공개소프트웨어 기반 환경에서의 소프트웨어 개발 교육

국민대학교 황선태 · 정낙수 · 허대영

1. 서 론

소프트웨어에 대한 교육을 하려면 소프트웨어가 무엇인가에 대해서부터 명확한 인식이 필요하다고 할 수 있다. 이런 경우 직접적인 정의가 어렵다면 존재의 가치를 따져보는 것도 의미 있는 일이라고 할 수 있겠다. 혹 예술 작품이라면 만들어서 전시해 놓고 감상하는 것이 존재의 이유이겠지만 소프트웨어는 그렇지 않은 것이다. 아마 사람의 의사를 컴퓨터에게 전달하는 매체의 역할을 하는 것이 소프트웨어일 수 도 있겠다. 즉 사용자가 어떤 목적을 가지고 사용해야하는 일종의 도구인 것이다. 이런 존재의 가치를 충족시키려면 소프트웨어는 사용자에 의해서 사용되어야 하는 것이다. **사용되지 않는 소프트웨어는 이미 그 존재의 가치를 갖지 못한 것이 되는 것이다.** 개발자가 이런 소프트웨어 존재의 본질을 깊이 이해하고 소프트웨어를 만든다면 훨씬 가치 있는 소프트웨어를 만들 수 있을 것이다.

그렇다면 어떻게 하면 소프트웨어가 사용되게 할 수 있을 것인가? 사실 막대한 예산을 들여서 연구, 개발되어진 소프트웨어도 사용되지 않고 사장돼버린 예는 쉽게 찾을 수 있다. 이런 관점에서 볼 때 소프트웨어 개발의 개념을 약간 바꿀 필요가 있는 것 같다. 즉 **소프트웨어를 개발해서 발표한다는 선언적 의미 보다는 우호적 토양에 소프트웨어를 심고 가꾸어서 자라게 한다는 진행형의 의미를 부여**하는 것이 훨씬 성공 확률이 높은 것이다. 공개소프트웨어의 활용은 이런 문제에 대해서 어느 정도 답을 주는 것 같다. 이미 어느 정도 사용자층을 확보하고 있으므로 공개소프트웨어를 기반으로 해서 성능 개선이나 기능 추가 등을 위한 개발을 시도한다면 안정되게 사용할 수 있을 정도의 소프트웨어로 완성하는 일이나 사용자를 확보하는 일 등에서 유리할 수 있다.

여기서 관심있는 것은 실제로 사용되어지는 소프트웨어 개발에 대한 체험적 교육을 어떻게 하느냐이다. 일반적으로 교육에 적절한 환경이 주어질 때 그 교육의 절반 이상이 해결된다고 볼 수 있다. 공개소프트웨어를 교육하려면 그 교육 환경을 이에 가장 적절하도록 구축하는 것이 우선인 것이다. 즉 교육은 처한 환경에 대해서 반응하는 것에서부터 출발한다고 볼 수 있는데 공개소프트웨어로 구성된 교육 환경을 호기심을 가지고 들여다보는 것만으로도 이미 많은 것을 배울 수 있는 것이다. 게다가 대학의 경우 그 구성원 자체가 개발자 이면서 사용자가 될 수 있다. 즉 자체적으로 개발된 소프트웨어를 강제적으로라도 사용할 수 있는 것이다. 이렇게 소프트웨어가 사용되어지면 그 소프트웨어는 생명력을 가졌다고 볼 수 있는데 여러 가지 사용자 요구 사항이 도출되어 후속 개발로 이어질 수 있는 것이다. 또한 공개소프트웨어로 구성된 교육 환경은 경우에 따라서는 매우 복잡한 시스템 통합 이슈와 세심한 운영 관리의 문제가 야기 되는데 이 또한 대학 구성원의 좋은 경험거리가 될 수 있는 것이다.

본 논문에서는 대학에서 소프트웨어 교육을 위해서 공개소프트웨어를 기반으로 어떤 실습 환경을 구축할 수 있는지를 설명하고 이런 환경에서 도출될 수 있었던 시스템 통합, 시스템 운영 및 소프트웨어 개발 교육 관련 이슈들을 언급하고 몇 가지 개발 사례를 설명한다.

2. 공개소프트웨어 기반으로 구축된 소프트웨어 교육 환경

일반적으로 대학교의 소프트웨어 교육을 위해서는 높은 수준의 실습 환경 구축을 필요로 한다. 또한 실습 환경을 다양하게 구성하여 다양한 실습을 할 수 있도록 고려되어야 하고, 관리 및 운영이 용이하도록 설계되어 최상의 환경을 유지하고 장애발생시 신속하게 복구될 수 있어야 한다. 학교의 특징은 이런 실습 환경을 중심으로 필요로 하는 여러 구성원을 자체적으로 확보할 수 있다는 것인데 그 역할에 따라서 다음과 같이 구분할 수 있다.

- **사용자** : 실습 환경에서 프로그래밍과 같은 실습을

한다.

- **개발자** : 실습 환경 개선에 필요한 소프트웨어 개발
- **관리자** : 실습 환경에 대한 설계 및 구성과 운영

이런 구분이 서로 배타적인 것은 아니며 거의 모든 구성원이 사용자이면서 그 중 일부는 학년이나 개발 능력에 따라서 관리자 또는 개발자의 역할도 하게 되는 것이다.

높은 수준의 다양한 소프트웨어 교육을 위해서는 실습 환경이 다음과 같은 조건을 만족해야 한다.

사용자 및 개발자를 위한 실습 환경 구성 조건

- 멀티 플랫폼을 지원하여 다양한 환경에서의 실습과 개발, 테스트를 진행할 수 있어야 한다.
- 다양한 플랫폼을 이용하는 만큼 단일화된 인증시스템을 가져야 한다.
- 실습환경은 일관성있게 제공되어야 한다. 즉, 실습환경에 사용되는 컴퓨터의 구성 및 설치된 프로그램은 모두 동일하여야 한다.
- 사용자는 실습실의 모든 PC에 대해 사용이 가능하여야 한다.
- 실습환경에 대하여 사용자별로 권한이 할당되어 원활한 실습이 될 수 있도록 하여야 한다.
- 실습환경 하에서 다양한 개발이 가능하도록 여러 개발 소프트웨어가 설치되어야 하고, 테스트가 가능하여야 한다.

관리자를 위한 실습 환경 구성 조건

- 하드웨어와 소프트웨어 모두 관리자의 관리사항이 최소화되도록 구성되어야 한다.
- 실습실의 모든 PC에 대해 동일한 설정이 적용될 수 있는 시스템이 구성되어야 한다.
- 서버들은 유기적으로 동작하여 계정 관리에 대한 비용이 최소화되어야 한다.
- 사용자가 실습 환경을 변경할 수 없도록 하여야 한다.

이와 같은 요구사항을 충족시키기 위해서는 시스템과 네트워크 구성이 일반 엔터프라이즈 환경보다도 더 복잡한 구조가 될 수 있고 다양한 플랫폼을 포함한 시스템 연동은 완성도가 높지 않으면 장애를 일으키기 쉽다. 또한 공개소프트웨어를 적용하여 시스템을 구성할 경우에는 상용 소프트웨어를 이용하여 구성하는 것보다 더 높은 수준의 시스템에 대한 이해를 필요로 한다. 따라서 관리자 역할을 하는 학생뿐만 아니라 사용자의 경우에도 구축된 시스템을 사용하고 관찰하는 것만으로도 시스템 설계 및 통합에 대한 기본적 이해를 할 수 있게 된다. 또한 공개소프트웨어를 사용하기 때문에 자유롭게 코드를 수정하고 개선할 수 있는 이점을 얻을 수 있으며 이미 완성된 공개소프트웨어를 분석함으로써 소프트웨어 개발 능력을 향상시킬 수도 있다.

