플랫폼 독립적 성능 개선 소프트웨어 스트리밍 기술 구현 및 성능 평가

정회원 오 창 훈*, 종신회원 전 용 희**°

Implementation and Performance Evaluation of Platform Independent Performance Enhanced Software Streaming Technology

Chang-Hun O* *Regular Member*, Yong-Hee Jeon**° *Lifelong Member*

요 약

소프트웨어 스트리밍 기술은 네트워크에서 스트리밍을 통하여 여러 응용 소프트웨어를 지원할 수 있는 서비스 방법이다. 본 논문에서는 플랫폼 독립적인 성능 개선 소프트웨어 스트리밍(PESS: Performance Enhanced Software Streaming) 기술을 제안한다. 이 기술은 자바 언어를 기반으로 설계하고 구현하였다. 구현 시스템에서의 주요 특징은 윈도우 시스템 이외의 다중 운영 체제에서 사용될 수 있는 플랫폼과 개선된 성능이다.

구현된 스트리밍 방법에서, 응용 소프트웨어는 서버에 위치하며 단지 필요한 팩들만 순간적으로 전송된다. 필요한 경우, 가상 파일 시스템과 클라이언트의 가상 레지스터리에 의하여, 사용자 요청이 매우 작은 팩 단위를 전송함으로써 처리된다. 그러므로 서버 부하가 감소될 수 있고 스트리밍 속도 또한 개선될 수 있다. 본 논문에서는 제안 시스템에 대한 구현 결과를 제시하고 여러 가지의 성능 특성을 분석한다.

Key Words : Software Streaming, Media Streaming, Performance Evaluation, Characteristics Analysis

ABSTRACT

Software streaming technology is a service method which can support several application software via streaming in networks. In this paper, we propose a platform independent PESS(Performance Enhanced Software Streaming) technology. We design and implement the technology based on Java language. The main features in the implemented system are both platforms to be used in multiple operating systems in addition to Windows system and enhanced performance.

In the implemented streaming method, application software is placed on the server and only necessary packs are transmitted in an instant. By virtual file system and clients' virtual registry, if necessary, the users' request is processed by transmitting a very small pack unit. Therefore, server load can be reduced and the streaming speed can also be improved. We present the implementation results and evaluate several performance characteristics of the proposed system.

Ⅰ. 서 론

컴퓨터의 급속한 보급은 조직 및 개인의 일상생활

* 한국소프트웨어개발원(ksoftceo@paran.com)
** 대구가톨릭대학교 컴퓨터정보통신공학부(yhjeon@cu.ac.kr, (° : 교신저자)
논문번호 : KICS2010-09-450, 접수일자 : 2010년 9월 14일, 최종논문접수일자 : 2011년 4월 20일

과 다양한 업무환경에서 혁신적인 변화를 가져왔다. 소프트웨어의 고성능화 및 다양화(새로운 버전의 등장 등)는 그 구입비용의 증가를 가져와 조직의 정보운영 비용에 상당한 부담이 되고 있다. 이런 소프트웨어의 고성능화는 결국 하드웨어의 잦은 교체를 가져와 자원낭비와 지구환경 파괴를 가속화하고 있다[1]. 또한 조직 및 개인은 불법 소프트웨어 사용에 따른 저작권 문제 등으로 법적분쟁소지를 만들기도 한다[2].

이러한 여러 가지 문제를 해결할 수 있는 방법으로 소프트웨어의 중앙집중식 관리를 통한 소프트웨어의 효율적인 관리가 가능한 PESS (Performance Enhanced Software Streaming) 기술을 제안한다. PESS 시스템은 근본적으로 소프트웨어의 불법사용을 방지하면서 사용자가 필요한 소프트웨어를 편리하고 효율적으로 사용하게 되고, 소프트웨어를 개발하는 기업들은 지속적으로 소프트웨어를 연구개발 할 수 있어 관련 산업의 지속적인 발전이 가능하리라 판단한다.

컴퓨팅을 위한 기술로는 전통적 소프트웨어 인스톨 방법 외에도 최근 출현한 네트워크 컴퓨터 형태인 씬 클라이언트 컴퓨팅 기술, 웹 기반 터미널서비스, 버추얼 컴퓨팅, 클라우드 컴퓨팅 등이 있다. 이런 기존 기술의 효율적인 측면도 있지만 활용에는 여러 가지 제한점이 있다.

본 논문에서 제안된 PESS 방식은 스트리밍 방식의 중앙 집중식 관리를 통하여 기존에 투입되던 시간 및 자원을 최소화 하면서 정확하고 빠르게 소프트웨어를 관리할 수 있으며, 언제 어디서나 원하는 소프트웨어를 설치 없이 편리하게 사용할 수 있다. 또한 한정된 자원을 적극 활용하여 기존 인프라에 재투자를 하지 않고도 최고의 효율을 낼 수 있도록 설계되었다. 이를 통하여 조직의 전산관리 비용을 획기적으로 절감하고 소프트웨어 낭비 및 관리업무의 비효율을 최소화 할 수 있다.

본 논문의 나머지 구성은 다음과 같다. 제 Ⅱ장에서는 관련연구로 기존 관련기술의 기술특징에 대하여 분석한다. 제 Ⅲ장에서는 시스템 설계 및 구조에 대하여, 제 Ⅳ장에서는 제안된 PESS 시스템의 구성요소 및 프로토콜을 제시하고, 제 Ⅴ장에서는 성능 평가 결과를 제시하고, 마지막으로 제 Ⅵ장에서는 본 논문의 결론 및 향후 연구 과제를 제시한다.

Ⅱ. 관련 연구

2.1 컴퓨팅 환경의 변화

전통적 방법인 인스톨의 의미는 '설치'란 의미로, '어떤 장소에 새로운 컴퓨터 시스템을 도입하여 동작하도록 하거나 이미 존재하는 컴퓨터 시스템에 새로운 하드웨어나 소프트웨어를 추가하는 일'을 일컫는다. 인스톨방식은 필요한 소프트웨어를 개별 디스켓이나 CD롬을 통해 직접 설치하는 경우와 인터넷 또는 네트워크를 통해 웹서버나 조직의 중앙서버로부터 다운로드 후 설치하여 사용하는 구조이며, 이 방식은 컴퓨터가 도입 된 후 지금까지의 컴퓨터 사용 환경전반의 일반적인 방법이다. 1970년대에서부터 1980년대까지 초기컴퓨터 환경에서는 소프트웨어 패키지 형태의 운영방식이었고 1990년대부터는 웹 또는 서버의 자원으로부터 다운로드 받은 자원을 클라이언트 컴퓨터에 설치하는 방식으로 발전하였다[3].

