[Field Research]

A Study on the necessity of Open Source Software Intermediaries in the Software Distribution Channel
소프트웨어 유통에 있어 공개소프트웨어 중개자의필요성에 대한 연구

Seung-Chang Lee(이승창)*, Eung-Kyo Suh(서응교)**, Sung-Hyuck Ahn(안성혁)***, Hoon-Sung Park(박훈성)****

Abstract

Purpose − The development and implementation of OSS (Open Source Software) led to a dramatic change in corporate IT infrastructure, from system server to smart phone, because the performance, reliability, and security functions of OSS are comparable to those of commercial software. Today, OSS has become an indispensable tool to cope with the competitive business environment and the constantly-evolving IT environment. However, the use of OSS is insufficient in small and medium-sized companies and software houses.

This study examines the need for OSS Intermediaries in the Software Distribution Channel. It is expected that the role of the OSS Intermediary will be reduced with the improvement of the distribution process.

The purpose of this research is to prove that OSS Intermediaries increase the efficiency of the software distribution market.

Research design, Data, and Methodology − This study presents the analysis of data gathered online to determine the extent of the impact of the intermediaries on the OSS market. Data was collected using an online survey, conducted by building a personal search robot (web crawler). The survey period lasted 9 days during which a total of 233,021 data points were gathered from sourceforge.net and Apple's App store, the two most popular software intermediaries in the world. The data collected was analyzed using Google's Motion Chart.

Results − The study found that, beginning 2006, the production of OSS in the Sourceforge.net increased rapidly across the board, but in the second half of 2009, it dropped sharply. There are many events that can explain this causality; however, we found an appropriate event to explain the effect. It was seen that during the same period of time, the monthly production of OSS in the App store was increasing quickly. The App store showed a contrasting trend to software production. Our follow-up analysis suggests that appropriate intermediaries like App store can enlarge the OSS market. The increase was caused by the appearance of B2C software intermediaries like App store. The results imply that OSS intermediaries can accelerate OSS software distribution, while development of a better online market is critical for corporate users.

Conclusion − In this study, we analyzed 233,021 data points on the online software marketplace at Sourceforge.net. It indicates that OSS Intermediaries are needed in the software distribution market for its vitality. It is also critical that OSS intermediaries should satisfy certain qualifications to play a key role as market makers. This study has several interesting implications. One implication of this research is that the OSS intermediary should make an effort to create a complementary relationship between OSS and Proprietary Software. The second implication is that the OSS intermediary must possess a business model that shares the benefits with all the participants (developer, intermediary, and users).The third implication is that the intermediary provides an OSS of high quality like proprietary software with a high level of complexity. Thus, it is worthwhile to examine this study, which proves that the open source software intermediaries are essential in the software distribution channel.

Keywords : Software distribution channel, Open source software, Open source software intermediary, Proprietary software.

JEL Classifications : D83, L86, M15.

1. 서론

공개소프트웨어와 ICT(Information & Communication Technology)는 소프트웨어 시장에 많은 변화를 가져왔다. 기업은 업무를 효율적으로 처리하고, 정확하고 신속하게 의사결정을 하기 위해 소프트웨어를 사용한다. 공개소프트웨어는 시스템 서버에서 핸드폰에 이르기까지 사용범위가 넓어지면서 기업들의 정보기술 기반구조에도 큰 변화를 초래하고 있다. 이제 모든 기업들은 소프트웨어를 사용하지 않고 업무를 처리할 수 없는 상황에 직면했고, 이러한 가운데 기업들은 소프트웨어 투자비용이 지속적으로 증가하는 부담을 가질 수밖에 없게 되었다.특히 급변하는 비즈니스 환경과 정보기술 환경에서 대응하기 위해 자사에 적합한 소프트웨어를 신속하게 개발하는 것은 기업생존 문제가 되었다. 소프트웨어 개발업체와 일반 기업들은 값비싼 소프트웨어의 대안으로 공개소프트웨

* Professor of Small & Medium Business Corporation, 24 Gukjaegeumyung-ro, Yeongdeungpo-gu, Seoul, South Korea, E-mail: lee_seungchang@yonsei.ac.kr

** Corresponding author, Assistant Professor, School of Business, Yonsei University, 50 Yonsei-no, Seodaemun-gu, Seoul, South Korea. Tel: +82-2-2123-6565, E-mail: eksuh@yonsei.ac.kr

*** Director, R&D Center, Newrun System Co. Ltd., 506 Ace techno Tower, Gasan-dong, Kumchon-gu, Seoul, South Korea, E-mail: ash@newrun.co.kr

**** CTO, UCUBE Inc., 105 Sungdong-gangbyunparkvill, Guii-3dong, Gwangjin-gu, Seoul, South Korea, E-mail: hspark@ucube.kr

어 활용에 관심을 갖게 되었다.

공개소프트웨어가 등장한지 30여년이 지난 지금 공개소프트웨어는 다양한 분야에서 활용되고 있다. 공개소프트웨어(Open Source Software)는 개인사무용 소프트웨어에서 임베디드 소프트웨어(Embedded Software)에 이르기까지 다양한 영역에서 사용되어 지고 있다. 스마트폰 운영체제(Operating System)인 안드로이드도 공개소프트웨어이다.

대표적인 공개소프트웨어 LAMP(Linux, Apach, MySql, Php) 사용률은 기존 유명벤더 사유소프트웨어보다 많다. 소스포지닷넷(sourceforge.net)에는 2011년 11월 현재 약 30만개 프로젝트가 진행되고 있고 하루에 다운로드수가 2백만 회를 기록하는 세계 최대 오픈소스 프로젝트 커뮤니티 사이트로 소프트웨어 소스코드와 함께 공개되어 있다(Ko & Koo, 2012).

공개소프트웨어는 참여, 공유, 개방이라는 메커니즘(작동원리)에 의해 동작한다. 이러한 선순환구조 속에서 지속적으로 개선된 공개소프트웨어는 사유소프트웨어만큼 완성도가 높다. 기업공급자가 소멸할 경우, 함께 소멸되는 사유소프트웨어와 달리 소스가 오픈되어 있어 수명이 길다. Debian GNU/Linux버전 4.0를 전통적인 방법으로 개발할 경우, 연 73,000명의 개발자들이 참여하여 약2억8천3백만 줄의 소스코드를 개발해야하고, 약 54억 달러 비용이 소요될 것이다(Kim et al., 2009, Amor et al., 2007). 이는 특정 사유소프트웨어 업체만 보유하고 있는 고난이도 기술을 여러 공개소프트웨어 개발자를 통해 구현할 수 있다는 것을 의미한다. 이처럼 공개소프트웨어는 수많은 소프트웨어 개발자들이 자발적으로 참여하고 지식을 공유해서 이를 통해 고급 소프트웨어를 만들 수 있다.

그러나 소스포지닷넷에 게시된 공개소프트웨어들은 극히 일부 완성도 높은 공개소프트웨어를 제외한 대부분 공개소프트웨어는 수정 없이 바로 사용되기 보다는 자사 비즈니스 요구에 적합한 형태로 수정해서 이용하고 있다. 즉, LAMP 등 일부 완성도 높은 공개소프트웨어를 제외한 대다수 공개소프트웨어는 자사에 맞게 사용하기 위해서는 소스코드를 수정과 변경을 해야 한다. 이러한 이유 때문에 소스코드가 있어도 고난이도 기술이기 때문에 전문 인력이 없는 중소기업과 소프트웨어 개발업체들은 공개소프트웨어를 활용하는데 어려움이 있다.

본 연구는 B2B(Business to Business) 소프트웨어 시장에서 공개소프트웨어 중개자 존재 필요성에 대한 연구를 하고자 한다. 이를 위해 공개소프트웨어 이해와 소프트웨어 유통과 진화 과정을 먼저 살펴본다. 그 다음으로 본 연구에서 직접 제작한 웹크롤러(검색로봇)를 가지고 소스포지닷넷에서 데이터를 수집해서 분석하고 구글 앱스토어 분석을 통해 소프트웨어 유통에서 공개소프트웨어 중개자 필요성과 가져야할 특성을 살펴보고자 한다.

2. 이론적 배경

2.1. 공개 소프트웨어 이해

컴퓨터가 처음 보급되었을 당시 소프트웨어는 무료로 배포되었다. 이때 소프트웨어 개발자들은 소프트웨어 자체가 아닌 프로그래밍 작업에 대한 대가를 받았고, 기업소유 소프트웨어까지 자유롭게 복제와 수정을 했다, 1980년대 들어 소프트웨어는 사적 소유 대상이 되면서 더 이상 자유로운 공유가 어려워졌다. 그러던 중 1983년 MIT연구원 리차드스톨만(Richard Stallman)은 '소프트웨어는 특정인만 사용하는 것이 아니라 누구라도 자유롭게 사용해야 한다' 는 기치아래 자유소프트웨어(free software)운동을 시작하였다. 자유소프트웨어라는 단어의 정의는 정보통신부 공개소프트웨어 가이드(Ministry of Information and Communication, 2006)를 통하여 자유 소프트웨어를 포함하여 오픈소스 소프트웨어를 공개소프트웨어라는 용어로 사용하고 있어, 본 연구에서도 공개소프트웨어라는 용어로 통일해서 사용하고자 한다.

사유소프트웨어(proprietary software)와 자유소프트웨어는 정보재의 사적·공적 소유, 정보 독점·공유라는 관점에서 대립된다. Microsoft는 그들이 가지고 있는 윈도우 운영체제가 시장 지배력을 견고히 하고, 핵심 소프트웨어가 윈도우 기반에서 동작하기 때문에 이루어진 사적소유와 정보독점의 대표적인 사례이다. 자유소프트웨어는 프로그램, 소스코드, 그리고 정보를 자유롭게 공유하는 것을 의미한다.