앞에서 언급한 실습 환경의 구성은 사용자를 위한 클라이언트 환경 부분과 클라이언트에 서비스를 제공하기 위한 서버 부분으로 구분되어진다. 전체적인 실습 환경 구성은 그림 1과 같다.

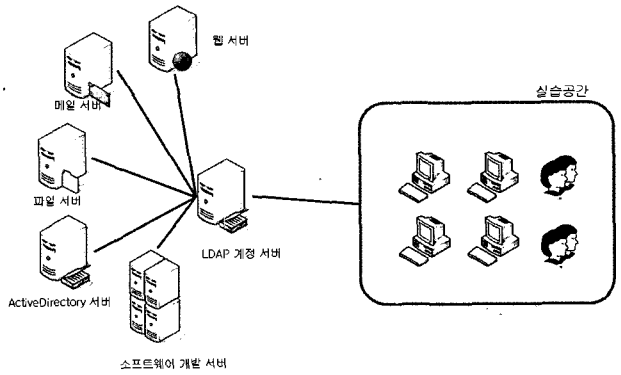


그림 1 실습 환경 구성

2.1 클라이언트 환경 구성

클라이언트 환경은 사용자나 개발자가 임의의 실습환경으로 사용 및 개발을 할 수 있도록 되어야 한다. 기본적인 클라이언트의 환경은 표 1과 같이 구성되어 있는데 기본적으로 공개소프트웨어인 KMU-Linux로 구성되어 있으며 다양한 실습과 개발을 위해 Windows로 멀티부팅할 수 있다.

표 1 클라이언트 환경 구성

	일반 클라이언트
OS	KMU-Linux[1] WindowsXP[2]
권한	KMU-Linux : 사용자 권한 Windows : 사용자 권한
인증 서버	KMU-Linux : LDAP[3] Windows : ActiveDirectory[4]
용도	수업 및 실습

사용자 관점에서의 클라이언트 환경 구성

사용자는 클라이언트 환경에서 수업 및 실습 을 진행하게 된다. 수업 및 실습은 기본적으로 Linux 클라이언트에서 진행되고 특정 수업의 경우 Windows로 멀티 부팅되게 하여 Windows 환경 하에서도 실습 및 수업이 원활히 진행되도록 구성되었다. 사용자가 클라이언트 환경에서 웹을 사용할 경우에는 특정 포트나 사이트의 접근 금지 등의 제약을 받게 된다.

사요자 및 개발자는 기본적으로 Linux와 Windows 클라이언트를 이용하여 개발을 진행하게 되나 사용권

한에 의해 제약을 받을 수 있다. Linux의 경우 root 권한을 얻지 못하므로 커널 프로그래밍이나 KMU-Linux와 같은 배포판 개발 시에 제약을 받게 된다. Windows의 경우에도 마찬가지의 경우가 발생하게 되는데 이와 같은 경우를 극복하기 위해 일반 클라이언트 실습 공간이 아닌 별도의 개발 공간을 제공한다.

관리자 관점에서의 클라이언트 환경 구성

클라이언트 환경은 관리비용이 최소화 되도록 구성되어야 한다. 따라서 기본적으로 사용자의 권한을 축소함으로써 클라이언트 환경이 변경될 수 있는 여지를 최소화하고 통합 인증, 네트워크 파일 서비스 등을 통해 각 데스크탑에 불필요한 데이터를 저장하는 것을 최소화하였다.

2.2 통합 인증 서비스

사용자는 실습환경에서 제공하는 웹 서비스, Linux 클라이언트, Windows 클라이언트를 비롯하여 ssh, 네트워크 파일 서비스 등과 같은 인증이 필요한 다양한 서비스에 접근하게 된다. 이 때 당연히 사용자의 인증 절차를 거쳐서 접근 권한이 확인된 사용자에 대해서만 서비스를 제공하게 된다. 그러나 서비스간 인증이 통합되지 않을 경우 각각의 서비스 별로 독자적 인증 절차를 거쳐야만 해당 정보에 접근이 가능하다. 통합 인증 서비스는 이러한 서비스의 인증 절차를 통합하여 서비스에 대한 접근을 용이하게 한다.

사용자 관점에서의 통합 인증 서비스

사용자는 실습환경의 서비스들에 단일 ID/PAS소프트웨어D를 통해 접근한다. 단일 ID/PAS소프트웨어D를 사용할 경우 클라이언트에 접속하는 것만으로 SSO(Single Sign On)[5]가 되어 다양한 다른 서비스에 추가 인증절차 없이 편리하게 접근할 수 있게 되고 하나의 서비스에서 패스워드를 변경하면 다른 서비스에도 변경된 패스워드를 이용할 수 있다. 또한 클라이언트 환경의 모든 데스크탑에 부팅된 플랫폼에 상관없이 로그인이 가능하다.

관리자 관점에서의 통합 인증 서비스

계정 관리 서버만 관리함으로써 관리 포인트를 단일화 시킬 수 있으며 관리 비용을 최소화할 수 있다.

통합 인증 서비스의 구성

Linux에서의 대표적인 인증 서비스로는 전통적인 PAM을 들 수 있으며 PAM에서 확장된 NIS, LDAP, Kerberos[6]를 통한 인증 역시 가능하다. 기본적으로 PAM, NIS 등은 Linux 클라이언트를 위해 제공되는 인증 서비스이며 LDAP의 경우 다양한 시스템과 연동이 가능하다. Windows의 경우 SAM과 ActiveDirectory가 있는데 최근의 Windows에서는 ActiveDirectory가 기본적인 인증 서비스이다.[7]

LDAP을 통한 인증 서비스

기본적인 Linux 계정 정보는 LDAP으로 재구성되어 제공되어져서[8] 클라이언트의 인증처리도 LDAP을 통해 할 수 있도록 구성되어 있다. 또한 웹 서버와의 연동 역시 LDAP을 통한 인증방식을 사용하고 있으며 웹 서비스를 위한 커넥션 모듈을 개발함으로써 홈 페이지 등 웹 서비스를 외주로 개발할 때에도 제공되는 모듈을 사용하여 SSO를 구성할 수 있다.