1970년대부터 현재에 이르기까지의 컴퓨팅 환경의 다양한 변화의 형태를 그림 1에서 보여주고 있다. 1970년대는 Central Power & Thin Terminals 형태의 중앙집중식 컴퓨팅 유형에서 1980년대의 Transactions & Fat Client한 클라이언트 서버 컴퓨팅 유형이었다[4]. 1990년대 이후부터 n-Tier Web과 Thin Web이 가미된 인터넷 공간의 네트워킹 컴퓨팅 시대를 통해 2000년대부터는 웹 서비스의 다양한 기능과 기술이 접목된 유비쿼터스 기술을 지향하는 서버기반 컴퓨팅이 본격 출현하게 되었다.

대부분 기존 방식은 서버에 소프트웨어를 설치해야 하고, 클라이언트 컴퓨터는 단순 단말기 또는 최소 사양의 컴퓨터를 사용하고 있어 고사양의 서버를 활용하여 작업을 하고 있다. 유사 스트리밍 기술은 많은 접속자와 고사양의 소프트웨어를 사용하는 경우 대부분 서버에 상당한 부하를 주게 된다.

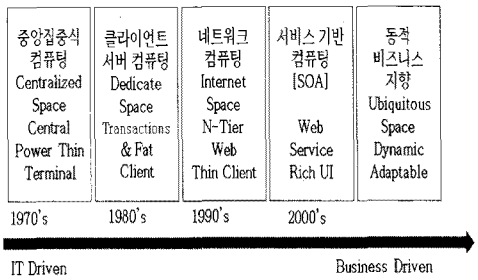


그림 1. 컴퓨팅 환경의 변화 과정

2.2 스트리밍 방식

전통적 의미의 스트리밍 기술은 주로 오디오, 비디오 등 웹 서버의 동영상의 실시간 전송, 실시간 구현을 위해 탄생되고 전제된 것이다[5]. 대용량의 동영상 파일을 동시에 파일 전체를 보내는 것이 아니라, 시간

의 흐름에 따라 해당 분량만큼을 작게 나누어 조금씩 보내게 된다[3,6].

인터넷에서 동영상을 다운로드가 아닌 실시간 전송과, 구현이라는 개념을 가능케 한 것이 바로 미디어 스트리밍 기술이다[3]. 보내는 서버에서 파일을 작게 분할하여 보내면 받는 컴퓨터에서는 분할된 파일 조각들이 들어오는 대로 바로 재결합되어 원래의 파일로 재생하는 방식이다[7]. 스트리밍은 전송되는 데이터를 마치 끊임없고 지속적인 물 흐름처럼 처리할 수 있는 기술을 의미한다.

2000년 이후 들어와 소프트웨어 스트리밍 관련기술들이 시장에 출현하게 되었다[3,8-10]. 전 세계적으로는 미국의 AppStream, Softricity사의 SoftGrid, Stream Theory, IBM, Microsoft, Sun Microsystems 사 등에서 여러 가지 관련 기술 및 전략을 발표하고 있으나 현재까지 국내에 상용화 또는 레퍼런스가 전혀 없는 상황이다[11-17]. 뿐만 아니라 AppStream, Softricity사의 SoftGrid, Stream Theory, IBM, Microsoft, Sun Microsystems사의 개발기술들은 대부분 마이크로소프트사의 윈도우 기반에서만 구동되고 있는 실정이다.

Ⅲ. 시스템 설계와 구조

3.1 설계

PESS 시스템에 적용한 운영체제 관련기술은 가상 파일시스템(VFS), 가상페이징, 시스템 관련 디바이스 드라이버기술이며, 표준 인터넷 실시간 전송 프로토콜(RTP), 실시간 스트리밍 프로토콜(RTSP), 서비스검색 프로토콜(SDP), MMS 등의 스트리밍 프로토콜 기술을 또한 사용하였다. 그리고 EXE, Byte Code 관련 실행파일 포맷기술, JAR, ZIP 등 소프트웨어 패키징 포맷기술, OS, CLI, JVM, WIPI 등의 가상 수행환경 처리기술, 그리고 RSA, DES 등의 암호화기술 등이 있다[18]. 표 1은 제안된 PESS 시스템의 요소 기술을 보여준다.

특히 PESS의 온디맨드 소프트웨어 스트리밍은 운영체제의 가상 메모리 개념을 확장하여 인터넷 서버를 통해 응용프로그램을 실행시키는 기술을 사용하고자 한다[18].

그림 2는 설계 개념도를 보여준다. 클라이언트는 서버상의 PESS 시스템에 접속하여 PESS 클라이언트 모듈을 네트워크를 통해 클라이언트에 설치한다. 설치된 PESS 클라이언트 모듈을 실행시키면 이 장치들을 통해 서버의 PESS와 연결되게 된다.

표 1. 제안된 시스템 요소 기술

요소 기술	세부 요소 기술	내 용
프로토콜	제어 프로토콜	소프트웨어 스트리밍 동작을 제어하기 위한 표준 프로토콜
	전송 프로토콜	소프트웨어 스트리밍 데이터 전송을 위한 표준 프로토콜
	세션 프로토콜	소프트웨어 스트리밍 세션 표현 처리를 위한 프로토콜
패키징 포맷	소프트웨어실행 파일 포맷 기술	소프트웨어 온디맨드 서비스에 최적화된 소프트웨어 바이너리 포맷기술
	소프트웨어실행 파일 분석기술	고급 기능 제공 및 기술을 네트워크로 확장한 기술
플랫폼	분산 가상 페이징기술	기존 운영체제의 페이징 기술을 네트워크로 확장한 기술
	가상 소프트웨어 실행 환경 기술	소프트웨어를 가상으로 설치하기 위한 기술
보안 기술	암호화 및 저작권보호	소프트웨어 저작권 및 사용자 보호를 위한 기술

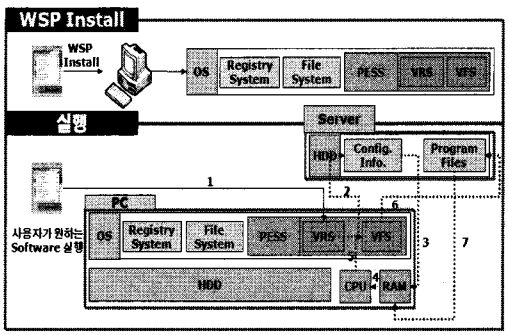


그림 2. PESS 설계 개념도

PESS 클라이언트 모듈이 설치된 컴퓨터들은 서버의 PESS 모듈과 연결되어 클라이언트와 서버가 마치 하나의 컴퓨터처럼 인식하게 된다. 클라이언트는 인증 후 서버상의 필요한 소프트웨어를 선택하면 최초 필수모듈이 클라이언트로 순간 스트리밍 되고 작업이 진행된다. 그 후 사용자 요구에 따라 다음 순위 소프트웨어 파일이 순간 스트리밍 되어 사용자는 클라이언트 컴퓨터가 사용하고자하는 소프트웨어의 인스톨 없이도 가능하다.