자유소프트웨어재단은 '자유소프트웨어를 소프트웨어 사용, 복제, 배포 자유와 소스코드에 대한 접근을 통해 학습, 수정, 개선할 수 있는 자유를 부여하는 소프트웨어'라고 정의하고 있다. 자유소프트웨어재단(www.fsf.org)은 위키피디아에 따르면 1985년에 자유소프트웨어 생산과 보급을 장려하기 위해 세워졌다. 리처드스톨만이 세웠고, 주로 GNU 프로젝트에 관해 일한다. 이 재단은 그때그때 사용가능한 소프트웨어 배포보다는 주로 새로운 소프트웨어 개발에 초점을 맞춘다. 현재 공개소프트웨어는 자유 소프트웨어를 포함하는 넓은 의미로 사용되며 소스코드를 공개한다는 측면에서 Freeware나 Shareware와는 구별 된다. Freeware는 무료 사용이 가능하지만 소스코드는 미공개이고, Shareware도 일정기능 제약이 있으면서 무료사용이 가능하지만 소스코드는 미공개이다.

공개소프트웨어는 개인과 다수가 참여해서 무료로 소스코드와 소프트웨어를 제공하고 누구나 사용하고 소스코드가 공개되어 누구든지 수정이 가능한 장점이 있지만 사용허가(licence)가 존재하여 사용허가만 준수한다면 얼마든지 사용이 가능하다. 사유소프트웨어는 특정기업이 소프트웨어를 사용자에게 제공하고 그 대가로 사용료를 지불하고 사용할 수 있지만 소스코드는 수정할 수 없다. 공개소프트웨어는 소스코드를 자사 비즈니스 환경과 기술 환경에 맞게 수정할 수 있지만 그에 대한 기술지원을 제공받을 수는 없다 (<표 1>참조).

<표1> 공개소프트웨어와 사유소프트웨어 비교

구 분	참여자	제공형태	소스코드 공개여부
공개소프트웨어 (Open Source Software)	개인과 다수	원시코드, 실행파일 (사용허가에 따라 배포여부 결정)	공개
사유소프트웨어 (Proprietary Software)	특정기업	실행파일	미공개

공개소프트웨어는 1992년 리누스토발드(Linus Benedict Torvalds)가 Unix를 개인용 컴퓨터에서 사용하고자 만든 리눅스(Linux) 운영체제가 소스코드를 공개하면서 널리 알려졌다. 1995년 롭맥쿨(Rob McCool)이 아파치(Apache: 인터넷 핵심 구성요소 NCSA 웹서버 소프트웨어)를 오픈소소화하여 배포하면서 공개소프트웨어 사용은 더욱 증가했다. 이와 함께 동적으로 웹페이지를 만들기 위한 CGI(Common Gateway Interface) 기능이 포함된 PHP(Personal Hypertext Preprocessor: 프로그래밍 언어 일종으로위키피디아의 정의에 따르면 원래는 동적 웹 페이지를 만들기 위해 설계되었으며 이를 구현하기 위해 PHP로 작성된 코드를 HTML 소스 문서 안에

넣으면 PHP 처리 기능이 있는 웹 서버에서 해당 코드를 인식하여 작성자가 원하는 웹 페이지를 생성한다. PHP는 아파치 웹서버와 함께 동작한다.)와 DBMS(Database Management System)인 MySQL 등이 아파치와 함께 사용되면서 쉽게 홈페이지 구축이 가능해져 공개소프트웨어 사용이 폭발적으로 증가했다.

1998년 공개소프트웨어 활성화와 인증을 위하여 OSI(Open Source Initiative: 1998년 2월에 브루스페렌스와에릭레이먼드가네스케이프커뮤니케이션스코퍼레이션이 대표 제품인 네스케이프커뮤니케이터에 대한 소스 코드를 출시한 데 자극을 받아 설립함)가 결성 되었다. OSI는 캘리포니아주에 등록된 시민단체로서 주요 활동으로는 오픈소스의 정의를 정립하고, 공개소프트웨어 관련 주요 정책을 조정하는 역할을 하고 있으며, 공개소프트웨어에 관련하여 대중들에게 널리 알라는 역할을 하고 있다. OSI의 활동에 의해 공개소프트웨어 개념이 정립되고 사용허가 폭이 넓어져 더욱 많이 이용하게 되었다. 공개소프트웨어는 소프트웨어 혹은 하드웨어 제작자의 권리를 지키면서 소스코드를 누구나 열람할 수 있도록 한 소프트웨어를 의미한다(Wikipedia, 2012).

<표 2> 공개소프트웨어 조건

조건	설명
자유로운 공개소프트웨어 재배포	공개소프트웨어 사용허가(licence)는 몇 개의 다른 출처로부터 모아진 프로그램들로 구성된 집합 저작물 형태의 배포판 일부로 소프트웨어를 판매하거나 무상 배포하는 것을 제한해서는 안 된다.
소스코드 제공	공개소프트웨어는 소스코드가 포함되어야 하며, 컴파일된형태 뿐 아니라 소스코드 배포도 허용되어야 한다. 만약 소스코드가 함께 제공되지 않는 제품이 있다면 소스코드를 복제하는데 필요한 합당한 비용만으로 소스코드를 구할 수 있도록 널리 알려진 방법이 제공되어야 한다.
파생 저작물 허용	공개소프트웨어 사용허가는 프로그램 개작과 2차 프로그램 창작이 허용되어야 한다. 이러한 파생 저작물들이 원 프로그램에 적용된 것과 동일한 사용 허가 규정에 따라 배포되는 것을 허용해야 한다.
저작자 소스코드 원형 유지	공개소프트웨어 사용허가는 바이너리 생성시점에서 프로그램을 수정할 목적으로 소스코드를 수반한 '패치 파일' 배포를 허용한 경우에 한해서 패치로 인해 변경된 소스코드 배포를 제한할 수 있다. 그러나 이 경우에도 변경된 소스코드를 통해 만들어진 소프트웨어 배포는 명시적으로 허용해야 한다.
사용분야, 개인과 단체에 차별 금지	공개소프트웨어 사용허가는 프로그램이 특정분야에서 사용되는 것을 금지하는 제한을 설정해서는 안 된다.
사용허가 배포	프로그램 권리는 배포에 따른 각 단계에서 배포자에 따른 별도 사용 허가 없이도 프로그램을 재배포 받은 모든 사람에게 동일하게 인정되어야 한다.

Source: Open Source Initiative (2012)

소스코드를 공개한다고 해서 공개소프트웨어가 되는 것이 아니다. 공개소프트웨어는 OSI가 제시한 조건을 만족해야 된다(<표 2> 참조). 예를 들어, 마이크로소프트는 극소수 고객(정부, 다국적 기업, 대학교, 연구소)에게 마이크로소프트 윈도우 소스코드를 공개했다. 오로지 보안 유지를 위해서만 소스를 직접 수정할 수 있으며, 그 수정본을재배포하는 것은 금지되어 있다. 이것은 공개소프트웨어의 의의에 어긋나므로 공개소프트웨어라 부르지 않는다. 따라서 공개소프트웨어가 참여, 기여, 공개라는 기본정신을 유지하면서 발전하려면 원활한 유통구조를 가져야 한다. 특히, 공개소프트웨어 유통은 각 참여자(생산자, 중개자, 그리고 사용자)가 가치사슬에서 모두가 창출된 부가가치를 공유할 수 있는 선순환구조를 가져야 한다.

공개소프트웨어 장점은 적은 초기 개발비용, 유연한 개발, S/W 간에 상호연동성을 보장하는 오픈포맷과 프로토콜, 강력한 네트워킹 기능 지원이 있다. 그러나 애플리케이션의 부족, 빈약하고 체계적이지 못한 문서, 불확실한 개발 로드맵 등과 같은 단점으로 인해 공개소프트웨어 도입에 걸림돌이 되고 있다(Kwon et al., 2008). 공개소프트웨어의 시장점유율을 높이기 위해서는 사유소프트웨어 사용자 집단보다는 공개소프트웨어 사용자가 전문적인 기술을 보유하고 있어 사용자 요구에 따라 적절히 보완할 수 있는 능력이 있어야 한다(Lin, 2008). 그렇지 못한 경우 유지보수의 어려움과 비용에 대한 우려로 말미암아 도입을 꺼릴 수밖에 없다.

공개소프트웨어는 인터넷 커뮤니티를 통해 무상으로 많은 소프트웨어 개발자들이 소프트웨어 개발과 디버깅(Debugging: 디버깅은 컴퓨터 프로그램 상 오류를 찾아내어 바로잡는 과정임)에 참여하는 협력방식을 통해 공개되고 있다. 최대 공개소프트웨어 개발자 커뮤니티 소스포지닷넷은 미국 Geeknet사 운영 중이며 2011년 11월 기준으로 약 30만개 프로젝트가 진행 중에 있으며 200만 명 이상 개발자가 가입하여 활동하고 있다. 프로젝트 카테고리는 데이터베이스, 보안, 시스템, 교육, 보안, 시스템 관리 등 다양한 분야로 구성되어 있다.

공개소프트웨어 커뮤니티는 참여목적과 이용대상에 따라 분류하면 개발자 커뮤니티와 사용자 커뮤니티로 구분된다. 개발자 커뮤니티는 개발자가 직접 참여하여 프로젝트 형식으로 개발 산출물(데이터)을 공개소프트웨어 소스코드와 문서를 공유하여 공개소프트웨어 활성화에 기여하는 커뮤니티이다. 사용자 커뮤니티는 공개소프트웨어의 사용과 관련된 정보를 교환하는 커뮤니티로 공개소프트웨어 활성화에 간접 기여를 하고 있다. 국내 대표적인 사용자 커뮤니티는 한국리눅스문서화프로젝트(KLDP.org)가 있다. 1996년 개인 홈페이지로 출발하여 리눅스 관련 문서를 한글로 번역해서 인터넷으로 제공하는 것을 주 활동으로 하고 있고, 지금은 문서화뿐만 아니라 개발자 커뮤니티와 프로젝트 호스팅 등 다양한 활동을 하고 있다.