ActiveDirectory를 통한 인증 서비스

Windows 클라이언트의 경우 LDAP을 통한 인증이 불완전하여 클라이언트에 대한 다양한 제어를 위해 ActiveDirectory를 사용하였다. 일반적으로 Active-Directory와 Linux와의 통합인증에는 Kerberos를 이용한 External Trust 방식의 통합인증 방식[9]과 SFU(Service For Unix)[10]를 이용한 계정 동기화 방식이 있다. Kerberos를 이용한 External Trust 방식은 완벽한 SSO를 구성할 수 있다는 장점이 있으나 계정 정보가 Linux에 있는 이상 사용자에 대한 다양한 제어가 불가능하다는 단점이 있다. 반면 SFU를 이용할 경우 사용자와 컴퓨터에 대한 다양한 제어가 가능하며 Windows 클라이언트 환경에 대해서 정책을 적용할 수 있다는 장점이 있다. 이 경우에는 Windows 클라이언트 환경에 대하여 적절한 정책을 수립하고 배포하기 위해 SFU를 통한 통합 인증 서비스를 채택하였으며 이를 통해 Linux와의 계정 동기화와 Windows 클라이언트의 완벽한 제어가 가능하게 되었다.[11]

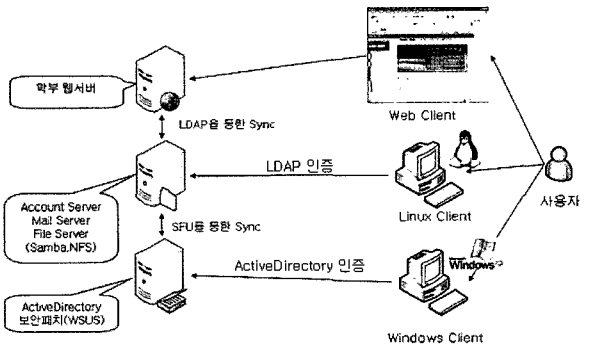


그림 2 통합 인증 시스템 구조

2.3 네트워크 파일 서비스

네트워크 파일 서비스는 서버의 파일 시스템의 일부를 공유하여 해당 데이터를 한 명 이상의 사용자가 사용할 수 있도록 하는 서비스이다. 네트워크 파일 서비

스가 필요한 이유는 사용자나 개발자가 클라이언트에서 작성한 데이터를 별도로 보관하지 않고 파일 서버에 저장함으로써 사용자나 개발자에게 편의성을 제공하고 데이터의 안정성을 보장하여 데이터에 대한 신뢰성이 높은 실습 환경을 제공한다.

사용자 관점에서의 네트워크 파일 서비스

사용자는 지정된 데스크탑이 아닌 임의의 데스크탑을 사용하게 된다. 따라서 어느 위치에서나 자신의 데이터에 접근할 수 있도록 파일 서비스가 제공되어야 한다. 또한 데스크탑에 Linux와 Windows 등 다양한 플랫폼이 설치되어 있는 만큼 각각의 플랫폼에서 접근 가능한 파일 서비스가 제공되어야 한다. 그리고 각 사용자에게 할당된 공간은 해당 사용자 이외의 사용자로부터 보호되어야 한다.

관리자 관점에서의 네트워크 파일 서비스

사용자의 데이터가 파일 서버에 집중되어 있으므로 관리가 용이하다. 그러나 백업에 대한 대책이 있어야 하고 보다 강력한 보안 정책을 적용할 필요가 있다. 각 사용자별 사용량의 상한선을 관리할 수 있어야하고 사용자 수의 변동에 따른 파일 시스템의 확장 축소가 유연해야 관리가 수월하다.

네트워크 파일 서비스의 구성

네트워크 파일 서비스를 제공하는 서버는 Direct Access Storage로 구성된 Linux 서버이다. 따라서 Linux에서 제공하는 파일 서비스를 통해 클라이언트에 네트워크 파일 서비스를 제공하게 되며 대표적인 공개소프트웨어 네트워크 파일 서비스는 NFS[12]와 Samba[13]가 있다. Linux 기반이므로 LVM을 이용할 수 있다.

LVM을 이용한 파일 시스템의 확장 및 축소

파일 시스템의 파티션 단위는 입학년도로 구성되어진다. 동일한 입학년도인 사용자들은 동일한 파티션을 사용하게 되며 해당 파티션에 개인 사용자 공간이 생성되게 된다. 각 파티션은 LVM으로 구성되며 기본적인 할당량에서 사용자의 증감에 따라 파티션의 크기를 유동적으로 확장하거나 축소하게 된다. 또한 물리적 Storage 추가 시에도 LVM을 통해 확장함으로써 유연성을 유지할 수 있다.[14]

LDAP+NFS를 통한 Linux 클라이언트 네트워크 파일 서비스의 구성

Linux / Unix의 대표적인 네트워크 파일 시스템으로 LDAP을 통해 사용자의 권한 정보를 제공하고 NFS를 통해 공간을 할당한다.[15] NFS는 기본적으로 사용자에 대해 개인 공간 할당을 하게 되며 시스템 구성을 위한 공간과 개발자가 공동 개발을 위한 공간을 제공한다. LVM을 통해 생성된 파티션을 클라이언트 머신에 할당하게 되며 사용자는 LDAP 인증을 통해 자신의 할당 공간에 접근 허용을 허가받아 사용하게 된다. Linux 클라이언트에서는 사용자가 로그인하면 자동으로 자신의 공간을 사용할 수 있도록 AutoMount를 이용하여 구성하였다.

Samba를 통한 Windows 클라이언트 네트워크 파일 서비스의 구성

공개소프트웨어로 개발된 네트워크 파일 시스템 중 Windows를 지원하기 위한 파일 서비스가 Samba이며 Samba를 통해 Linux서버-Windows 클라이언트 간 네트워크 파일 서비스를 구성할 수 있게 된다. 클라이언트에는 사용자가 로그인시 자동으로 자신의 공간을 사용할 수 있도록 ActiveDirectory를 통한 Logon Script를 사용하여 구성하였다.

2.4 클라이언트 일관성 유지

사용자 및 개발자가 실습환경을 사용하는 데 있어 중요한 부분은 어떠한 클라이언트에서도 동일한 작업을 수행할 수 있어야 한다는 점이다. 특정 클라이언트에만 특수한 소프트웨어가 설치되어 있거나 특수한 환경이 구성되어 있고 이를 이용해야만 한다면 사용자는 특정 클라이언트에 종속적일 수밖에 없다.

사용자 관점에서의 클라이언트 일관성 유지

사용자는 모든 클라이언트에 대해 사용이 가능하여야 하며 클라이언트가 제공하는 서비스는 동일하여야 한다. 이를 통해 자유롭게 클라이언트를 변경하여 사용 및 실습, 수업을 진행할 수 있다.

관리자 관점에서의 클라이언트 일관성 유지

관리자는 사용자나 개발자가 수업 및 실습을 위해 필요로 하는 사항을 최대한 빠른 시간 내에 반영할 수 있어야 한다. 이때 최소의 관리 인력을 투입하여 최단 시간 내에 보다 많은 클라이언트에 대해 변경이 가능하여야 한다.

일관성 유지 시스템의 구성

일관성 유지는 크게 시스템 초기화 부분과 소프트웨어 설치 부분으로 나뉘게 된다. 시스템 초기화는 실습 환경에서 요구하는 기본적인 사항을 충족시키는 환경 구성을 구축하는 것을 뜻하며 소프트웨어 설치는 추가적으로 발생하는 변경사항을 적용하는 것을 뜻한다.