제안하는 PESS 기술의 소프트웨어 스트리밍 방식은 그림 3에서 보는 것 같다. 먼저 클라이언트의 컴퓨터의 운영체제에 PESS 시스템이라는 별도의 가상 레지스터 시스템과 가상 파일시스템을 구축하여 기본환경을 만들어놓고 사용하고자 하는 모든 소프트웨어를

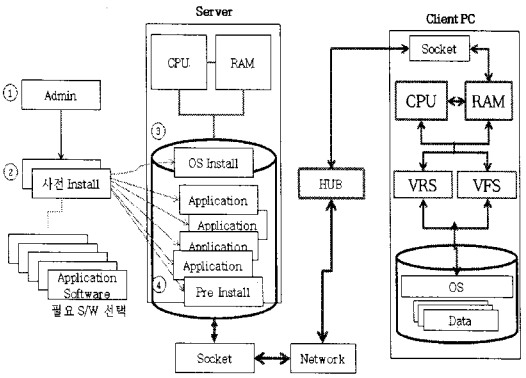


그림 3. 상세 구성도

서버에 단 한번 설치하고 실행 대기하게 된다.

사용자가 원하는 소프트웨어를 실행하면, 클라이언트의 기존의 운영체제의 레지스트리나 파일시스템과 상관없이 이미 만들어진 가상의 소프트웨어인 PESS 플레이어를 통해 실행된다. 즉, 가상 레지스터 시스템과 가상 파일시스템을 통해 실행된다. 네트워크를 통해 서버상의 하드디스크 장치내의 구성환경 중 필수 요소만 순간스트리밍을 통해 클라이언트 컴퓨터 메모리에 순차적으로 적재하게 된다. 사용자 요구에 따라 서버에 설치된 파일의 필수 팩만 아주 작은 단위로 나누어 순간 스트리밍 방식으로 네트워크를 통해 클라이언트 컴퓨터 메모리에 로딩한 후 클라이언트 컴퓨터의 중앙처리장치를 통해 실행하게 되는 방식이다.

PESS 방식에는 기본적으로 서버와 네트워크가 필요하고 그 서버에는 시스템 전체를 관리하는 관리서버와 실질적인 사용 소프트웨어를 별도로 패킹하여 서비스하는 컨테이너서버가 있다. 두 서버는 별도로 두거나 소형 시스템에서는 한 서버에 동시에 설치하여 사용하는 경우도 있다. 각각의 서버가 구축되고 관리자는 각종 사용자 정보, 클라이언트가 사용할 소프트웨어를 별도로 전송이 용이하도록 패킹하여 사전 인스톨하여 준비한다. 사용자는 클라이언트 컴퓨터에서 관리서버에 접속하여 사용자 권한을 획득 후 전용 PESS 모듈을 전송받아 설치를 완료하여 연결통로를 만든다. 그 후 컨테이너서버에 접속, 작업에 필요한 애플리케이션을 선택하여 작업이 실행된다. 그림 4는 그 실행절차를 보여준다. 이때 접속서버는 기존방식과는 달리 일반 PC급 저사양의 서버와 접속하는 네트워크가 높은 성능이 아니라도 실행에 별 문제는 없다. 다만 서버 및 네트워크와 클라이언트 컴퓨터가 높은 사양 일수록 실행의 속도는 빨라지게 된다.

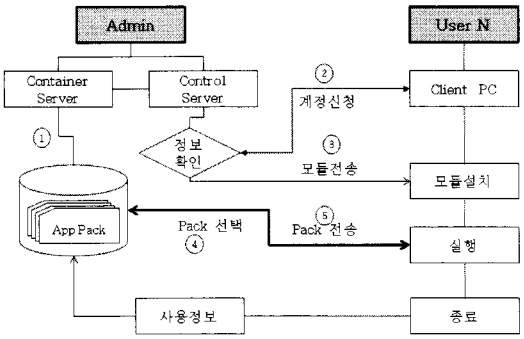


그림 4. 기술의 실행 절차

3.2 시스템 구조

PESS 시스템 중요 구성요소는 다양한 기능들의 효과적인 연결을 통해 효율적인 서비스를 제공하게 된다. 그림 5는 시스템 구조를 보여준다.

각 기능을 요약하면 표 2와 같다.

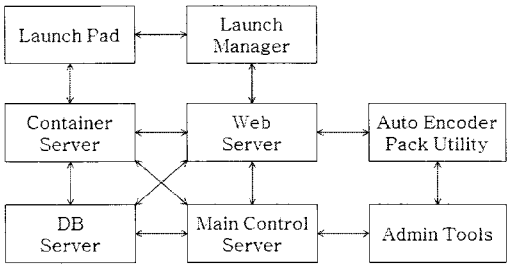


그림 5. 시스템 구조

표 2. 구성요소 역할

구성요소	역 할	서비스 내용
서비스 구성 요소	컨테이너 서버	실질서버역할, 자료전달, 데이터처리
	런치관리자	버전제어, 업그레이드
	런치패드	사용자 애플리케이션 서비스
	웹 데스크 탑	사용자 역할 인터페이스
관리 구성 요소	중앙 관리 서버	중앙 관리자 역할
	관리자	서버설정, 관리, 사용자 책 관리
	팩 인코더	사용되는 컨텐츠(팩)생성 기능
	팩 유틸리티	내부 데이터 업데이트 수행

3.3 시스템 요구조건

시스템 서비스를 제공하는 서버들의 하드웨어적인 성능과 소프트웨어적인 주요 요구사항을 요약표로 정리하면 표 3과 같다.