레드햇·조지아공과대학교 '오픈소스 인덱스' 공동연구보고서(CIOBIZ, 2010)에서 공개소프트웨어 활용률 순위를 보면(<표 3>참조), 프랑스 1위, 스페인 2위, 미국 9위, 일본 14위, 중국 15위인데 IT강국인 대한민국은 20위로 매우 낮은 수준에 있다. 더 심각한 것은 정부는 5위이지만 업계는 41위이고 커뮤니티/교육은 26위이다. 필란드는 활용률이 5위(정부 19위, 업계 1위, 커뮤니티/교육 18위)로 업계주도로 활용되고 있지만 대한민국은 정부주도로 추진되고 있어 상반된 실정이다(Nam et al., 2006).국내 공개소프트웨어 활용률이 정부와 업계간 큰 차이를 보이고 있는 것은 아직 국내 공개소프트웨어 시장이 낮은 수준에 있고 기업들이 공개소프트웨어에 대한 보안결함 오해 등으로 유용성에 대한 인식이 낮고 자사 내부인력으로 공개소프트웨어를 활용하는데 기술능력이 낮기 때문인 것으로 판단된다. Kwon et al.(2008)의 공개소프트웨어 인식조사에서도 사용자들이 도입 여부와 상관없이 공개소프트웨어에 낮은 신뢰성을 가지고 있고, 기존 시스템과의 호환성에도 문제가 있을 것이라는 편견이 존재하는 것으로 나타났다.

공개소프트웨어를 활용한 전 세계 IT 와 소프트웨어 시장이 급격하게 성장하고 있음에도 대한민국은 특정 품목과 대기업을 제외하면 공개소프트웨어를 활용한 소프트웨어 개발이 저조하다는 것을 알 수 있다.

<표 3> 국가별 오픈소스 이용환경과 활용률 순위

국가	오픈소스 활용률	정부	업계	커뮤니티/교육	오픈소스 환경	정부	업계	커뮤니티 교육
프랑스	1	1	25	3	15	18	18	17
스페인	2	2	22	10	20	20	29	21
독일	3	4	19	5	16	14	23	18
호주	4	14	4	11	11	7	16	10
핀란드	5	19	1	18	5	12	5	8
영국	6	7	15	7	8	8	13	4
노르웨이	7	14	2	29	3	4	17	3
에스토니아	8	45	5	1	21	21	49	14
미국	9	28	13	2	2	3	7	2
덴마크	10	12	8	31	4	2	9	6
이탈리아	11	8	20	15	22	25	34	23
브라질	12	3	43	14	45	40	65	42
네덜란드	13	19	7	17	9	5	12	7
일본	14	11	27	6	13	6	14	11
중국	15	6	69	4	50	70	8	59
싱가포르	16	34	3	21	14	35	1	22
벨기에	17	10	18	32	18	17	22	19
스웨덴	18	22	11	20	1	1	4	1
아일랜드	19	32	14	9	19	19	11	24
대한민국	20	5	41	26	12	16	10	12
뉴질랜드	21	45	10	13	6	9	6	9
스위스	22	34	9	28	7	10	3	13

공개소프트웨어 덕택에 이제는 미국이외의 나라들도 시스템 소프트웨어 기술을 확보할 수 있는 기회를 갖게 되었다. 공개소프트웨어를 활성화시키기 위해서는 인력양성과 시장형성을 해야 한다(Patrenko et al., 2007). 공개소프트웨어와 시스템소프트웨어는 서로 맞물리는 관계로 선순환 구조를 만들기 위해서는 공개소프트웨어 중개자가 시장 형성을 해야 한다. 즉, 소프트웨어 개발자가 중개자와 함께 일정 수익을 공유하고 지속적인 개선을 하고 사용자가 공개소프트웨어를 저렴한 가격에 쉽게 이용할 수 있으면 된다.

전산 분야에서 유명한 농담 중에 '야크 털깎기(Yak shaving)'라는 말은 원래 하려던 일은 나무를 깎으려던 것인데, 도끼가 더 잘 들면 나무를 더 빨리 벨 텐데 해서 도끼날을 세우다가, 좋은 숫돌이 있으면 도끼날을 더 잘 세울 텐데 해서 좋은 숫돌이 있는 곳을 수소문하고, 저 멀리 어디 있단 이야기를 들어 야크를 타고 가려다가 야크 털이 길어 털을 깎고 마는 비유에서 알 수 있듯이 주어진 도구와 시간으로 본래 목적에 부합하는 곳에 시간과 노력을 들이는 것이 필요한 시기이다. 결국 소프트웨어는 비즈니스 목적을 달성하기 위한 도구일 뿐이기 때문이다.

2.2. 공개소프트웨어 유통과 진화

2.2.1. 국내 소프트웨어 유통과 진화

1990년대 전, 국내 소프트웨어 유통은 하드웨어를 공급할 때 소프트웨어를 함께 무상 제공 방식으로 이루어져 왔다. 소프트웨어는 하드웨어를 제조하는 대기업을 중심으로 발전하여 왔다. 그 이유는 소프트웨어 전문업체가 없었기 때문에 컴퓨터시스템을 도입할 때 하드웨어 제공업체가 소프트웨어까지 일괄적으로 시스템을 구성하여 납품하는 Turn-key방식으로 소프트웨어가 공급되었기 때문이다. 하드웨어 제공업체가 납품, 설치, 운영, 교육, 유지보수에 이르기까지 모든 부분을 전담하여 처리했다. 그 당시 소프트웨어는 하드웨어에 종속되어 있고 하드웨어 공급업체가 자체개발한 소프트웨어를 공급하기 때문에 범용성이 낮고 도입비용과 유지보수 비용이 상당히 고가였다.

1990년 이후 전문 소프트웨어 업체(일명: 소프트웨어 하우스)가 만든 특정 하드웨어에 종속되지 않은 범용 소프트웨어(예, 워드프로세스)가 나타났다. 범용소프트웨어는 주로 패키지 소프트웨어 형태의 사무용 소프트웨어였다. 패키지 소프트웨어는 상대적으로 가격이 저렴하였으며, 이용도가 높은 프로그램이나 업무 연관성이 높은 프로그램을 일괄적으로 묶어서 상품을 제공했다. 국내 소프트웨어 시장은 연평균 30% 이상 높은 성장을 하였지만 해외 소프트웨어들이 국내시장을 지배하고 있었다. 해외 소프트웨어는 기업용 소프트웨어를 Turn-key방식으로 보급했고 국내 소프트웨어는 상대적으로 가격이 저렴한 패키지 소프트웨어를 주로 보급했다. 컴퓨팅 능력이 빠르게 향상되어 서버급 컴퓨터들이 개인용 컴퓨터로 사용 가능해졌고 비즈니스 환경도 빠르게 변화함에 따라 국내 소프트웨어업체들은 더욱 어려움을 겪었다.

1990년대 중후반부터 인터넷 등장은 소프트웨어 유통흐름에 많은 변화를 가져왔다. 이때부터 고가 하드웨어와 소프트웨어를 도입하지 않고도 네트워크 인프라를 이용하여 다양한 기업 솔루션(예, ERP)을 사용할 수 있는 애플리케이션 임대 서비스인 ASP (Application Service Provider)방식이 등장했다. 중소기업들이 고가 소프트웨어 도입비용을 줄일 수 있는 장점 때문에 많이 도입하기 시작했다. ASP 방식은 많은 기업들에게 소프트웨어를 구매하는 방식에서 아웃소싱 방식으로 변화하게 했다.

이제는 ASP 방식을 넘어 SaaS(Software as a Service) 방식으로 전환되고 있다. SaaS는 기존 ASP를 확장한 개념이다. SaaS는 소프트웨어 사용 고객을 대상으로 소프트웨어를 서비스처럼 사용하고 비용을 지불하는 방식이다. SaaS는 전자상거래 개념이 강하고, ASP는 아웃소싱 개념이 강하다. ASP는 별도 서버와 애플리케이션을 구매하지 않고 비용·시간·관리 인력 부담이 없이 중소기업이 도입할 수 있었다. 지금까지 소프트웨어 유통 흐름를 보면 소프트웨어 유통은 공급자 중심에서 수요자 중심으로 소프트웨어 자체를 공급하는 개념에서 서비스를 위한 소프트웨어 개념으로 유통방식이 변화하고 있는 알 수 있다(<표 4>참조).

<표 4> 소프트웨어 유통 진화

구분	방식	설명	이용방식
1990년대 이전	Turn-key	컴퓨터 시스템 공급자가 하드웨어, 소프트웨어, 사용자 교육, 그리고 사후 관리까지 제공하고 책임을 지는 관리방식	구매
1990년대 전반	Package	OS, 오피스등, 이용도가 높은 프로그램이나 업무, 업종에 적합한 프로그램을 묶어서 상품으로 제공하는 방식	구매
1990년대 후반	ASP	고가 하드웨어, 소프트웨어를 도입하지 않고도 네트워크 인프라를 이용하여 다양한 정보화 솔루션을 사용할 수 있는 어플리케이션 임대 서비스 방식	임대
2000년대 후반	SaaS	공급업체가 하나 플랫폼을 이용해 다수 고객에게 SW서비스를 제공하고, 사용자는 이용한 만큼 돈을 지불하는 방식	종량제
2010년대 이후	Cloud	iCloud등 미디어 리소스를 서버에 놓고, 위치에 관계없이 소프트웨어를 사용	종량제

2.2.2. 공개소프트웨어 유통과 진화

소프트웨어는 프로그램과 프로그램 정보일체(개발, 운용, 유지, 보수)를 포함한다. 소프트웨어는 개발이 어렵고 많은 비용이 들지만 저렴한 비용으로 배포와 복제가 가능하다. 그리고 소스코드가 있으면 쉽게 수정이 가능하다. 초기 소프트웨어 개발자들은 공개소프트웨어 생산자이자 사용자였다. 공개소프트웨어 생산자는 전문 프로그래머들로 운영체제에 사용되는 소스코드를 개발하고 개발에 필요한 유틸리티들을 만들어 공유했다. 유틸리티 개발자들은 스스로 다른 개발자가 만든 유틸리티 사용자가 되기도 했다. 단순 사용자가 아닌 소스코드가 오픈된 유틸리티에 자신이 필요한 기능을 추가하여 재생산자의 역할을 동시에 했다. 소스코드가 오픈된 유틸리티는 인터넷 전신 ARPANET을 통해 소프트웨어 개발자들이 개인이 운영하는 파일전송 서버를 통해 배포했다.