Image Tool을 이용한 시스템 초기화

시스템에 심각한 오류 발생시, 혹은 장치의 문제로 인하여 불가피하게 시스템을 교체나 변경해야 하는 경

우 기존 시스템과 동일하게 구성하기 위해서는 정상적인 설치방법을 이용할 경우 많은 관리비용이 발생하게 된다. 따라서 관리비용을 최소화하기 위해 주기적으로 정상적인 시스템을 Image Backup하여 이를 이용하여 Image Restore 함으로써 설치에 필요한 관리비용을 최소화 할 수 있다. 그러나 Image Backup과 현재 시스템과는 Backup 시점에 따라 상이할 수 있기 때문에 이에 대한 보완으로서 시스템 동기화를 이용하여 Image Restore 이후에 발생한 변경사항을 적용할 수 있다.

rsh를 이용한 시스템 동기화

시스템이 초기화 된 경우, 혹은 추가적인 소프트웨어 설치가 필요한 경우 시스템 동기화를 통해 소프트웨어 설치 및 업데이트를 진행할 수 있다. Linux 클라이언트의 경우 rsh과 같은 서비스를 제공하기 때문에 보다 효과적으로 시스템을 동기화 할 수 있다. Linux 클라이언트는 부팅 과정에서 rsh를 이용하여 서버에서 변경된 사항을 체크하고 변경사항이 있을 경우 해당 내용을 반영함으로써 시스템 동기화가 이루어지게 된다. 또한 정상적으로 설치 및 삭제되었는지에 대한 부분을 체크하기 위해 설치에 관한 로그를 남기는 소프트웨어를 개발하고 이를 통해 로그정보를 관리자에게 제공함으로써 보다 완벽한 시스템 동기화를 구성하였다.

ActiveDirectory를 이용한 시스템 동기화

Windows 클라이언트의 경우 소프트웨어 설치뿐만 아니라 보안 업데이트 역시 중요하다. 따라서 시스템 동기화는 소프트웨어 설치와 보안 업데이트로 구분되어지며 소프트웨어 설치의 경우 ActiveDirectory에서 제공되는 GPO(Group Policy Object)중 소프트웨어 설치 부분을 통해 소프트웨어 배포 및 관리가 가능하다. 그러나 GPO를 통해 배포 가능한 소프트웨어는 MSI(Microsoft Installer)에 대해서만 배포 및 삭제가 가능하기 때문에 이를 보완하기 위해 Logon Script를 이용하여 설치 및 삭제하도록 구성하였다. 또한 보안 업데이트의 경우 WSUS[16]를 통해 자동으로 업데이트하도록 하여 보안 업데이트에 대한 동기화도 가능하도록 구성하였다.

2.5 네트워크 구성

대학교 실습실의 컴퓨터는 특정 사용자가 전용으로 사용하는 것이 아니며 불특정 다수가 사용할 수 있기 때문에 보안에 매우 취약할 수 있다. 그런데 대학의 경우 실습실 자체에 있는 정보를 보호하는 측면 보다는 웜이나 바이러스에 의한 네트워크의 성능 저하나 다른 곳을 해킹하기 위한 경유지로 이용되는 것을 막는 것이 더 중요하다. 따라서 네트워크 구성을 적절히 하여 실습 환경을 웜, 바이러스, 해킹으로부터 보호하도록 하여야 한다.

사용자 관점에서의 네트워크 구성

외부의 네트워크 장애에도 불구하고 실습 환경 안에서는 정상적인 수업 및 실습이 이루어질 수 있어야 한다. 특히 클라이언트에는 사용자의 정보나 데이터가 없고 모든 것을 서버에 의존하도록 되어있기 때문에 중요한 이슈가 된다. 또한 실습 환경 외부에서도 사용자 자신의 데이터에는 접근이 가능해야 한다. 그리고 네트워크 관련 프로그램을 개발하고 테스트 할 수 있는 기본 환경이 마련되어 있어야 한다.

관리자 관점에서의 네트워크 구성

관리자는 사용자 및 개발자가 실습 및 개발에 필요한 네트워크 자원을 제공하고 지원하여야 하나 불필요한 네트워크 서비스는 통제하여 미연에 위험을 방지하여야 한다. 또한 사용자가 네트워크 환경의 설정을 변경 못하도록 제한하여야 하며 네트워크 환경을 관리자가 통제할 수 있어야 한다.

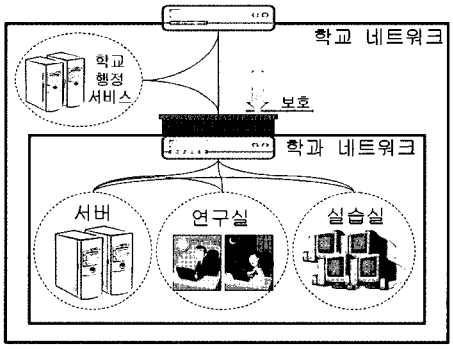


그림 3 네트워크 구조

네트워크 구성

웜, 바이러스, 해킹으로부터 보호하기 위해 네트워크를 섬과 같이 분리하였다. 또한 개발자를 위한 개발 서버를 구성함으로써 네트워크 응용 프로그램 개발이 가능하도록 하였으며 DHCP, DNS 등과 같은 서버를 통해 실습 환경에 대한 네트워크 환경을 관리자가 통제할 수 있도록 구성하였다.

IPTable을 통한 네트워크의 분리

대학에서 제공하는 네트워크에서 학부 네트워크를 분리시킴으로써 외부로부터의 위험에 대처가 가능하다. 이는 학부 네트워크를 사설 IP로 구성하고 방화벽을 설치함으로써 가능하며 방화벽은 Linux에서 기본적으로 제공되는 IPTable을 이용하여 구성하였다. 특정 서버의 경우 IPTable에서 NAT(Network Address

Translation)를 사용하여 공인 IP를 부여함으로써 외부로부터의 접근이 가능하도록 하였다. IPTable은 제대로 관리하려면 복잡한 커맨드를 자유롭게 쓸 줄 알아야 하지만 대부분의 경우 매우 쉬운 특정 설정만을 변경하면서 관리하게 된다. 이런 단순 관리를 위한 툴을 개발하면 저학년이라도 지침에 의해서 관리의 일부를 맡을 수 있다.

POPTOP[17]을 통한 외부 네트워크로부터의 접근

POPTOP은 공개소프트웨어 기반의 Linux용 PPTP (Point-to-Point-Tunneling Protocol) VPN(Virtual Private Network) 서비스이다. 외부의 사용자가 실습 환경 혹은 개발 환경의 자원에 접근하려고 할 경우 네트워크가 분리되어 있기 때문에 NAT로 개방된 서버가 아닌 경우 접근이 불가능하다. 따라서 이와 같이 NAT로 개방된 서버가 아닌 시스템에 접근하고자 하는 사용자에 대해 VPN을 통해 접근이 가능하도록 하고 있다.

DHCP/DNS를 통한 네트워크 환경 자동 구성

사용자의 클라이언트는 네트워크 설정은 IP를 지정하지 않고 DHCP로 구성하는 것이 각 클라이언트를 동일한 Image로 관리 할 때 편리하다. 따라서 네트워크 환경을 제공해주는 DHCP 서버가 존재하여야 하며 DHCP 서버 구성을 통해 네트워크 환경 정보를 구성하게 된다. DHCP는 Linux의 dhcpd를 이용하여 구성하였으며 클라이언트의 경우 실습실 공간에 할당받은 네트워크 정보를 제공받게 된다. 클라이언트 환경은 관리자에 의해 관리되는 시스템이며 이동이 발생하지 않는다. 따라서 DHCP를 통해 네트워크 환경을 제공하더라도 네트워크 환경 정보가 유동적이지 않고 고정적이어야 관리가 용이하게 된다. 따라서 DHCP 서버는 MAC(Media Access Control) 고정을 통해 특정 MAC에 대해서는 사전에 정의된 정보를 제공하게 된다. 즉, 클라이언트 A는 항상 동일한 IP Address를 비롯한 동일한 네트워크 정보를 제공받게 되며 구성되게 된다. 그리고 DNS의 경우 Linux 기반의 BIND를 이용하여 구성하였으며 DHCP와의 연동을 통해 클라이언트의 모든 Host, IP 정보를 제공하게 된다.