표 3. 시스템 주요 요구사항

	하드웨어 요구사항	소프트웨어 요구사항
서버 사양	Intel Pentium Ⅲ 700 MHz 이상	OS는 자바 가상머신이 설치되는 모든 종류지원 (윈도우 2k, Res Hat 권장)
	메인 메모리 512MB이상 (1GB권장)	웹 서버는 IIS5.0 (Win 2k) Apache(리눅스)
	하드디스크 10GB (30GB권장)	DBMS는 Oracle, MS-SQL My SQL(Default)
	네트워크 대역폭 T1급 이상	JRE 1.3 이상 요구
사용자 사양	Intel pentium II 300MHz이상(450이상 권장)	OS는 윈도우 95이상 계열
	RAM 64M이상(권장 128MB이상)	웹 브라우저 Internet explorer 5.0이상 (5.5이상 권장)
	하드 디스크는 최소 500MB이상	서비스 애플리케이션에서 필요로 하는 DirectX7.0 이상, Microsoft Media Player7이상(선택적)
	네트워크 대역폭은 최소 128Mbps이상(xDSL, Cable Modem, etc)	

Ⅳ. 시스템 중요 기능 및 프로토콜

4.1 중앙 관리자 서버

중앙 관리자서버의 주요 역할은 컨테이너서버 관리, 관리자 툴에 대한 관리용 기능제공, 팩 파일관리 및 컨테이너서버로 팩을 분배하는 것 등이다. 부가적인 기능으로는 전체 사용자의 서비스 애플리케이션에 대한 사용권한을 설정할 수 있으며, 전체 컨테이너서버들에 대한 부하균등 기능도 제공한다. 그림 6은 관리자 정보관리 프로토콜을 보여준다.

먼저, 컨테이너서버 정보관리기능은 모든 컨테이너서버는 중앙 관리자서버와 연결되어 있어야 한다. 중앙 관리자서버는 컨테이너서버가 새롭게 구동이 되었거나, 서비스를 중지했는지를 실시간으로 검색하여 그 상태를 DB에 저장 한다. 또한, 컨테이너서버의 상태를 가지고 관리서버 시스템 서비스의 부하균등(load balancing) 역할을 수행한다. 중앙 관리자서버가 작동하지 않는 상황이더라도, 컨테이너서버는 정상적으로 서비스를 마무리 지을 수 있도록 연결을 유지한다.

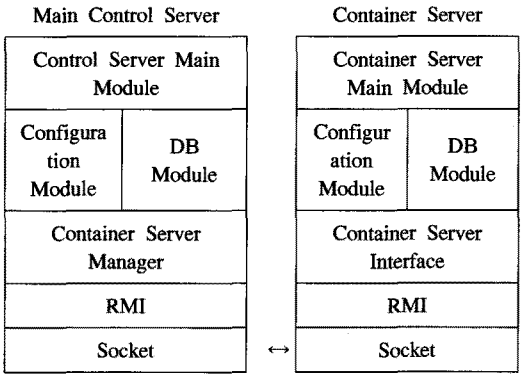


그림 6. 관리자 정보 관리 프로토콜

4.2 컨테이너 서버

컨테이너 서버의 기본적인 역할은 런치패드가 요청하는 서비스 애플리케이션 자료를 전달해주는데 있다. 시스템 구성요소 중, 진정한 의미의 서버라고 할 수 있다. 과금 정보는 런치패드가 시스템 서비스를 사용하고자 시도할 때 기록되며, 컨테이너 서버에서는 그 최종 사용자에 대한 과금 정보를 DB에 기록한다. PESS시스템 서비스를 좀 더 효율적으로 제공하기 위해서, 중앙 관리서버는 컨테이너 서버들을 중앙에서 관리한다. 컨테이너 서버는 중앙 관리 서버가 관리에 필요한 정보를 전달하게 되면 중앙 관리 서버는 그 정보를 토대로 부하균등을 시도한다.

4.3 관리자 툴

관리서버의 환경설정 기능은 시스템 서버 - 팜 상태 정보 모니터링기능, 팩의 시스템 서비스등록 / 삭제 / 정보 관리기능, 다양한 시스템 서버 - 팜 제어기능, 관리자를 위한 유저 인터페이스 제공기능이다. 컨테이너 서버 환경설정은 중앙 관리서버에 관한 설정 값들은 대부분 설정파일에 들어있는 내용을 관리자가 직접 조작하여 변경을 시도했으나, 직접 조작해야 하는 불편함과 조작내용에 대한 신뢰성문제로 인하여 관리자 툴을 이용해 필요한 설정 값을 변경할 수 있도록 하였다.

4.4 런치 관리자

시스템 런치관리자는 기본적으로 런치패드가 필요로 하는 파라미터 전달과 버전관리를 수행하는 추가 프로그램으로 볼 수 있다. 첫 번째 주요기능은 런치패드 버전관리로 런치패드 구성물 중, 시스템을 재 시작해야 작동할 수 있는 구성요소도 있기 때문에 각 버전별 충돌을 막고 시스템의 원활한 서비스제공을 위해

업그레이드와 다운 그레이드를 자동으로 수행한다.

두 번째 런치패드는 일종의 시작프로그램으로 서비스 애플리케이션을 실행시키기 위해서 컨테이너 서버로부터 라이센스 인증을 받아야 하며, 서비스 애플리케이션 작동에 필요한 실행인자들이 필요하게 되는데, 이러한 파라미터 전달 작업 또한 런치관리자가 수행한다.

4.5 런치패드

시스템 런치패드는 시스템 서비스에 있어서 클라이언트 프로그램 역할을 수행하며, 크게 세 가지 모듈로 나누어져 있다. 먼저, PESS시스템의 가상 디바이스 드라이버 모듈인 가상 파일시스템과 가상 레지스터 시스템, 그리고 서비스 애플리케이션 프로세스 컨트롤 모듈인 진행관리로 나누어지며, 서비스 애플리케이션을 실행시키는 것이 런치패드의 가장 주된 임무이다.

시스템 런치패드의 주요 역할은 실시간으로 컨테이너 서버로부터 데이터를 전달 받아 서비스 애플리케이션을 실행 시키는데 있다.

4.6 서비스의 제어흐름

PESS 시스템의 구현을 위해서는 반드시 PESS 시스템의 제어흐름을 이해하는 것이 중요하다. 다음의 단계별 설명은 최종사용자가 서비스 애플리케이션 사용을 시도할 때, 이루어지는 구성요소 들의 동작흐름이다.