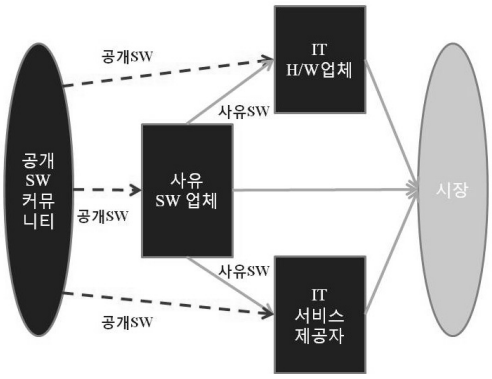
서서히 공개소프트웨어 유통은 소프트웨어 개발자 자신이 구축한 시스템에 의해 운영되었고 동일기술 주제와 관심사로 메일을 주고받던 개발자 모임은 주제별 커뮤니티로 발전하게 된다. 공개소프트웨어 생산자들은 개별수준에서 활동했지만 메일링시스템, 버그트래킹 시스템, 그리고 인터넷이 발전하면서 생산자 그룹 커뮤니티로 변화하면서 발전했다. 일부 규모가 큰 커뮤니티는 위원회를 구성하고 민간 기업으로부터 자금지원을 받아 커뮤니티 유지 비용으로 사용했다.

공개소프트웨어는 초기에 무료 사용하는 자유소프트웨어로 받아들여졌으나, 이후 소스코드와 프로그램 사용이 자유롭다는 것이지 비용을 청구할 수 없다는 것은 아니라는 의미로 인식하기 시작했다. 이때부터 공개소프트웨어 활용가치에 따라 비용 지불을 하는 공급자 라이선스 방식이 나타나기 시작했다. 공급자 라이선스 방식은 두 가지 유형이 있다. 첫 번째 유형은 한 개의 공개소프트웨어에 두개의 라이선스를 적용해서 2차 생산자가 공개소프트웨어를 수정하여 기능을 강화한 소프트웨어를 사용자에게 전달할 때 반드시 소스를 오픈하지 않고 판매가 용이할 수 있도록 하는 라이선스 방식이다. 두 번째 유형은 기존에 공개소프트웨어를 변경한 2차 생산자도 1차 생산자가 적용한 소스오픈 규정을 계속 지켜야 하는 의무를 강제하는 듀얼라이선스(dual licence) 방식이다. 이러한 공급자 라이선스 방식은 공개소프트웨어가 비즈니스화 하기 어려운 부분인 소스코드 공개 조건을 완화한 상업적 라이선스 적용을 한 방식이다.

사유소프트웨어의 유통은 소프트웨어 개발회사가 직접하거나, 유통업자가 소프트웨어와 하드웨어를 조합해서 판매한다. 하지만 공개 소프트웨어는 일반적으로 개별 제품에 대한 마케팅이 없다. 유통은 물리적으로 제품을 고객이 접근하여, 사용하고, 살 수 있도록 적정한 장소로 옮기는 것이다. 즉, 취합은 유통의 한 부분이며, 소프트웨어를 수정하지 않고 부가기능도 추가되지 않고 유통하는 것이다.사유소프트웨어에서 취합은 개발 과정의 한 부분이나, 공개소프트웨어는 유통업자에 의해 수행되고, 이것 역시 별도의 공개소프트웨어 사업 모델의 근간을 이룬다(천명권, 2002)

이처럼 공개소프트웨어와 사유소프트웨어는 유통과 진화과정이 다르다. 소프트웨어 유통 참여자는 소프트웨어업체(패키지와 솔루션 제공), 하드웨어업체, 서비스 제공자, 그리고 사용자가 있다. 서비스 제공자는 사유소프트웨어와 공개소프트웨어를 통합해서 고객맞춤화된 서비스를 제공한다. 사유소프트웨어는 각 참여자들이 사용자까지 전달하는 과정에서 소프트웨어를 변경할 수 없지만, 공개소프트웨어는 유통 참여자들이 사용자에게 전달하는 과정에서 자체소프트웨어와 결합을 통해 소프트웨어를 변경할 수 있다.

사유소프트웨어는 소프트웨어 지적재산을 독점 보유한 기업이 유통 독점을 갖는다. 하지만 공개소프트웨어는 지적재산을비독점하게 되므로 유통 독점을 보장받지 못한다. 사유소프트웨어 시장은 하드웨어업체와 서비스 제공업자가 소프트웨어업체가 제공하는 소프트웨어에 잠긴(Lock-in) 효과로 의존성이 높다. 하지만 공개소프트웨어 등장으로 인해 이 소프트웨어 유통구조가 변화하기 시작했다. 공개소프트웨어 등장 이후, IxV(Independent (Software/Hardware/Service) Vendor: ISV, IHV, ISV 등이 있음)가 공개소프트웨어를 이용해 소프트웨어를 단기간에 개발해서 제공하게 되었다. 또한, 소프트웨어 개발업체간 기술수준이 평준화되어 소규모 인력으로도 신속하게 소프트웨어 개발이 가능해지고, 소프트웨어 기술 습득이 용이해졌다(<그림 1>참조).



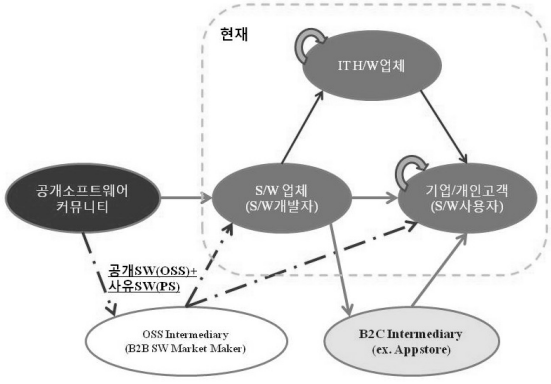
<그림 1> 공개소프트웨어 등장 전·후 유통구조

공개소프트웨어가 정보시스템 적용영역(정보시스템이 사용될 비즈니스 영역)과 구성 요소간 결합정도(예, 소프트웨어간, 솔루션간, 하드웨어간)에 따른 상호 구속력(예, library dependancy)이 변화한다. 상호구속력은 정보시스템 구성요소의 결합구조 가치를 의미한다. 결합구조 가치는 정보시스템 설계시 구체화 된다. 예를 들면, 웹서버 공개소프트웨어 아파치를 사용할 때와 사유소프트웨어 IIS(Internet Information Server)를 사용때 상호연관이 있는 라이브러리, 운영체제 등 구성요소 선택에 차이가 있다. 선택폭의 차이는 다양성 측면에서 정보시스템의 초기 가치가 달라진다. 이는 정보시스템 도입 기업이 특정 사유소프트웨어 벤더로부터 의존성이 낮아지기 때문에 독립영역이 발생하게 된다. 이는 사유소프트웨어 기업과 다른 하드웨어, 소프트웨어, 그리고 IT 서비스 기업간에 독립적 관계로 변화되는 것을 의미한다. 인터넷 포탈업체 네이버와 다음은 더 이상 오라클DBMS를 사용하지 않고 국산 공개소프트웨어 큐브리드(Cubrid.com)를 사용하고 있다. 이제 소프트웨어 시장은 사유소프트웨어의 높은 의존성에서 탈피하는 쪽으로 변화하고 있다. 사유소프트웨어가 제공하는 모든 기능을 제공하지 못하고 일부 기능은 제한되어 있지만 많은 기업들에게 가치 있는 공개소프트웨어 제공함으로써 사유소프트웨어 의존성으로부터 탈피할 수 있게 되었다. 특히, 스마트폰과클라우드 컴퓨팅 기술 등장으로 이러한 유통체계의 재편이 더욱 빠르게 진행되고 있다.

공개소프트웨어 등장이후, 하드웨어업체와 IT서비스 제공업체들은 단순 사용자 역할뿐만 아니라 생산자 역할도 하는 프로슈머(Prosumer)로 변화하고 있다. 프로슈머란앨빈토플러 등 미래 학자들이 예견한 기업의 생산자(producer)와 소비자(consumer)를 합성한 말로써, 소비자가 소비는 물론 제품개발, 유통과정에까지 직접 참여하는 '생산적 소비자'로 거듭나는 의미다. 글로벌 IT 업체 IBM은 사유소프트웨어 단순 사용자에서 Linux와 Eclipse(소프트웨어 저작도구) 같은 공개소프트웨어를 생산하는 프로슈머(Prosumer)이다.

인터넷 검색회사 구글(Google)도 공개소프트웨어를 사용하면서 Big Filesystem과 같은 공개소프트웨어를 개발해서 시장에 공개하고 있다. 공개소프트웨어는 사유소프트웨어 특정 기능을 공개소프트웨어 커뮤니티 소프트웨어 개발자들에 의해 개발되어 공유되고 있고, 계속 진화하면서 사유소프트웨어 기능과 안정성을 보완하고 있다.

현재 사유소프트웨어는 생산자가 구축한 유통채널을 통해 공급되고, 공개소프트웨어는 공개소프트웨어 커뮤니티를 통해 공개되고 있다. 1980년대 공개소프트웨어가 출현할 때 공개소프트웨어 특성 때문에 부정적인 시각이 있었지만, 현재 공개소프트웨어는 사유소프트웨어 대체재가 아닌 보완재로서 그 역할을 하고 있다. 공개소프트웨어와 사유소프트웨어는 개발방식과 제품성격(예, 패키지: 오피스웨어, 솔루션:ERP)에 따라 유통방식에 차이가 있다. B2C(Business to Customer) 소프트웨어 시장에서 대표적인 중개자로 애플(Apple) 앱스토어와 안드로이드 안드로이드마켓 등이 있다. 따라서 공개소프트웨어는 생산자와 사용자 사이에 중개자가 등장한 유통구조가 구축되어야 발전적 방향으로 진화할 수 있을 것으로 판단된다(<그림 2>참조).