2.6 보안

대학에서의 사용자는 보안에 대한 개념이 확고하지 않을 수 있으며 실습 환경 또한 외부에 노출되었거나 외부 사용자의 접근이 허용되기 때문에 보안적으로 매우 취약한 구조를 가지고 있다. 또한 의도하지 않았으나 발생할 수 있는 문제에 대해서도 사전에 안전장치가 마련되어야 하며 이를 통해 실습 환경을 보다 견고하게 유지할 수 있다.

사용자 관점에서의 보안

실습환경이 해킹으로부터 보호되어야 하며 스니퍼링과 같은 가로채기에 의해 정보가 유출되지 않도록 시스템이 구성되어야 한다. 또한 응용 프로그램 개발 시 발생할 수 있는 예기치 못한 오류로부터 시스템이 보호되어야 한다. 일반적으로 발생할 수 있는 예기치 못한 오류는 프로그램 오류로 인한 무한 프로세스 생성이나 무한 파일 시스템 점유, 무한 네트워크 시스템 접속 시도 등이 있다.

관리자 관점에서의 보안

관리자는 정책적으로 실습 환경에 대해 불필요한 접근이 차단되도록 하여야 하며 허용된 서버 이외의 서버에 대해 관리자가 아닌 사용자가 접근하는 것을 차단하도록 하여야 한다. 또한 개발서버의 경우 사용자가 원격으로 접속하여 실습할 때 발생할 수 있는 오류에 대하여 방지할 수 있어야 한다.

보안의 적용

KMU-Linux의 경우 충실한 보안을 기본 설계 개념으로 하여 개발되었다. 평문으로 전송되는 서비스를 제공하지 않고 Selinux를 제공하여 최대한 견고하게 구성되었다.[18] 실습환경의 클라이언트의 경우 KMU-Linux를 사용하기 때문에 telnet이나 ftp와 같은 평문 전송 서비스가 기본 탑재되지 않았으며 설치되지 않도록 구성하였다. 또한 ulimit을 사용하여 자원(CPU, 메모리, 등)의 사용을 제한하였다. ulimit은 생성할 수 있는 프로세스의 수, 메모리 용량, 동시에 열수 있는 파일 수에 대해 제한하도록 구성하였다. 서버의 경우는 관리자가 지정한 IP 이외의 IP에서 접근하지 못하도록 IPTable을 이용하여 접근을 제한하였으며 사용할 수 있는 서비스도 SSH[19]와 같은 안전한 서비스에 대해서만 제한적으로 접속이 허용되도록 구성하였다. 또한 관리자의 IP를 도용할 가능성이 있기 때문에 관리자의 ID 이외에는 접근이 불가능하도록 SSH 구성을 변경하였다. 그리고 네트워크에서 사용하는 모든 정보에 대해 암호화방식을 사용하고 있으며 대표적으로 LDAP의 패스워드 역시 암호화되어 인증을 받도록 되어 있다.[20]

3. 공개소프트웨어 기반의 소프트웨어 개발

앞에서 설명한 실습 환경은 사용자 입장에서 보면 통합 인증 및 파일 서비스를 제공하는 리눅스/윈도우 통합 환경이며 관리자 입장에서는 구축 및 관리의 난이도가 높지만 클라이언트를 동일하게 구성하고 네트

워크 구성 및 보안 정책에 의해 관리 부담을 어느 정도까지는 줄일 수 있는 시스템이다. 이런 시스템에서 시스템 통합을 위해서 또는 사용자 및 관리자의 편의성을 향상시키기 위해서 소프트웨어를 개발할 필요성이 생길 수 있는데 이를 자체 개발하여 바로 이 실습 환경에 적용할 수 있는 것이다.

대학 구성원이 스스로 개발할 수 있는 소프트웨어는 매우 다양하고 범위가 넓은데 이를 유기적으로 묶어주고 개발 동기를 높이기 위해서 중심에 세운 것이 KMU-Linux라는 자체 개발한 Linux 배포판이다. KMU-Linux를 기반으로 했을 때 그림 4와 같은 개발 트리가 형성되는데, KMU-Linux 개발, Kernel 개발, 응용 소프트웨어 개발, 모듈 및 라이브러리 개발이 공개소프트웨어 개발 형식으로 진행된다.

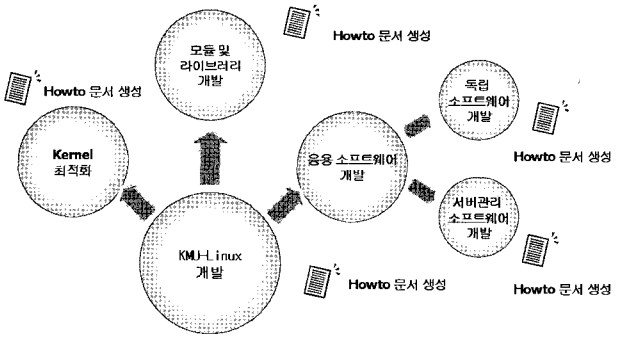


그림 4 공개소프트웨어 개발 구조

3.1 공개소프트웨어 개발 순환 구조

공개소프트웨어 기반으로 구축된 실습 환경을 관리하고 사용하다보면 새로운 소프트웨어 또는 기존 소프트웨어의 업그레이드, 여러 서비스의 기본 설정 변경 등 다양한 요구 사항이 도출 될 수 있다. 이런 요구 사항들은 구성원 안에서 자체적으로 개발팀을 구성하여 공개소프트웨어 기반의 소프트웨어 개발로 이어질 수 있는데, 이렇게 개발된 소프트웨어는 자체 실습 환경에 설치되어 다시 사용 및 관리 되어져서 자연스럽게 테스팅 과정을 거쳐서 안정화 되면서 새로운 요구 사항을 도출하는 그림 5와 같은 순환 구조를 형성하게 된다.[24] 실습 환경에서는 그림 4에서 보는 것처럼 매우 다양한 형태의 소프트웨어가 개발 될 수 있고 그 규모도 몇 줄 안 되는 작은 스크립트에서부터 팀 프로젝트로 해야 할 정도로 큰 소프트웨어까지 다양하다. 하지만 이 **다양한 개발 결과물을 모두 KMU-Linux라는 자체 배포판 제작으로 연관을 지으면 소규모이거나 라이브러리 또는 Howto 문서와 같은 형태의 결과물도 다 적절한 개발 동기를 찾을 수 있게 된다.**

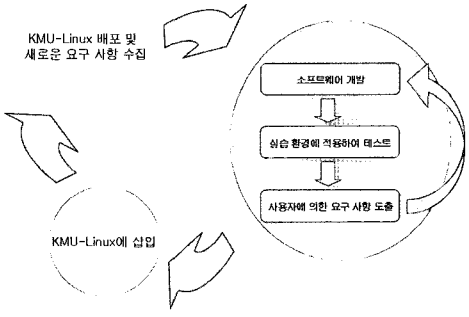


그림 5 KMU-Linux 기반의 공개소프트웨어 개발 순환 구조

3.2 KMU-Linux를 기반으로 한 여러 가지 소프트웨어 개발 사례

국민대학교 컴퓨터 학부에서는 지난 몇 년간 KMU-Linux라는 자체 배포판과 함께 여러 종류의 소프트웨어를 개발해 왔다. 아직 기간이 짧아서 그림 5와 같은 순환 구조가 두드러지게 나타나진 않지만 공개소프트웨어를 기반으로 한 실습 환경이 자체적으로 개발 및 테스트할 수 있는 여건을 잘 제공하고 있으므로 각 응용 소프트웨어의 지속적인 개발에 대한 필요성과 충분한 동기를 가지고 있다.