- 최종사용자는 시스템 웹 데스크 탑을 통해 사용할 서비스 애플리케이션을 선택한다.
- 시스템 웹 데스크 탑은 선택한 서비스 애플리케이션의 해당 정보를 DB로부터 검색한다.
- 시스템 웹 데스크 탑은 검색한 정보를 런치관리자에게 전달하여 최종적으로 런치패드가 사용 할 수 있게 한다.
- 런치패드는 런치관리자로부터 넘겨받은 파라미터, 시스템 서비스를 사용하는 데 필요한 정보를 번역한다.
- 필요한 정보를 번역하여 런치패드는 그 정보를 가지고 컨테이너 서버에 접속 하여 서비스 애플리케이션 실행에 필요한 데이터를 요청하게 된다.
- 런치패드는 필요한 데이터만 컨테이너 서버에 요청하기 위해 필요한 가상 파일시스템인 가상 파일시스템을 초기화 한다. 아울러, 가상 레지스트리 시스템이 가상 레지스터 시스템도 작동하게 된다.
- 런치패드는 가상 파일시스템으로부터 서비스 애플리케이션을 실행시킨다.
- 실행 도중 필요한 파일 데이터는 가상 파일 시스템으로부터 처리된다. 캐시 데이터에 존재하는 경우에는 로컬시스템에서 처리되며, 캐시 데이터에 존재하지 않는 경우에는 컨테이너 서버에게 요청하게 된다.
- 서비스 애플리케이션이 종료되는 경우, 가상 파일시스템과 가상 레지시터 시스템은 각각 다음 실행에 사용할 캐시 데이터와 레지스트리 데이터를 저장하게 된다.
- 컨테이너 서버는 최종 사용자의 서비스 애플리케이션 사용 정보를 계산하여 과금 정보와 함께 해당 데이터베이스에 저장한다.

4.6.1. PESS 시스템 서비스 기본 제어흐름

관리자는 서버에 사전 필요한 애플리케이션 소프트웨어를 팩 형태로 설치해 놓고 사용자가 접속하여 일정한 절차를 통해 계정을 받고 클라이언트 모듈을 설치한다. 작업에 필요한 애플리케이션을 선택하고 사용자 요구에 따라 클라이언트로 전송되는 팩을 통해 필요한 작업을 진행 할 수 있다. 종료 후 데이터는 클라이언트 컴퓨터에 저장 하는 것으로 종료되며 사용 정보는 서버에 저장된다. 그림 7에서는 기본 프로토콜을 보여준다.

4.6.2 PESS 서비스 설정 단계 프로토콜

중앙 관리 서버가 작동될 경우 그림 8처럼 시스템 서비스에 대한 여러 설정 값을 체크하게 된다. 초기화 과정에서 설정하는 값 들은, 팩 리스트, 컨테이너 서버 리스트, 데이터베이스 포트 검색, 시스템 서버들이 사용하는 포트 검색 등이 포함된다. 라이센스 정보반영 단계는 라이센스 관리항목인, 기한 만료일, IP, 동시 사용자 수 등의 항목을 검색하여 관리 서버가 가지고 있는 정보와 비교한다. 상이할 경우에는 서비스를 시작 할 수 없다. 환경설정 정보 반영 단계는 중앙 관리 서버가 가지고 있는 환경설정 항목인, 데이터 디렉토리, 로그레벨 설정 값, 데이터베이스 사용 여부, 각종 네트워크포트 설정, 컨테이너서버 IP 리스트, 만료기한 등을 참조하여 동작에 반영시킨다. 데이터베이스 서버 접속단계는 위의 각종 정보들이 반영되고 나면, 데이터베이스에 접속하여 필요한 데이터를 참조해야 한다.

참조가 필요한 데이터는 팩 리스트에 대한 자료와 컨테이너 서버 IP 주소 데이터이다. 팩 체크 단계는 중앙 관리 서버에 들어 있는 팩을 검색하여 데이터베이스로부터 알아낸 팩 리스트와 비교를 한다. 새로 등록되거나, 수정, 삭제된 팩에 대한 정보를 변경한다.

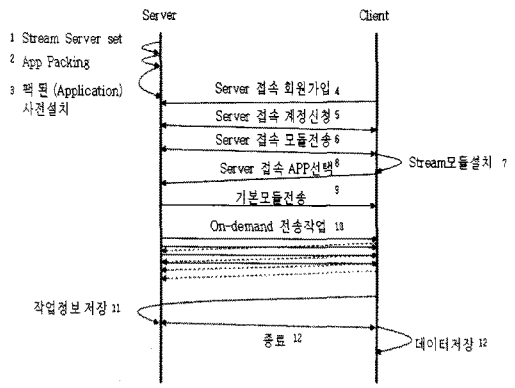


그림 7. 기본 프로토콜

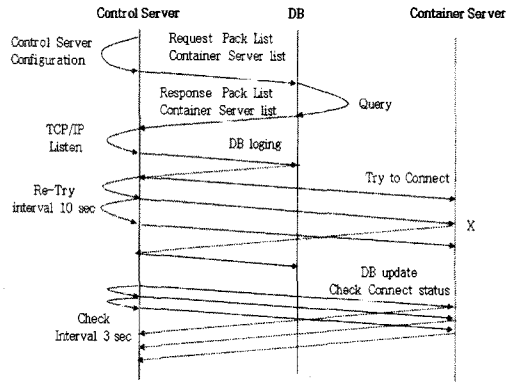


그림 8. 서비스 설정 단계 프로토콜

시스템 구성요소 네트워크 포트 검색 단계는 중앙 관리 서버가 구동되면서 각 구성요소들과 통신이 필요한데, 이때 통신에 필요한 네트워크 포트를 검색하여 활성화여부를 점검한다. 컨테이너 서버 접속검색 단계는 위에서 찾아낸 컨테이너 서버들에 대해서 접속을 시도한다. 접속을 시도할 경우에는 매 10초마다 접속을 시도하며 접속이 성공한 경우에는 3초마다 접속상황을 체크한다. 접속이 이루어지면 부하균등을 위해서 데이터베이스의 해당 필드에 접속여부에 대해서 기록한다.

4.6.3 서비스 준비 단계 프로토콜

이 단계에서는 컨테이너 서버들이 초기화과정을 거치고 동작되며, 중앙 관리 서버와의 통신을 통해, 사용자에게 시스템서비스를 수행할 수 있다.