<그림 2> 공개소프트웨어 중개자와 유통구조

소프트웨어 유통시장에서 독점지배력을 가진 기업이 자사 비핵심영역 신제품을 독점하고 있는 유통채널에서 경쟁제품을 배제함으로써 시장 지배력을 강화했다. 이는 소프트웨어 시장에서 핵심영역 제품과 신제품간에 기능적 결합력을 강화해 사용성을 높여 시장 지배력을 유지한 것이다. 애플 아이폰 사례에서 알 수 있듯이 하드웨어, 소프트웨어, 운영체제, 그리고 핵심 응용프로그램을 직접공급하고 있으며, 관련 응용프로그램간 기능적 결합을 넘어선 사용자의 사용성을 제공하면서 시장지배력을 강화하고 있다. 이처럼 시장지배력을 갖은 기업이 유통 독점을 통해 사용자의 자율 선택을 방해한 것이다. 공개소프트웨어는 자유로운 사용과 배포와 같은 공유 개념을 가지고 있어 유통 독점이 불가능하다. 유통 독점이 불가능해진다는 것은 소프트웨어 유통이 활성화된다는 것을 의미한다.

지금까지 공개소프트웨어 등장에 따른 소프트웨어 유통 변화를 살펴보았다. 공개소프트웨어도 사유소프트웨어와 같이 복잡한 구조체 특성을 가지고 있다. 공개소프트웨어는 소프트웨어 개발자나 사용자가 수정 없이 바로 사용은 어렵다. 그래서 자사 환경에 맞게 소스코드를 수정해야 한다. 또한 공개소프트웨어를 처음 사용하는 소프트웨어 개발자들과 사용자들은 시간과 비용에 대한 예측이 어렵고 정보시스템 전략수립도 쉽지 않다. 따라서 소프트웨어 개발자가공개소프트웨어에 대한 활용을 높이기 위해서는 이러한 문제를 해결해줄 공개소프트웨어 중개자가 필요하다.

3. 소스포지닷넷 분석

3.1. 소스포지닷넷 분석

공개소프트웨어 진화과정을 파악하기 위하여 소스포지닷넷(sourceforge.net) 분석하였다. 소스포지닷넷은2천7백만명 이상 개발자가 참여하고 있으며 2011년 11월 현재 약 30만개 프로젝트가 진행되고 있고 하루에 다운로드수가 2백만 회를 기록하는 세계 최대 오픈소스 프로젝트 커뮤니티이다. 본 연구는 소스포지닷넷을 분석하기 위하여 웹크롤러(검색로봇)을 자체 제작했다. 웹크롤러는볼랜드델파이(Borland Delphi) 7.0을 이용하여 만들었다. 웹크롤러를 이용한 자료 수집은 1개 자료 취득시간이 약 3초 소요되었으며 전체 자료 취득에 약 9일간(2011년 10월 25일~11월 2일)이며 233,021개 자료를 수집했다. 수집된 자료내용은 카테고리, 프로젝트명, 프로젝트URL, 생성일자, 최근 변동일자이며, 등록일 기준으로 19개 카테고리별(소스포지닷넷 분류기준)로 구분하였다(<표 5> 참조).

수집된 자료는 Google 모션차트(Motion Chart: 구글에서 제공하는 시간 경과에 따라 여러 지표에 대한 정보를 표시해 주는 동적 차트) 소스코드를 만들어서 분석했다. 소스포지닷넷에 등록되어있는 공개소프트웨어 중 소프트웨어개발 분야(38,111개)가 가장 많고, 그 다음으로 인터넷(33,286개)과 시스템(26,475개) 분야가 많을 것을 볼 수 있다(<그림 3>참조). 그동안 개발자를 위한 개발도구와 시스템 인프라 구축과 관련된 소프트웨어가 주를 이루고 있다. 그러나 최근에는 모바일 분야가 새롭게 등장했고, 개인용이나 특정 비즈니스 문제를 해결하고자 하는 비즈니스 솔루션이 나타났다.

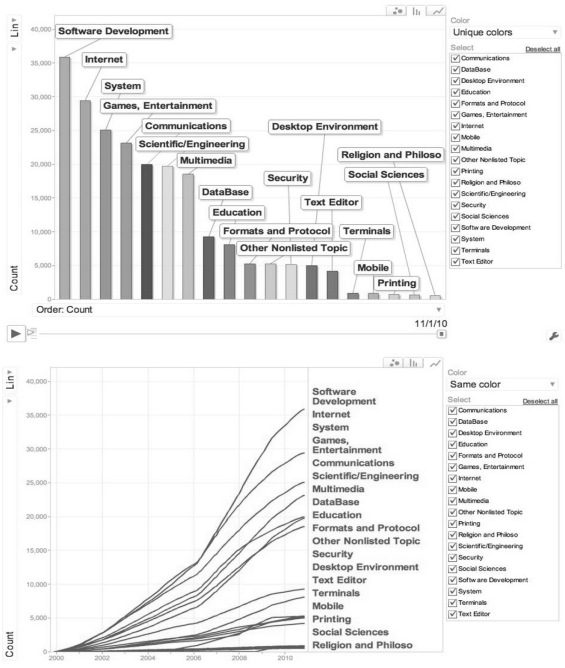
<그림 3>, <그림 4>에서 공개소프트웨어 년도별 등록추이를 분석하면, 2006년도를 기점으로 공개소프트웨어 등록이 가파른 상승세를 보이고 있다. 그러나 2009년도를 기점으로 소프트웨어 개발, 인터넷, 게임 등을 제외하고는 공개소프트웨어의 등록 증가율이 줄어들고 있는 것을 볼 수 있다. 이러한 현상은 차세대 플랫폼인 스마트폰에 관련한 애플과 안드로이드 어플리케이션 스토어에 개발자가 몰리면서 이런 현상이 나타났다고 추정된다.

<표 5> 분석한 오픈소스 소프트웨어의 인구통계학적 특성

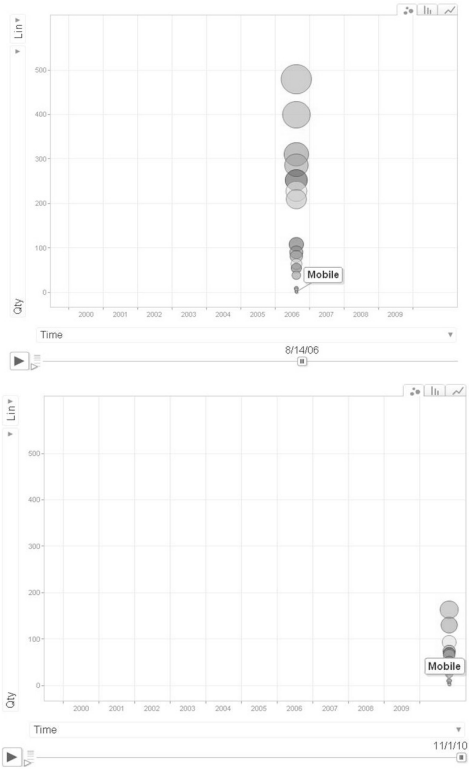
카테고리	등록된 S/W 갯수(개)	카테고리	등록된S/W 갯수(개)
소프트웨어 개발	38,111	기타	5,329
인터넷	33,286	보안	5,120
시스템	26,745	데스크탑 환경	5,031
게임	25,051	텍스트 에디터	4,582
커뮤니케이션	19,982	터미널 관련	2,328
과학/엔지니어링	19,911	모바일 관련	1,987
멀티미디어	18,937	프린터 관련	1,652
데이터베이스	9,286	사회과학 관련	932
교육	8,927	종교 및 철학	214
프로토콜	5,610		

이러 현상은 구글트렌드(google trend) 분석결과에서도 같은 추세에 있다는 것을 알 수 있다. 구글트렌드는구글에 사용된 검색어를 지역과 날짜별로 통계를 내서 웹사이트나 키워드의 트래픽 성

향을 비교해 볼 수 있게 해주는 구글웹서비스 중 하나다. 구글트렌드는 아직 국문화 되지 않은 서비스이다. '키워드'는 많은 사람들의 관심거리, 화젯거리와 일치하기 때문에 시장 동향에 관한 정보나 가까운 미래 예측에 유용하게 사용할 수 있다.

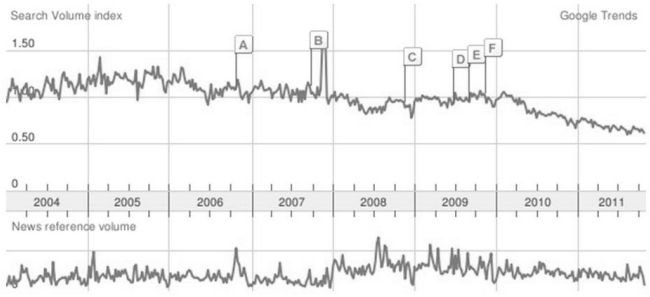


<그림 3>소스포지닷넷의 공개소프트웨어 분야별 분포도와 분야별 등록추이

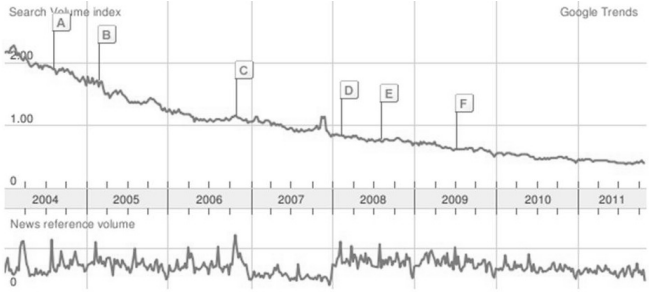


<그림 4>소스포지닷넷의 공개소프트웨어 분야별/년도별 등록추이(2008년, 2010년)

<그림 5>구글트렌드(google trend)에 "Open Source Software" 키워드를 입력하여 분석한 결과를 보면, 2009년도 기점으로 검색키워드가 서서히 줄어드는 추세에 있는 것을 볼 수 있다. <그림 6> "Linux" 키워드를 분석해보아도 같은 추세에 있는 것을 볼 수 있다. 하지만 그래프 하단의 참조관련 트래픽을 보면 꾸준한 트래픽을 유지하고 있는 것은 보편적으로 많이 보급되어 활용되고 있다는 것을 알 수 있다. 이러한 추세에 대해 본 연구는 애플 앱스토어가 급격히 성장하면서 관심이 모바일과스마트폰 어플리케이션으로 옮겨갔기 때문에 이런 현상이 나타난 것으로 추정한다.