3.2.1 KMU-Linux 개발

KMU-Linux 개발의 경우 KMU-Linux 배포판 자체의 개발뿐만 아니라 배포판 제작 과정을 분석하고 모델링하여서 소프트웨어 도구화하는 작업인 배포판 저작도구의 개발이 함께 진행되고 있다.

KMU-Linux

국민대학교 공개소프트웨어 연구소의 발해 프로젝트의 일환으로 시작된 리눅스 배포판 개발 프로젝트의 성과물로서 2004년 KMU-Linux 1.0 발표 이후 현재 1.5까지 지속적으로 기능향상이 이루어지고 있는 리눅스 배포판이며 공개소프트웨어 개발의 베이스가 되고 있다. 또한 IA-64, LiveCD 등 다양한 버전의 KMU-Linux를 개발하여 실험적이고 진보된 Linux배포판을 발표하였다. 가장 큰 특징은 설치 CD를 **Add On 형식으로 구성한 것인데 첫 번째 CD만을 설치하면 가장 기본적인 기능으로만 동작하고 여기에 나머지 Add On CD 중 원하는 조합을 설치**하게 되어있

그림 6 KMU-Linux 1.0의 바탕 화면

다. 현재의 Add On CD는 Desktop, Server, Develop의 3종류인데 특정 주제에 맞는 Add On CD를 새로 기획해서 추가할 수 있다. 예를 들면 Game Collection이나 특정 응용 연구자를 위한 소프트웨어 모음 등이 새로운 Add On CD로 만들어질 수 있다.

커널 조정

KMU-Linux가 다양한 하드웨어 환경에서 동작하거나 제한된 환경에서 동작하기 위해서는 커널에 대한 조정이 필수적이다. 따라서 커널 옵션 및 패치 적용을 통해 KMU-Linux의 성능과 안정성을 확보할 수 있으며 궁극적으로 커널에 대한 이해도를 높이고 커널의 성능 개선을 직접 구현하고 테스트를 진행한다. 차후 커널에 대한 기본 이해가 높아진 학생이 늘어나고 좋은 여건이 조성되면 본격적인 커널 프로그래밍으로 이어갈 계획이다. 예를 들면 새로운 네트워크 프로토콜을 구현해서 해당 리눅스 박스를 실습 환경의 네트워크 장비 대신 활용하면 새로운 프로토콜을 테스트할 수 있는 좋은 기회를 얻게 되는 것이다. 이는 짜임새 있는 실습 환경이 조성되었기 때문에 가능한 일이다.

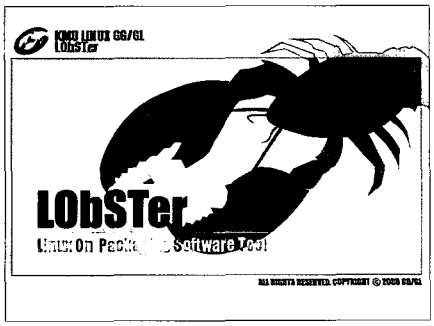


그림 7 LOBSTER:리눅스 배포판 제작을 위한 저작 도구

배포판 저작 도구

KMU-Linux 개발 경험을 바탕으로 하여 보다 손쉽게 배포판을 개발할 수 있도록 배포판 개발 프로세스를 모델링하고 이를 바탕으로 배포판 저작 도구(LOBSTER)를 개발하고 있다.

3.2.2 KMU-Linux 기반의 응용 소프트웨어 개발

KMU-Linux를 기반으로 하는 응용 소프트웨어는 시스템 관리 프로그램에서부터 일반 응용 프로그램까지 다양하다.

BaMTool(Balhae Management Toolkit)

KMU-Linux의 설정을 좀 더 편리하게 하고자 개발한 것으로 리눅스 사용자들에게 현재의 리눅스 설정에 대해 좀 더 사용자 친화적인 환경을 제공하고자 한 것이다. 또한 관리 모듈을 XML로 작성하여 Plug-in 방식으로 구현하여 확장이 용이하고 설정이 간편하도록 설계된 리눅스 관리툴이다.

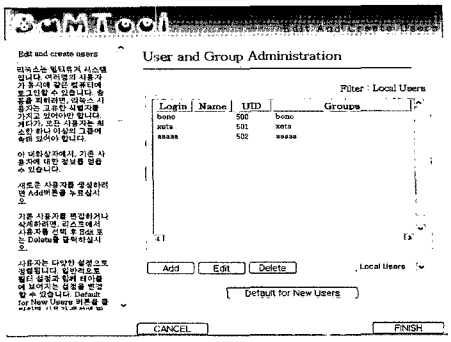


그림 8 BaMTool: KMU-Linux 관리 도구

C-Plus

KMU-Linux와 Windows와의 연동을 위한 Keberos 및 LDAP을 연동한 Identity Management Tool로서 Windows와의 상호 연계가 가능하여 Linux에서 Windows클라이언트에 대한 인증 및 정보제공이 가능하도록 구현되었다. Keberos 와 LDAP을 Linux에 설치하여 연동시키는 내용의 Howto 문서가 작성되었는데 매우 유용하게 활용되고 있다.

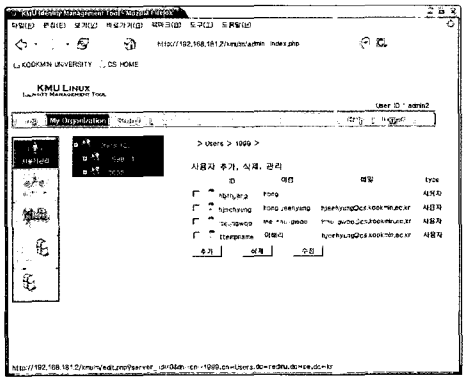


그림 9 C-Plus: LDAP+Keberos

AppleSeed

Bayesian Network 알고리즘을 이용하여 한글 환경에 최적화된 안티 스팸 메일 시스템을 구성하였다. 한글을 형태소 분석을 통해 처리하였기 때문에 한글로 된 이메일의 경우 공개소프트웨어인 스팸 어쎄신 보다 좋은 성능을 보여주고 있다. 차후에 스팸어쎄신과 같은 잘 알려진 공개소프트웨어와 접목하는 것을 계획하고 있다.

Emotion Avatar Messenger for Gaim

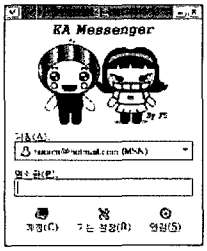


그림 10 Emotion Avata

메신저를 통해 메시지 전달시 메시지를 자연어 처리를 통해서 분석하여 사용자가 표현하고자 하는 감정을 알아내고 해당 감정에 따라 자신의 아바타 표정을 변화시켜 감정을 표현하는 메신저이다. 공개소프트웨어 기반 메신저인 Gaim과 통합하는 작업을 진행 중이다.