컨테이너 서버 출발 단계는 컨테이너 서버가 실행되기 시작한 단계이며, 중앙 관리 서버와 유사하게 환경 설정 값을 검토하여 구동하는데 사용한다. 로그기록 프로세스도 가동된다. 팩 체크단계는 컨테이너서버가 가지고 있는 팩 보관정보를 서비스가 가능한 상태로 초기화 하며, 설치되어 있는 팩의 정보를 체크한다. 데이터베이스 접속단계는 위의 각종정보들이 반영되고 나면, 데이터베이스에 접속하여 필요한 데이터를 참조해야 한다. 참조가 필요한 데이터는 팩 리스트에 대한자료, 컨테이너 서버 IP주소 데이터, 접속로그 데이터, 사용자정보 등이다. 시스템 구성요소 네트워크 포트 검색 단계는 컨테이너서버가 작동되면서 각 구성요소들과 통신이 필요한데 이때 통신에 필요한 네트워크 포트를 검색하여 활성화 여부를 점검한다.

기타 다음과 같은 프로토콜이 있다. 컨테이너 서버 추가 프로토콜, 팩 / 레지스터 추가 프로토콜, 클라이언트 컴포넌트 설정 프로토콜, 원 클릭 실행 프로토콜.

핵심 과정을 요약하면 다음과 같다. 중앙 관리 서버는 시스템서비스의 중앙 관리자 역할을 수행한다. 시스템서버- 팩 관리, 사용자 관리, 웹 구성요소 관리 등을 수행하며, 시스템 서비스의 부하균등 또한, 중앙 관리 서버의 역할이다. 컨테이너 서버는 시스템 서비스의 실질적인 서버로서의 역할을 수행한다. 런치패드가 요청한 자료를 네트워크를 통해, 실시간에 전달해 주는 것이 주목적이다. 또한, 최종 사용자가 시스템 서비스를 사용한 과금 데이터를 처리하는 역할도 수행한다. 런치관리자는 런치패드의 버전을 제어하며, 런치패드의 설치 및 업그레이드, 서비스 애플리케이션별 파라미터 전달 등의 기능을 제공한다. 런치패드는 최종 사용자가 선택한 서비스 애플리케이션이 동작할 수 있도록 컨테이너 서버에 필요한 데이터를 요청하며, 사용한 서비스 애플리케이션의 캐시 데이터를 재사용하기 위해서 저장하는 기능도 제공한다.

V. 성능 평가

5.1 테스트 베드 구성 및 평가 방법

성능 평가를 위한 테스트 베드 구성도는 그림 9와 같다. 구동 애플리케이션과 생성 유저수를 캐시 데이터의 진행이 없는 상태에서 사용자가 애플리케이션을 구동시킨 후 종료하고 나온 과정을 총 5개의 시나리오를 만든 후 1,000명의 사용자를 표 4와 같이 구분하여 동시에 서버에 접속시켰다. 애플리케이션별로 생성된 사용자들은 약간의 지연시간을 두어 실행하였다.

(1) 서버 부하 테스트

컨테이너 서버, 컨트롤 서버의 부하율 시험으로 위의 조건으로 1,000명의 사용자 동시에 접속하여 5가지의 애플리케이션을 실행시켰을 때의 중앙처리장치의 부하도를 조사한다.

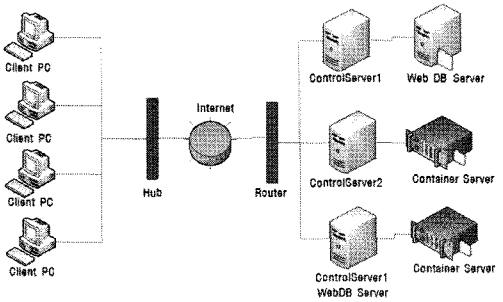


그림 9. 테스트 베드 구성도

표 4. 구동 애플리케이션과 생성 사용자 수

구 분	생성 User	비 고
Winzip	300	30%
한글 97	150	15%
오피스워드	150	15%
페인트 샵 프로	200	20%
드림위버	200	20%
총 User 수	1,000	100%

(2) 메모리 사용률
상기 조건으로 각각 서버에 동시에 접속하여 실행했을 경우의 메모리 율을 시험한다.

(3) 응답 속도(로그인, 애플리케이션)
상기 조건의 각각의 접속자 접속 로그인 시간 및 애플리케이션의 응답 시간을 측정한다.

(4) 변형된 응답 속도(캐시데이터 전송 후)
기존 방식에서 변형하여 100명의 클라이언트로부터 사전 캐시 데이터 전송 전/후의 응답 속도를 측정한 시험(Push Cache 측정)이다.

5.2 결과 및 평가

5.2.1 컨트롤, 컨테이너 서버의 중앙처리장치 사용률

약 1시간 30분 동안 1,000명의 사용자가 접속한 결과 컨트롤 서버, 컨테이너서버(1), 컨테이너서버(2)에 대한 중앙처리장치 사용률은 그림 10과 같다.

대체로 각 서버 모두 20% 이하의 중앙처리장치 사용률을 나타내고 있으며, 최고 정점 시간에서 40%를 약간 넘고 있지만. 컨트롤 서버는 사용자가 로그인을 시도할 때 중앙처리장치를 거의 25% 이하로 사용하다가 사용자 접속이 완료 된 후에는 10% 미만으로 감소하였다. 이는 컨트롤 서버에 웹서버와 데이터베이스 서버를 같이 구동시키고 있기 때문에 나타나는 현상으로 볼 수 있다.

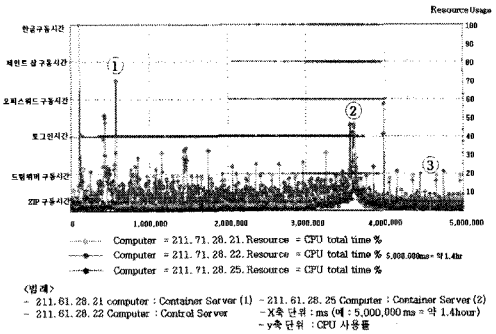


그림 10. 컨트롤, 컨테이너 서버의 중앙처리장치 사용률

컨트롤 서버는 전체적으로 25% 미만의 중앙처리장치 사용률을 보이다가 접속이 가장 많이 생길 때 40% 정도 사용하다가 다시 감소하는 것으로 나타났다.