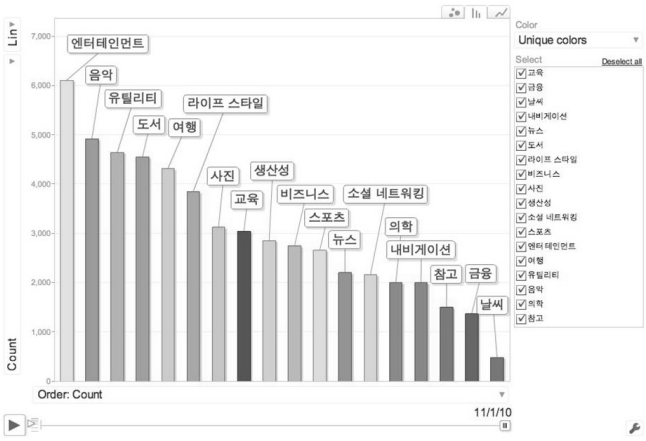
<그림 5> Google Trends 분석: "Open Source Software" 키워드



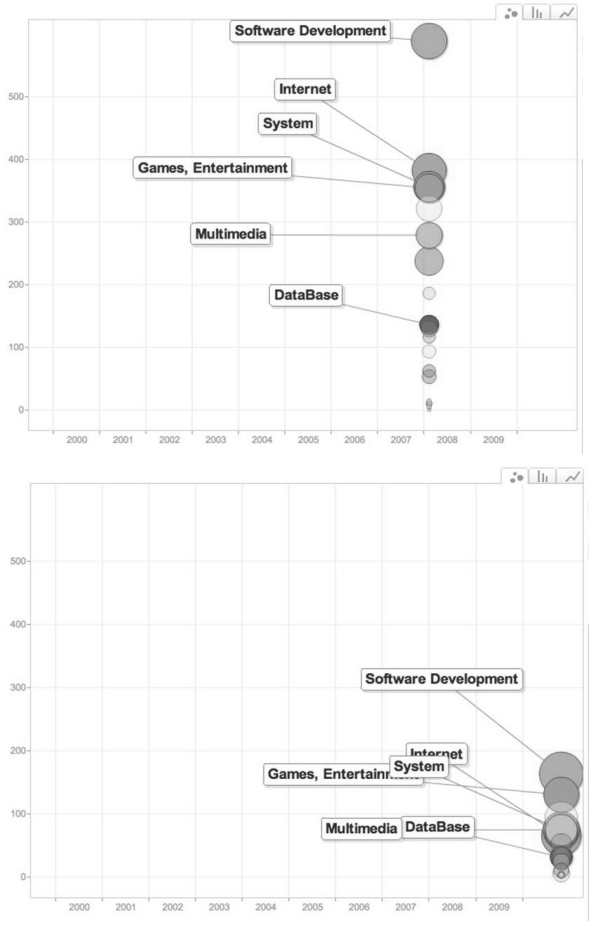
<그림 6> Google Trends 분석: "Linux" 키워드

소스포지닷넷에 게시된 공개소프트웨어와 구글트렌드 분석 결과, 애플 앱스토어의 등장으로 2008년부터 달라지고 있는 것을 알 수 있었다. 이에 본 연구는 애플 앱스토어가 지난 4년간 어떻게 진화되고 있는지를 살펴보기로 했다. 그래서 본 연구는 애플 앱스토어를2008년부터 2011년 10월까지 올라와 있는 소프트웨어를 추가 분석했다.

애플 앱스토어는 각 국가별로 앱스토어가 구분되어 있는데 본 연구는 애플 한국 앱스트어를 가지고 분석했다. 한국 애플 앱스토어 경우는 2011년 10월 기준으로 62,254개 소프트웨어가 등록되어 있다. 분야별 소프트웨어 분포를 보면 <그림 7>과 같다. 분석결과, 애플 분류기준에 따라 살펴보면 엔터테인먼트, 음악, 유틸리티 순으로 소프트웨어가 등록되어있다. 이는 스마트폰 전용 소프트웨어 특성과 개인화 특성으로 인해 주로 개인 관련된 소프트웨어가 주를 이루고 있는 것으로 판단된다. 아이폰앱 리뷰사이트 148Apps (148apps.biz)에서 발표한 2011년 5월에 발표한 자료에 따르면 앱 개발자는 85,569명, 개발자당 평균 앱의 수는 4.6개, 전체 앱의 개수는 50만개가 넘었다고 발표하고 있다.

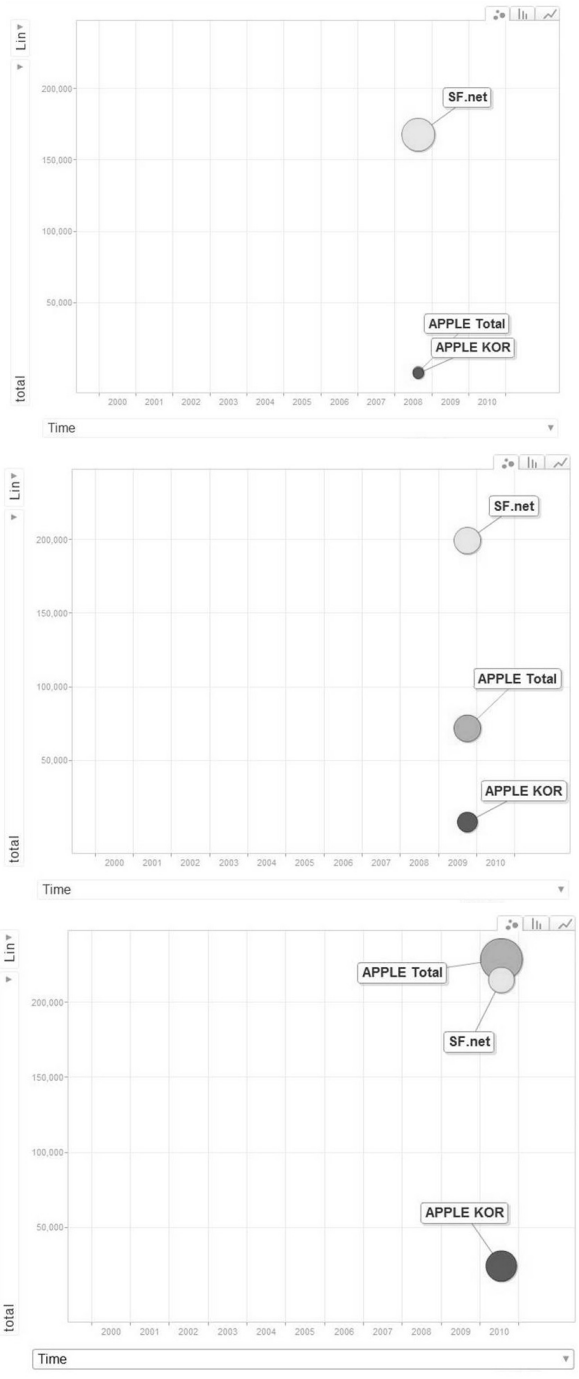


<그림 7> 애플 한국 앱스토어 소프트웨어 분야별 분포도



<그림 8>소스포지닷넷의 월 생산량(좌 2008년도, 우 2010년도)

<그림 8>에서 소스포지닷넷 생산량의 변화추이를 보면 2008년도 까지는 전반적으로 일정한 분포로 생산이 이루어지고 있었으나 2009년 하반기 이후에는 생산량이 급격히 감소하는 것을 알 수 있다. 특히 소프트웨어 개발, 인터넷, 엔터테인먼트, 멀티미디어, 그리고 시스템 분야 공개소프트웨어 월 생산량이 급격하게 줄어든 것을 볼 수 있다.



<그림 9> 2008년~2010년도 월 생산량비교
(원의크기: 월 생산량, 세로축: 전체수량)

그러나 <그림 9>애플 앱스토어월생산량을 보면 2008년에서 2009년으로 넘어가면서 급격하게 증가하고 있는 반면 소스포지닷넷 월 생산량은 줄어들고 있는 것을 볼 수 있다. 이 그림을 통해 소스포지닷넷과 애플 앱스토어의 생산량 증감에 대해 서로 상관관계가 있음을 알 수 있다. 2009년이후 앱스토어에 월 소프트웨어 등록건수 즉, 월 생산량이 급격하게 증가해서 2010년에서는 월생산량에서소스포지닷넷을 능가하고 년간 전체수량에서도 능가하게 된다. 전 세계에서 가장 큰 커뮤니티 사이트인 소스포지닷넷을 불과 2년 만에 월 생산량이 역전되고 년간 전체수량에 있어서도 능가했다. 이는 소프트웨어 개발자들이 스마트폰 어플리케이션으로

급격하게 이동하고 있다는 것을 의미한다.