3.2.3 모듈 및 라이브러리 개발

실습 환경에 대한 확장이 가능하도록 실습 환경과 연관된 서비스에 대한 다양한 모듈과 라이브러리를 개발함으로써 다른 개발자의 개발을 손쉽게 할 수 있다. 대표적인 사례로서 SSO 구현을 위한 LDAP 웹 모듈을 개발하여 제공함으로써 외주에 의한 웹 서비스 개발 시 인증과 관련되어 새로운 계정 시스템을 개발하거나 SSO 관련 부분을 개발할 필요가 없게 하여 실습 환경 전체 구성과 일관성을 유지하기가 용이하게 하였다.

4. 공개소프트웨어 개발 인력 양성

서론에서 언급한 것처럼 교육에 있어서 환경만큼 중요한 것은 없다. 앞에서 기술된 공개소프트웨어 기반의 실습 환경을 기반으로 하여 각 구성원이 사용자, 관리자, 개발자의 역할을 하면서 소프트웨어 개발 교육을 받으면 그야말로 실제로 사용되면서 업그레이드되고 있는 살아있는 소프트웨어를 개발할 수 있는 기회를 갖게 된다. 이미 언급한 공개소프트웨어의 개발 사이클이 좀 더 조직적으로 순환되도록 정규 교과목, 전공 동아리 및 특강 수강자 등의 커뮤니티, 그리고 발전적으로는 산학협력 과제 등을 연계해서 공개소프트웨어 인력 양성이라는 목적을 보다 효율적으로 이룰 수 있다.

4.1 정규 교과 과정과 연계

대학의 풍부한 인력으로 구성된 잠재적인 개발 커뮤니티를 활성화하기 위해서 관련 과목의 커리큘럼을 조정하고 실습 결과를 학점에 일정 부분 반영함으로써 공개소프트웨어에 대한 지식을 습득할 수 있는 기반을 조성하였다.

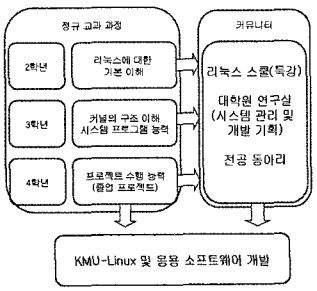


그림 11 인력 양성 구조

2학년

- UNIX 실습 : Linux 관련된 내용도 소개하며 실제 실습 위주의 과목으로 전환하여 Linux 프로그래밍의 기초 지식을 충분히 얻도록 한다.
- 기타 과목 : 반드시 윈도우 환경이 아니어도 되는 경우에는 클라이언트에서 리눅스로 부팅해서 작업하게 한다.

3학년

- 운영체제 : 실제 운영체제를 수정하거나 개발하는 실습 위주의 과목으로 전환하여 운영체제의 이론적 개념을 습득할 뿐만 아니라 실제적인 지식도 얻을 수 있도록 한다. 커널의 개념과 구조를 이해한다.
- 리눅스 프로그래밍 : 리눅스 상에서 시스템 소프트웨어를 개발할 때 필요로 하는 기술들을 실제 실습을 통해서 습득하도록 한다.

4학년

- 졸업프로젝트 : 졸업 작품 제작을 목표로 하는 팀 프로젝트 과목인데 작품 주제 중 일정량은 공개소프트웨어를 개발하도록 하도록 유도한다.

이와 같은 구조를 통해 정규 교과목을 수강하는 과정에서 자연스럽게 공개소프트웨어와 친숙해지고 공개소프트웨어 개발을 시도할 수 있는 능력을 가질 수 있다. 게다가 사용하고 있는 실습 환경의 구축과 관리가 자신과 같은 구성원에 의해서 이루지고 있고, 일부 패키지 소프트웨어나 서비스들 역시 같은 소속의 구성원에 의해서 개발되었음에 자부심을 느끼며 자신도 이를 향상하는데 기여할 수 있다는 사실이 커다란 개발 동기가 될 수 있다.

4.2 커뮤니티 연계

공개소프트웨어 개발 과정에서 커뮤니티의 역할은 매우 중요한 것으로 인식되어 왔다. 하지만 국내 공개소프트웨어 관련 산업이 매우 취약한 현 시점에서 활발한 커뮤니티 활동을 기대하기는 매우 어렵다. 즉 학교 외부로부터는 공개소프트웨어 관련 커뮤니티 활동을 하게할 정도의 강한 동기 부여가 없는 것이다. 그리고 배포판 제작 정도의 프로젝트는 학부 수준에서 충분히 소화할 수 있는데 문제는 개발 기간이다. 정규 교과 과정의 숙제에 밀려서 학기 중에 많은 일을 하는 것은 기대하기 어려운 일인 것이다. 그래서 도입한 것이 "리눅스 스쿨"이라는 특강이었다. 즉 학기 중에는 교육을 하고 대학원 레벨에서 기획하며, 방학 중에 심도있는 개발을 하는 순환 구조를 만들려고 하였다.

리눅스 스쿨

리눅스 스쿨은 리눅스에 관심있는 학생들을 대상으로 자발적인 참여를 통해 리눅스에 대한 심도있는 지

식 공유를 목적으로 하고 있다. 리눅스 스쿨의 실습환경은 리눅스를 최대한 응용하고 자유롭게 실습할 수 있도록 배려되었으며 이를 위해 관리자권한이 부여되고 다양한 디바이스를 구축하여 학생들의 관심과 부합하는 공개소프트웨어 개발을 유도하고 있다.또한 다양한 공개소프트웨어를 접목시키고 구성할 수 있도록 교육하고 지도하여 지식뿐만 아니라 경험도 쌓게 된다. 이를 통해 자연스럽게 공개소프트웨어에 대한 관심을 유도하고 공개소프트웨어에 심취할 수 있도록 하였다. 실제로 제 리눅스 스쿨 수강생들은 작은 규모의 커뮤니티처럼 움직였다.

4.3 산학협력

산학협력은 기업의 기술력과 자본, 대학의 인력과 기술력을 바탕으로 실험적인 면과 실용적인 면을 접목시킬 수 있는 좋은 기회이다. 이미 많은 산학 협력 사례가 이를 입증하고 있으며 공개소프트웨어의 경우에는 수많은 공개소프트웨어 프로젝트가 이러한 산학협력을 통해 개발되어지고 있다. 산업계와 대학이 가지고 있는 장점을 활용할 경우 시너지 효과를 충분히 발휘할 수 있으며 이를 통해 공개소프트웨어 활성화에 기여할 수 있게 된다.[22]

그림 12와 같이 대학은 테스트 인력과 초급 개발인력을 보유하고 있는 반면 산업체는 고급 개발 인력을 보유하고 있다. 또한 대학이 우수한 개발 환경과 테스트 환경이 갖춰진 반면 중소산업체는 개발 환경과 테스트 환경이 열악한 대신 고급기술과 노우하우를 가지고 있다. 이러한 산학간 장점을 극대화 시킬 경우 개발 소프트웨어의 완성도를 높일 수 있다. 실제로 KMU-Linux도 산학 협력 과제에 의해서 탄생하였다. 초기에 배포판 제작에 관한 노우하우 등은 해당 업체의 전문가로부터 특강을 통해 전수 받았고 이를 학생들이 소화해서 계속 업그레이드 하고 있는 것이다. 이런 산학 협력에 의한 기술 이전 역시 그 전부터 존재해온 공개소프트웨어 기반의 실습 환경이 있었고 학생들이 이미 어느 정도는 그 환경에 익숙했었기 때문에 가능한 일이었다.