시험 내용을 좀 더 상세하게 설명하면, 로그인 시간을 제외한 경우는 접속자의 접속 상태가 수시로 변동하고 있고, ①부분의 접속율이 70%에 가까워지는 부분은 시험 시작지시와 동시에 전 접속자가 거의 동시 로그인 한 경우이다. ②부분의 40%를 넘는 경우는 시험기간 중 접속 후 다양한 애플리케이션을 작동하다가 정해진 표 4정책에 전원 동시 정해진 모든 애플리케이션을 가동한 결과 순간적으로 약 40%를 약간 넘는 접속 율(CPU사용 율)을 나타내고 있다. 이후 전 접속자들이 작업을 진행하고 있어도 그림 10의 ③부분과 같이 평균 10%대의 서버 중앙처리장치 활용 율(부하)을 보여주고 있다. 일반 PC급 서버에서 전혀 서버 부하 없이 가동되고 있는 시험결과를 보여주고 있다. 전체적으로 컨트롤 서버는 컨테이너 서버보다 약간 중앙처리장치 사용률이 높은 경향을 보이고 있다(접속자 수, 애플리케이션 사용내역 등의 로그 남음). 만약 고급 사양의 서버였다면 이보다 낮은 부하율을 나타낼 것으로 예상된다.

5.2.2 컨트롤, 컨테이너 서버의 메모리 사용률

메모리 사용률은 그림 11과 같다. 대체로 컨테이너 서버의 메모리 사용률은 큰 변동 없이 일정한 비율로 사용되고 있음을 알 수 있다. 컨트롤 서버는 동시 접속자가 증가될수록 메모리 사용률이 증가되는 경향을 보이고 있어 접속자 수에 따라 변동률이 의존적임을 알 수 있다. 참고로 컨테이너 서버(1)가 컨테이너 서버(2)보다 전체적으로 메모리 사용률이 크게 나타난 것은 도구 및 기타 메모리 상주프로그램이 같이 구동되어 있기 때문이다. 좀 더 상세하게 설명하면 컨테이너 서버의 각 평균 메모리 사용 율은 최고 시점이

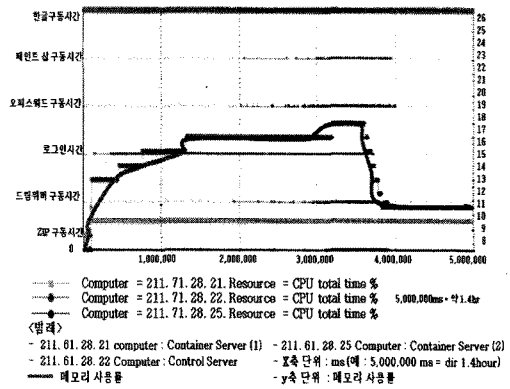


그림 11. 서버별 메모리 사용률

18%정도로 거의 서버의 메모리 부하나 점유율은 별로 크지 않은 것으로 나타났다. 그림 10의 CPU시험과 마찬가지로 최고 지점을 지난 경우도 전체 접속자가 정해진 애플리케이션을 활용하고 있는 중에도 10~11% 정도의 점유율을 보이고 있다.

5.2.3 응답시간 비교

(1) 로그인 시간

웹 서버에 대한 로그인 응답률은 표 5와 같다. 다음에서 보는 바와 같이 10명의 사용자의 접속에 비해 1,000명의 사용자가 약 240배 로그인 반응 시간이 나타남을 볼 수 있다. 위 결과는 시스템 자체의 결과라기보다는 웹서버(아파치서버)의 성능과 하드웨어의 성능에 더 깊이 관련 된 것으로 보인다. 따라서 하드웨어 성능을 높여서 로그인 응답 시간을 줄 일 수 있는 부분이다.

시험결과에서 나타나는 수치는 10명에 비해 240.7%의 응답률을 나타나지만 앞서 살펴본 각각의 컨테이너 서버나 관리자 서버의 부하율이나 메모리 사용 율에 대해서는 별로 부담을 주는 결과는 아니라고 판단한다.

표 5. 접속자에 따른 로그인 응답률(단위: %)

구분	Performance 1	Performance 2
로그인 시간	1	240.7

(2) 애플리케이션 응답률 비교

약 1시간 30분 동안 1,000명의 사용자가 접속한 결과 컨테이너 서버에 대한 애플리케이션 응답율은 표 6과 그림 13과 같다.

10명의 사용자가 접속하여서 각각 하나의 애플리케이션을 구동한 반응 시간은 10명이 접속한 것에 비

표 6. 접속자수에 따른 애플리케이션 응답률6

구 분	Performance 1 (10User)	Performance 2 (1,000User)
Winzip	1	10.65
드림위버	1	5.28
오피스워드	1	9.29
페인트샵프로	1	7.47
한글	1	8.32

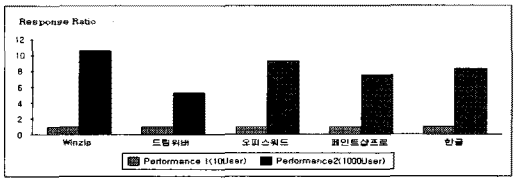


그림 12. 접속자수에 따른 애플리케이션 응답률

해 그림 12에서 보는 것과 같이 Winzip은 10.65배, 드림위버는 5.28배, 오피스-워드는 9.29배, 페인트샵 프로는 7.47배 , 한글 97은 8.23배 반응시간이 나타난 것을 알 수 있다.

이 실험은 서버나 클라이언트의 성능에 그다지 중요한 요인이 되지 못한 결과이지만 참고적으로 각 애플리케이션의 응답시간의 결과 치를 살펴본 것이다. 일반 클라이언트 컴퓨터에서도 각 애플리케이션의 특성에 따라 부팅 속도의 차이가 나기 때문이다.

이는 애플리케이션의 크기에 따라 반응 시간의 차이가 달리 나타났으며, 특히 Winzip이 사이즈에 비해 반응시간이 10.65배나 나온 이유는 전체에서 Winzip을 구동한 사용자가 30%로 가장 많았기 때문이다.

(3) 각 애플리케이션에 대한 반응

실제 본 시험에서 서버로부터 클라이언트 컴퓨터로 별도로 100명의 접속자가 상용 애플리케이션의 정책과 상관없이 각자 필요한 애플리케이션을 작동시켜 사전 Push Cache(모듈을 시험 전 사전에 일부 실행파일을 클라이언트 컴퓨터로 전송받아 놓은 경우 또는 1차 실행 후 재차 실행 시 일어나는 현상)한 전/후 반응이 표 7, 그림 14, 그림 15, 그림16과 같다.