지금까지 여러 관점에서 분석한 결과, 최근 4년 사이에 소스포지닷넷에 게시하던 소프트웨어 개발자들이 애플과 안드로이드앱스토어로 일부 옮겨갔다고 볼 수 있다. 이러한 현상이 생긴 이유는 애플과 안드로이드앱스토어 같은 B2C 소프트웨어 중개자 즉 이마켓플레이스가 등장했기 때문이다. 많은 소프트웨어 개발자들이 앱스토어라는 중개자를 통해 일반 사용자에게 손쉽게 접근할 수 있게 되었고, 스마트폰 이용자가 급격히 증가되면서 많은 수익을 창출할 수 있을 것으로 판단되었기 때문에 앱스토어를 선호하는 것으로 생각된다.

3.2. Linux 배포판 사례

공개소프트웨어대표적 성공은 리눅스배포판이다. 리눅스배포판은리눅스커널, GNU 소프트웨어와 여러 가지 자유 소프트웨어로 구성된 유닉스 계열 운영체제이다. 리눅스는리누스베네딕트투르발스(Linus Benedict Torvalds)가 만든 커널(kernel, 운영체제의 핵심 프로그램)에서 시작했다. 회사 차원에서 관리하고 배포하는 레드햇리눅스(Red Hat Linux), 우분투(Ubuntu), 수세 리눅스(SUSE Linux) 등도 있고, 커뮤니티 차원에서 관리하고 배포하는 데비안(debian), 젠투(Gentoo) 리눅스 등이 있다. 여러 소프트웨어를 모으고 시험하여 배포판을 만든다. 오늘날에는 전 세계적으로 약 300여 가지 배포판이 존재한다.초기 리눅스는 일반 사용자가 사용하는데 불편한 점이 많았고 전문적인 지식이 있어야만 사용이 가능했다. 1992년 초, 국내는 컴퓨터 전문가들과 잡지가 리눅스 관련 기술에 대한 분석을 소개하는 오프라인 중개자 역할을 하였다. 컴퓨터 잡지는 특별부록으로 슬랙웨어2x, 3x, 레드햇3.x, 리눅스 응용 프로그램 CD롬 타이틀을 제공하여 오프라인 중개자 역할을 하였다.

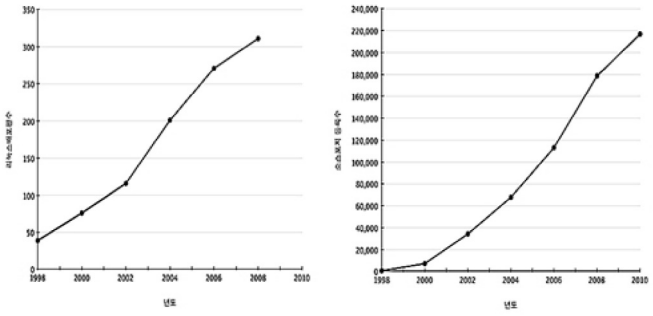
<표 6>리눅스배포판의년도별출시수

배포판	'93	'94	'95	'96	'97	'98	'99	'00	'01	'02	'03	'04	'05	'06	'07	'08	'09	'10	계
Debian	1						3	4	3		3	4		1					19
Fedora												2	2	1	2	3	1		11
Gentoo										1	1	1	5	2	1	1			12
Knoppix											9	10	6	6			1		32
RedHat		1	1	1	5	5	10	8	10	6	11	10	1	5			1		75
S.u.S.E		1				1		3			2	1		2					10
Slackware	1				1			1		4	2	10	7	6	1		2	1	36
Ubuntu								1	2	1	1	4	6	11	9	13	6	5	59
기타	1	1	6	3	3	3	5	2	7	6	6	8	7	2	7	3	4	4	82
총합계	3	3	7	4	9	9	18	19	22	18	35	50	34	36	20	20	15	10	332

Source: GNU/Linux Distro Timeline (2011)

1990년 중반, 사용하기에 쉽고 편리한 리눅스배포판이 등장했다. 리눅스배포판은 수 만개 공개소프트웨어중에 운영체제에 필요한 기본적인 공개소프트웨어프로젝트를 모아 사용자가 편리하게 사용할 수 있도록 했다. 슬랙웨어(slackware), 데비안(Debian), 그리고 레드햇(RedHat)과 같은 리눅스배포판이 등장하면서 최종 사용자들도 쉽게 구입하고 사용할 수 있게 되었다. 2001년부터 리눅스배포판은 더욱 다양한 사용자 요구에 대응하면서 빠르게 성장했다. 리눅스는 이처럼 중개자들에 의해 최종 사용자가 쉽고 편리하게 사용할 수 있게 되어, 리눅스배포판년도별 출시수가 증가하고, 리눅스 사용자도 급격하게 증가했다(<표 6>참조)

초기 리눅스는 사유소프트웨어(예, Window, OS/2)를 대체 할 수준이 아니었지만 공개소프트웨어 커뮤니티에 많은 개발자들이 참여하면서 이제는 사유소프트웨어를 대체할 수준까지 왔다. 소스포지닷넷에 게시된 공개소프트웨어를 배포판과비교하면 2002년 이후 급격히 증가하고 있다. 리눅스배포판이 출시되면서 관련 공개소프트웨어가 커뮤니티에 올라오고, 이를 다시 활용한 공개소프트웨어가 개선되어 올라오는 선순환 구조가 형성되어 리눅스배포판이 크게 성공하게 되었다(<그림 10>, <표 7>참조).



<그림 10>리눅스배포판과소스포지닷넷의 등록추이

<표 7>리눅스배포판과소스포지닷넷의 등록건수

년도	'98	'00	'02	'04	'06	'08	'10
배포판	39	76	116	201	271	311	336
소스포지	581	7,240	34,301	67,873	113,250	178,737	216,955

이러한 성공은 리눅스배포판 업체가 공개소프트웨어중개자로서 역할을 제대로 했기 때문이다. 공개소프트웨어 중개자 등장은 공개소프트웨어 사용성과 편의성을 높이게 되어 개발자와 사용자간 연결고리가 견고해졌다. 예로, 리눅스배포판에 의해 리눅스 접할 기회가 많아진 소프트웨어 개발자들이 공개소프트웨어 커뮤니티를 통해 리눅스개발에더욱 참여했다. 여러 리눅스가 경쟁하면서 사용자 사용성과 편의성이 높아지고, 이 때문에 사용자가 더욱 좋은 리눅스를 쉽게 사용할 수 있게 된 것이다.

4. 결론

4.1. 기대효과 및 의의

지금까지 공개소프트웨어 이해, 공개소프트웨어 유통과 진화, Linux 배포판 사례, 소스포지닷넷 분석 등을 통해 공개소프트웨어에서 중개자 필요성에 대해 살펴보았다. <그림 1>과 같이 공개소프트웨어 시장이 성장하려면 공개소프트웨어 중개자(intermediary)가 등장해야 한다. 이를 위해서는 소프트웨어 개발자들이 원하는 공개소프트웨어를 편하고 신속하게 이용할 수 있도록 공개소프트웨어 중개자가 소프트웨어 유통채널에 등장해야 한다.

유통채널에서 중개자 존재해야 하는 이유는 거래효율성 증대와 정보탐색 용이성 때문이다. 공개소프트웨어 도입은 수요자가 직접 관여하여 개발 하는 기존 개발방법과 다르다. 도입기업(또는 사용자)이 필요한 기능을 온라인상에 존재하는 공개소프트웨어를 탐색하여 사용해야 한다. 도입 기업에게는 공개소프트웨어를 사용하기

위한 검토방법과 공개소프트웨어의 완성도를 확신할 수 없어 사용할 때 위험과 저항이 발생할 수 있다. 사용자가 사유소프트웨어를 구입하거나 자체개발보다 공개소프트웨어로 전환하고자 할 때 인지되는 비효율(경제적, 심리적 비용)이 크다면 공개소프트웨어 도입장벽이 높아 공개소프트웨어를 사용하는데 주저할 것이고, 도입비용이 낮아진다면 보다 쉽게 전환할 수 있을 것이다. 전환비용은 한 개의 대안으로부터 다른 것으로 전환하는 것과 관련된 경제적, 심리적 비용이다(Jones et al., 2002).

초기 공개소프트웨어 커뮤니티에 올라온 공개소프트웨어들은 개발자 중심이었고 자생적으로 각 분야별 1~2개 공개소프트웨어로 수도적고 기능도 충분하지 못했다. 하지만 공개소프트웨어는 양(量)과 질(質)에서 성장하고 분야별로 그룹핑되면서 기능 보다 우수하고 안정성이 높은 공개소프트웨어가 사용자 선택을 많이 받게 되고, 보다 많은 관심을 받은 공개소프트웨어는 오류를 수정할 수 있는 기회를 더 많이 갖게 되어, 기능이 추가되고 보완되면서 소프트웨어 완성도가 높아진다.예를 들면, 같은 분야 공개 소프트웨어 A와 B가 있다. 공개 소프트웨어 A는 전반적인 기능이 B 보다 우수하고 잘 만들어진 소스로 만들어져 있다. 그런데 B는 특수한 기능 한 가지가 A보다 우수하다. 그러면 공개 소프트웨어 A는 B의 우수한 부분 기능을 A에 이식하거나 새로 개발해 넣을 개발자를 확보할 기회가 B 보다 많아진다. 공개 소프트웨어가 소프트웨어로서 외형적인 기능뿐만 아니라 얼마나 잘 짜인 소스코드냐가 평가와 선택에 주요 요소로 작용한다.