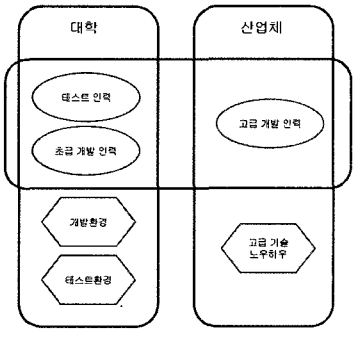


그림 12 산학 협력 구조

4.4 자작 실습 환경 운영의 효과

현재 국내의 산업 현황으로 볼 때에 공개소프트웨어에 대한 관심을 가질 만한 동기를 학교 외부로부터 찾기는 어려운 것 같다. 하지만 해외의 움직임을 볼 때, 그리고 우리나라 소프트웨어 산업의 구조로 볼 때 결코 소홀히 해서는 안되는 분야인 것이다. 표 2는 국민대학교 컴퓨터학부의 지난 7년간의 졸업 작품 주제의 기반을 분류한 것이다. 표에서 보이는 것 처럼 2004년부터 Linux 기반의 작품이 2배로 늘었고 전체 과제수에 대한 점유율도 급증하였다. 실제 KMU-Linux v1.0이 2003년에 개발되기 시작되어 2004년 7월에 공개된 시점과 무관하다고 볼 수 없다. 학생들이 자신의 힘으로 리눅스 배포판을 만들고 그 것을 실제로 실습 환경에 설치해서 사용하고 또 그 위에서 응용 프로그램을 개발하는 과정에서 공개소프트웨어에 대해서 많은 관심을 갖게 되었고 자부심과 함께 개발에 직접 참여하고 싶은 동기부여가 되었다고 볼 수 있다.

표 2 졸업 프로젝트 주제 분석

	Linux App	Windows APP	Java App	Web App	ToTal	%
1999년	0	9	1	4	14	0.00%
2000년	3	3	2	13	21	14.29%
2001년	4	13	2	1	20	20.00%
2002년	5	12	1	0	18	27.78%
2003년	4	11	2	0	17	23.53%
2004년	8	7	2	0	17	47.06%
2005년	7	10	1	1	19	36.84%

5. 결 론

지금까지 **공개소프트웨어를 기반으로 한 소프트웨어 교육 환경의 구축 및 이 환경에서 이루어지는 사용자와 관리자의 역할에 대해서 이야기하였고 이런 인프라 위에서 같은 구성원이 개발자가 되어서 좋은 모델의 공개소프트웨어 개발 사이클을 순환시킬 수 있음**을 설명하였다. 그리고 이런 개발 사이클을 잘 활용할 수 있는 인력 양성의 구조에 대해서도 이야기하였다. 이런 환경이라면 앞에서 언급한 것처럼 사용되어지는, 그래서 살아있고 개선되어질 수 있는 소프트웨어의 개발이 가능하고 이런 소프트웨어를 만드는 양질의 인력을 양성하기가 수월해지는 것이다. 교육은 적절한 환경에서부터 시작한다. **자신이 교육 받고 있는 내용이 바로 자신이 처한 환경이고 그 환경을 구성하는 요소 중에 자신들이 만들어서 기여한 부분도 있다면 거기서 얻는 교육 효과는 매우 높고 생동감이 있을 것**이다.

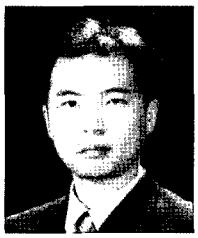
참고문헌

[1] KMU-Linux : http://www.kmulinux.org
[2] WindowsXP : http://http://www.microsoft.com/windowsxp/default.mspx
[3] Openldap : http://www.openldap.org/
[4] ActiveDirectory : http://www.microsoft.com/windowsserver2003/technologies/directory/activedirectory/default.mspx
[5] Camp, J. and Osorio, C., Kennedy School of Govt., Harvard, "Privacy Enhancing Technologies for Internet commerce", 2002
[6] Kerberos : http://web.mit.edu/kerberos/
[7] Microsoft, "Domain Migration Cookbook", 2000
[8] TomBialaski, "NIS to LDAP Transition: Exploring", Sun Microsystems, 2000
[9] Emmanuel Blindauer, Strasbourg "Kerberos : Linux, Windows et le SSO", JRES 2005
[10] Charlie Russel, Microsoft, "Windows Services for UNIX 3.5 White Paper", 2004
[11] Jun Futagawa, Hosei University "Integrating Network Services of Windows and UNIX for Single Sign-On", Third International Conference on Cyberworlds 2004
[12] NFS : http://nfsv4.org/
[13] Samba : http://www.samba.org/
[14] Layne Churchill, CD/CMS Fermilab "Logical Volume Manager", USCMS 2002
[15] William A. Adamson & Kendrick M. Smith, University of Michigan "Linux NFS Version 4: Implementation and Administration" OLS 2001
[16] WSUS : http://www.microsoft.com/windowsserversystem/updateservices/default.mspx
[17] POPTOP : http://www.poptop.org
[18] K. Sueyasu, T. Tabata, and K. Sakurai, "On the Security of Selinux with a Simplified Policy" Communication, Network, and Information Security 2003
[19] SSH : http://www.openssh.com
[20] Andrew Findlay, "Security with LDAP", Skills 1st 2002
[21] Joel West, "Contrasting Community Building in Sponsored and Community Founded Open Source Projects", The 38th Annual Hawaii International Conference on System Sciences
[22] 이철남, "공개소프트웨어 활성화 정책의 현황과 방향" 정보통신정책 제 15권 5호 통권 320호
[23] 송위진, "한국형 오픈소스 소프트웨어 기술개발 전략" 2002. 8
[24] 오픈소스 소프트웨어 활성화 워킹 그룹, "오픈소스 소프트웨어 연구 보고서" 2002. 12

황 선 태

1985 서울대학교 컴퓨터공학과(학사)
1987 서울대학교 컴퓨터공학과(석사)
1996 Manchester University (PhD)
1997~현재 국민대학교 컴퓨터학부 부교수
관심분야 : 공개소프트웨어, e-Science, 그리드시스템, PSE
E-mail : sthwang@kookmin.ac.kr

정 낙 수



2000 php.net 개발자 그룹 회원
2003 국민대학교 컴퓨터학부(학사)
2004 KMU-Linux 개발
2004 국민대학교 전산과학 석사과정
현 재 새한정보시스템 선임연구원
관심분야 : Security, Linux Kernel, LVS
E-mail : nsjun@cs.kookmin.ac.kr

허 대 영

2004 국민대학교 컴퓨터학부(학사)
2005 국민대학교 전산과학(석사)
2006~현재 국민대학교 전산과학 박사과정
관심분야 : 그리드 시스템, 시스템 아키텍쳐, 디자인 패턴, 공개소프트웨어
E-mail : dyheo@cs.kookmin.ac.kr