 의존할 수 있다. 즉, 전문가에게 맡기는 것이다.

시작하기

오픈소스는 여기에 존재할 것이다. 전 세계 조직들은 공개소프트웨어를 사용하고 있다. 비즈니스, 정부, 소프트웨어 밴더들조차 이 움직임에 가세하고 있다.

소프트웨어 인프라스트럭처로 시작할 때 공개소프트웨어가 주도하는 곳이라면 그리고 그 stack을 쌓아갈 때, 많은 수의 공개소프트웨어 제품은 OS, 데이터관리시스템, consumer software로 점점 많이 수용되고 있다. 상업적인 리눅스 배포판은 컨슈머용 out-of-box 데스크탑 소프트웨어를 판매하고, 오피스용 소프트웨어와 MS에서만 가능했었던 멀티미디어 소프트웨어를 번들로 제공한다.

오픈소스는 조직들이 그들의 IT전략으로 고려할 필요가 있는 대체품을 제안한다. 오픈소스는 모든 상황을 위한 것이라기 보다는 조직이 이유를 이해해야만하는 경우를 위한 것이다. 조직은 오픈소스의 비즈니스 밸류를 검증해야만 하고(아래에 있는 오픈소스의 비즈니스 밸류 참조) 그들의 IT 인프라스트럭처와 개발 프로세스를 자세히 들여다 보아야 한다.

(표)

조직과 상황에 따라 오픈소스 가치제안은 다양해진다. 오픈소스의 비즈니스 가치를 진단하는 한 가지 방법은 5개의 핵심 비즈니스 추동인자에 대한 보고 동향을 검증하는 것이다. 정확히 과학적이지는 않지만 이 표는 어떠한 동향이 어떤 분야에서 가장 강력하게 작용하는 지를 그림으로써 보여주어서 조직이 최고의 기회 영역에 주력하게 한다.

이어지는 내용은 조직들이 오픈소스 여행을 시작하기 위한 단계들이다.

비즈니스 전략 관점

- 공개소프트웨어를 사용하거나 공개소프트웨어로 전환하는 이유를 명확히 이해하라

- 독점소프트웨어에서 공개소프트웨어로의 전환비용을 이해하라
- 공개소프트웨어사용으로 얻게되는 잠재적으로 높은 이윤이 있는 비즈니스에서의 'sweet spot'을 찾으라
- TCO를 계산할 때 당신의 특수한 상황에 대한 중요한 TCO 요소를 밝히라. 독점소프트웨어와 공개소프트웨어간의 차이가 금전적인 가치가 있는가? 독점밴더로부터 당신은 얼마만큼 평화의 정신을 구매하는가? 등을 물어보라
- 독점밴더와의 협상도구로 공개소프트웨어를 사용하라
- 만일 전략적 차별인자가 아니라면 오픈소스처럼 내부소프트웨어를 공개하는 것을 고려하라(은행이 EAI를 위한 기술적 프레임워크를 개발한다면 그 은행은 그 프레임워크를 오픈소스커뮤니티에 공개할 것이다. 그러나 은행이 새로운 은행이체시스템을 개발한다면 아마도 그것을 공유하려하지않을 것이다) 이렇게 공개할 때 개발비용을 분산시키고, 품질을 개선하고, 사용율을 높이고, 잠재적으로 산업 표준을 제정하고 결국에는 그 최초 조직은 선두주자이고 혁신자가 된다.
- 조직이 공개운동에서 어떠한 역할을 해야할지를 논의하라

행정적 관점에서

- 오픈소스라이센스에 관한 내부 법률 전문가를 키우라
- 개발자들이 사용하는 오픈소스라이센스에 책임지고 순응하는 것과 더불어 배상과 책임에대해서 확실히 이해시키라
- 오픈소스를 반영하기위해 지적재산에관한 인적자원정책을 업데이트하라
- IT스텝과 유저들이 오픈소스를 능동적으로 지원하도록 하라
- 오픈소스로 전환을 위한 변화관리계획을 수립하라
- 변화에 있어서 챔피언을 찾아라
- 오픈소스로 전화하는 모든 단계가 관리가능하도록 하라
- 적절한 기술과 관리지원이 되는 팀을 결성하라

기술적 관점에서

- 전문기술을 구축하고 오픈소스밴더와 오픈소스커뮤니티와 관계를 형성하라
- 개발자와 시스템관리자가 그들의 작업과 관련된 오픈소스프로젝트에 참여하도록 독려하라
- 사용하거나 사용을 고려할 공개소프트웨어의 '안전리스트'를 만들고 그 리스트를 최신화하라
- 가장 유망한 기회의 시연을 착수하라. 시작하되 중요하지않은 시스템으로 하라
- 상호호환성에 관해 공개소프트웨어를 진단하라
- 플랫폼 독립성에 관해 공개소프트웨어를 진단하라
- 착수시점부터 보안을 설계하라

- 공개소프트웨어 활성화와 사용에관해 정책을 개발하라(예. 인프라스트럭처, 소프트웨어개발, 미션크리티칼한 애플리케이션, 보안애플리케이션에 관한 정책)
- 커뮤니티에서 신뢰를 쌓고 신뢰를 개선할 수있도록 공개소스프로젝트에 기여할 소프트웨어를 고려하라. 이를 통해 당신과 당신의 능력에 커뮤니티의 자원을 제공하려는 커뮤니티의 의지를 대폭적으로 개선할 것이다.

오픈소스는 시장에서 지속적으로 힘을 모아갈 것이다. 오픈소스는 전세계적인 운동이지만오픈소스의 채용은 (그리고 일반적으로 공개소스에 대한 태도)는 국가마다 상당히 다르다.

오픈소스는 신뢰할 수 있는 기술제공자이자 책임간격을 매꾸는 역할을 잘 수행하는 서비스 제공사에게는 좋은 기회이다. CSC는 오픈소스운동에 깊이 개입되어있다. 이 회사는 많은 제품, 서비스 인물 및 공개소스커뮤니티에 대한 지원을 포함하여 공개소프트웨어를 확장적으로 사용할 수 있도록 약정을 가지고 있다.

CSC는 오픈소스를 독점소프트웨어배너와의 전투로 바라보기보다는 고도의 개방성과 협력이 가져오는 가치를 이해하고 또한 이것은 비즈니스에 적용하는 것이라고 본다.

구매자, 생산자와 서비스제공사에게 있어서, 오픈소스로 착수되는 시간이 무르익어가고 있다. 이것은 IT 전략과 소프트웨어를 믹스하는 비즈니스적 결정 요소이다. 현명한 기업은 공개소스를 관리하고 활용하지 무시하지않을 것이다. 오픈소스는 여기 있다. 오픈소스는 비즈니스를 위해 공개되어있다.