공개소프트웨어에 대한 신뢰성, 안정성, 사용성 그리고 유지보수성은 사용자가 개인수준에서 판단할 수밖에 없다. 또한 원하는 기능의 공개소프트웨어를 찾는 탐색비용(searching cost)이 발생한다. 사유소프트웨어는 소프트웨어 공급자가 지속적인 기술지원을 하지만 공개소프트웨어는 대부분 커뮤니티가 생성을 하고 있기 때문에 품질보증과 지속적 기술지원에 대한 보장이 어렵다. 대표적 공개소프트웨어 MySQL은 소스코드 1만 라인당 약 2개 버그가 있는데 일반적인 사유 소프트웨어는 1만 라인당 1~7개 버그가 있는 것과 비교하면 대단히 안정성과 신뢰성 측면에서 우수한 품질을 제공하고 있다고 본다(Cnet News, 2005). 그러나 아쉽게도 대부분 공개소프트웨어는 기능은 일부이거나 특정 세부 요구기능 만을 만족시키고 공개소프트웨어 개발자와 사용자간 연결고리는 느슨하다. 이러한 특성 때문에 공개소프트웨어 사용자 요구를 신속하게 반영되어 공개소프트웨어 커뮤니티를 통해 제공되고 있지 못하고 있다. 즉, 커뮤니티 배포방식은 사용자의 공개소프트웨어(예, LAMP: Linux, Apache, MySQL, PHP) 채택을 높였지만 사용자 목적성에 맞게 발전한 것이 아니라 개발자 중심에서 발전했다.

그리고 공개소프트웨어의 장점이자 단점은 다양한 소스코드에 기반한 유사 소프트웨어가 넘치고 있다는 것이다. 이미 국내도 여러 전문 벤더에서 여러 버전의 Linux를 배포하여 있어 일반 사용자들은 선택의 즐거움 보다는 혼란스러움을 겪고 있다. 공개소프트웨어의 특성상 다양한 개발자의 참여가 필수적이지만, 공개소프트웨어 커뮤니티가 소스코드 공유뿐만 아니라 개발자 상호간의 지원과 협력 등이 수반되어야 한다(Henkel, 2006; 디지털타임스, 2006). 그렇지 않으면 비슷한 수준의 유사 소프트웨어만 양산할 수 있다. 그리고 공개소프트웨어가 비즈니스 환경에 유연한 구조의 소프트웨어를 만들기보다는 개발자 관점에서 기능적 우월함과 신기능 중심으로 개발되는 것도 하나의 장애요인이다. 빠르게 변화하고 있는 비즈니스 환경과 자사 비즈니스 요구에 적합한 소프트웨어를 사용하기 위해 공개소프트웨어를 사용자 중심으로 개발되고 활용 할 수 있다면 ICT 투자비용 문제를 해결할 수 있을 것이다.

공개소프트웨어 커뮤니티에 올라온 공개소프트웨어를 소프트웨어 개발자들과 사용자가 자신이 원하는 공개소프트웨어를 검색 및 활용이 용이하고 개발하고자 하는 소프트웨어를 수정하는데 어려움이 없어야 한다. 또한 소프트웨어 시장에서 공개소프트웨어가 사유소프트웨어 대체재가 아니라 보완재로 인식하는 것이 중요하다. 그 이유는 공개소프트웨어가 대체재로 인식되는 경우에는 사유소프트웨어 대체재로서 시장을 축소지만, 보완재로서 사용될 경우 공개소프트웨어 시장 성장에 상호 기여하기 때문이다.

위와 같은 공개소프트웨어 유통구조가 구축되기 위해서는 수익(또는 혜택)을 공유할 수 있는 체계를 제공하는 공개소프트웨어 중개자가 등장해야 한다. 이를 기반으로 고객과 접점을 이루고 있는 소프트웨어 개발기업과 IT 서비스기업의 기술력과 결합해서 개선된 소프트웨어를 개발 할 수 있을 것이다. 공개소프트웨어 중개자는 자체 기술력을 결합시켜 공개소프트웨어 정보서비스(특성과 장점)와 고객화(활용범위 선정, 소스코드 수정) 등 차별화된 서비스를 제공해야 한다. 공개소프트웨어 중개자는 공개소프트웨어 개발자와 구매자(SW업체)들이 공개소프트웨어 정보를 교환하고 구매할 수 있도록 도와주고 구매자별 맞춤형 서비스 등을 제공해야 한다.

4.2. 한계점 및 향후 연구방향

현재 국내 소프트웨어 시장은 대기업과 글로벌 기업이 높은 시장 지배력을 가지고 있다.이런 소프트웨어 시장에 공개소프트웨어에 의해 유통참여자간 구속력과 결속력이 느슨해진다면 중소 소프트웨어 업체들에게 진입장벽이 낮아져 소프트웨어 시장에 진입할 수 있을 것이다. 그리고 IT에 대규모 투자과IT 역량이 부족한 중소기업이 공개소프트웨어를 자사 환경에 맞게 시스템을 구축하고 전략적으로 활용 할 수만 있다면 경쟁우위를 확보할 수 있을 것이다. 예를 들어, 리눅스배포판에 포함되어 있는 아파치 웹서버는 많은 기능이 포함 되어있다. 하지만 중소기업은 많은 기능이 필요하지 않는 경우가 있다. 이때 기능이 가볍고 사용이 간편한 아파치 버전이 있다면 사용자 선택의 폭은 더 넓어지고 유지보수와 성능 측면에서도 많은 혜택이 있을 것이다. 공개소프트웨어는 소스코드가 공개되어 있기 때문에 특정 소프트웨어에 종속됨이 없이 개발기간을 단축시키고 자사에 맞는 정보시스템을 구축할 수 있다는데 있다. 이러한 이점이 다른 측면에서는 사용자가 소스코드 분석과 수정을 해야 하고 다시 그 소스코드를 공개해야 하기 때문에 IT 역량이 낮은 기업이 이용하는데 어려움이 있다.

지금까지 공개소프트웨어 중개자가 왜 필요하고 어떤 역할을 해야 할지에 대해 언급했다. 마지막으로 공개소프트웨어 중개자는 첫째, 여러 공개소프트웨어의 복잡한 내부 구조를 잘 이해하고, 둘째, 설치와 사용에 대한 축적된 경험을 가지고 있으면서, 셋째, 사용자가 요구하는 기능적, 비기능적(사용자경험, 성능) 요구사항을 개발할 수 있어야 한다. 넷째, 이를 위해 여러 공개소프트웨어를 시장요구사항에 맞는 소프트웨어 세트화 하여 새롭게 솔루션화 하는 능력과 다섯째, 개별 소프트웨어 세트의 기능과 비기능으로 해체 결합 할 수 있는 능력을 필요하다. 그리고 여섯째, 중개자는 시장에 제품을 제공하는 방법을 제품중심에서 서비스 중심으로 바꿔야 한다. 또한 공개소프트웨어 중개자는 공개소프트웨어 소스코드에 대한 분석·수정 능력과 사용자 요구사항을 공개소프트웨어와 사유소프트웨어 결합을 통해 사용자에게 제공할 수 있어야 한다.

Received: June 05, 2012.
Revised: January 14, 2013.
Accepted: February 15, 2013.

References

Amor, J.J., Robles, G., Gonzalez-Barahona, J.M. and Pena, J.F. (2007), "Measuring Etch: the size of Debian 4.0", from https://penta.debconf.org/~joerg/attachments/33-measuring_etch_slides.pdf.

CIOBIZ (2010), A Research Paper on Open Source Inex, from http://www.ciobiz.co.kr/news/articleView.html?idxno=3621.

Chun, M. K. (2002), Open source Software and Software Industry, *Communications of the Korea Information Science Society*, 20(12), 14-21.

Cnet News (2005), Week in review: March of the penguins, from http://news.cnet.com/Week-in-review-March-of-the-penguins/2100-1083_3-5829638.html.

Digital Times, (2006), 25% Adoption of Open Source Database, Office sofrware,from http://www.dt.co.kr/contents.htm?article_no=2006062002010151600001.

GNU/Linux Distro Timeline(2011), Distribution Time Line Data, Septermber 2011, Retrieved from www.futuris.se/gldt.

Henkel, J. (2006), "Selective Revealing in Open Innovation Processes: the Case of Embedded Linux", *Research Policy*, 35(7), 953-969.

Geeknet, Inc.(2012), SourceForge, from http://www.sourceforge.net.

Jones, M. A., Mothersbaugh, D. L. and Beatty, S. E. (2002) "Why customers stay: measuring the underlying dimensions of services switching costs and managing their differential strategic outcomes", *Journal of Business Research*, 55, 441-450.

Kim, H. W., Roh, S. E., Lee, H. R. and Kwak, K. Y., (2009), The Effect of Switching Costs on user Resistance in the Adoption of Open Source Software, Information System Reviews, 11(3), 125-146.

Ko, D. S. and Koo, B. K., (2012) Study on the Barrier of Introduction and Use of Open Source Software, *Sustainable ICT Conference*, 2012, 677-683.

Korea Chamber of Commerce and Industry(2009), The Task agenda for enhancing IT industry's performance, 2009 Annual Research Reports, Seoul, South Korea, 1-33.

Kwon, M. J., Kim, T. W. and Kim, M. H. (2008), A Exploratory Study on Korean Open source Software and Barriers to vitalization of Open source software, *Information Policy Research*, 15(4), 3-21.

Lin, L. (2008). "Impact of User Skills and Network Effects on the Competition between Open Source and Proprietary Software", *Electronic Commerce Research and Applications*, 7, 68-81.

Ministry of Information and Communication (2006). A Guide to Open Source Software, 2006 January.

Nam, I. K., Lee. J. D. and Kim, T. Y., (2006), A Case study of Enlarging Market for Activating Open Source Software, *Communications of the Korea Information Science Society*, 24(6), 18-23.

Open Source Initiative (2012), *Definition of Opensouce*, Retrieved September 30, 2012, from http://opensource.org/docs/definition.php.

Patrenko, M., Poshyvattyk, D., Rajlich, V., and Buchta, J. (2007), Teaching Software Evolutionin Open Source, *IEEE Computer*, November,25-29.

Wikipedia. (2012), Open Source Initiatives, November 2012, Retrieved from http://ko.wikipedia.org/wiki/%EC%98%A4%ED%94%88_%EC%86%8C%EC%8A%A4_%EC%9D%B4%EB%8B%88%EC%85%94%ED%8B%B0%EB%B8%8C.