표 7은 Push Cache 적용 전보다 Push Cache 적용 후 평균 51% 정도 실행 속도가 빨라진 것을 보여준다. 기본적으로 본 시스템은 한번 실행 후 재차 실행 경우는 실행속도가 현저하게 빨라지는 것이 특징이지만, 강제 Push 경우는 그림 13에서처럼 모든 애플리케이션에서 평균 성능 개선효과가 있음을 보여준다.

표 7. 각 애플리케이션 별 성능

소프트웨어 시간	Push Cache 적용 전	Push Cache 적용 후
HWP	24.6	12.8
PPT	20.8	10.2
Excel	26.6	14.6
알집	14.2	7.3
페인트샵	18.3	8.2
평균속도	21.3	10.0
비율	100%	49%

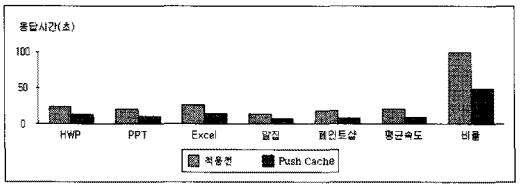


그림 13. 각 애플리케이션별 성능 그래프

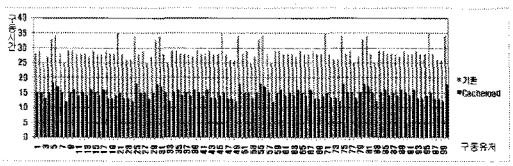


그림 14. 사용자 별 상세 분석표

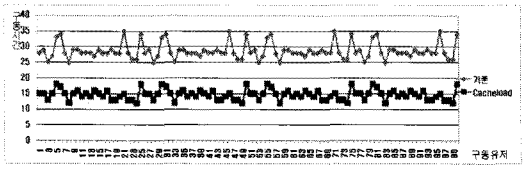


그림 15. 사용자 별 추이 분석

그러나 시스템의 성능 및 상태에 따라서는 다소 차이가 나는 경우도 발생하고 있다. 그림 14에서 보는 바와 같이 같은 서버에 접속하여 시험 한 결과의 차이가 나는 경우는 클라이언트 컴퓨터의 정리 상태와 동일 소프트웨어의 설치 유무도 속도 차이에 영향을 보이고 있다.

실제 클라이언트 컴퓨터에서 실행 시에는 만약 클라이언트 컴퓨터에 동일 종류의 애플리케이션이 설치되어 있고 별도로 지정하지 않으면 우선적으로는 클라이언트 컴퓨터에 설치된 애플리케이션이 실행되고, 서버의 애플리케이션을 실행을 원하는 경우는 별도로 컨테이너 서버의 애플리케이션을 선택하면 실행된다.

PESS 방식으로 실행된 파일을 다시 열고자 하는 경우는 자동적으로 스트리밍 서버와 연결이 되어서 자동으로 연동된 애플리케이션이 활성화 된다. 그림 15는 상세분석의 다른 형태의 표현으로 전체적으로 사전 캐시작업을 한 경우와 현격한 차이가 나는 것을 알 수 있다.

참고로 애플리케이션 실행 방법으로는 서버에 접속해 컨테이너 서버의 애플리케이션을 실행하는 경우, 작업파일을 실행하면 자동으로 연결되어 실행되며, 디렉토리를 찾아 실행할 수도 있는 방법도 있다.

본 시스템은 특별한 경우를 제외하고 클라이언트 컴퓨터가 실행되면 자동으로 활성화 된다.

5.2.4 종합 분석

본 시험결과를 요약하면, 인스톨방식, 서버 베이스 방식, 스트림 기술과는 구동방식 등의 차이로 직접비교 자체는 불가하다. 터미널 서비스나, 씬 클라이언트, 버추얼 컴퓨팅 방식과는 본 시험에서는 직접 비교할 경우가 되지 못하지만 여러 경우를 통해서는 간접적으로 비교해 볼 수 있다.

첫째, 시험결과에서 보여주는 것처럼 일반 저급 PC 급 서버에서도 본 시스템의 서버로서 충분한 역할을 할 수 있어 기존 기술들의 고가의 고사양의 서버들과의 충분한 경쟁력이 있다는 것을 알 수 있다.

둘째로는 일반 클라이언트 방식과 마찬가지로 각 애플리케이션의 응답 속도도 각 애플리케이션의 특성에 따라 부팅(응답)의 속도가 차이가 나지만 본 시스템의 영향으로 일어나는 현상은 아닌 것으로 판단된다.

셋째, 서버로부터 사전 한 번 실행 후 응답 시간이 절반이하로 빠른 응답 시간을 나타나므로 실행 전 보다 빠른 실행을 위해서는 사전 실행 후 작업하는 것이 유리한 것으로 판단된다. 다만 일반 클라이언트 컴퓨터에 인스톨 한 경우와는 어쩔 수 없는 시간적 차이가 날 수 밖에 없다. 서버에 로그인을 해야 하고, 접속 후 약간의 모듈 전송 시간이 존재하고 원격의 네트워크를 경유해야 하므로 약간의 불편함을 감 수 해야 할 부분이다. 다만 일반적으로 업무용 애플리케이션은 웹 서핑처럼 잦은 접속을 원하지 않기 때문에 접속자 대부분이 약간의 속도차이 외에는 특별한 불편한 점은 그다지 없는 것으로 분석된다.

VI. 결 론

본 논문에서는 소프트웨어의 효율적인 활용을 위한 방안으로, 소프트웨어의 중앙집중식 관리가 가능한 플랫폼 독립적인 PESS 시스템을 제안하고, 구현하였다.

기존의 전통적 컴퓨터 사용방법과 유사 스트리밍 방식은 정보관리 비용 및 업무 효율성 향상에 일정부분 기여한바 있지만, 고성능의 소프트웨어 적용 단계에서는 모두 여러 가지 제한점을 가지고 있다.

따라서 기존 시스템을 활용하면서도 소프트웨어의 설치 및 관리의 효율성을 높일 수 있는 PESS 기술을 제안하였다. 이미 멀티미디어 스트리밍 방식과 유사한 일반적인 스트리밍 방식이 존재하고 있으나, 본 논문에서는 다양한 플랫폼에서 실행될 수 있는 플랫폼에 독립적이고, 또한 일반적인 스트리밍의 여러 가지 문제점을 개선하여 보다 편리하고 효율성이 있는 성능이 개선된 PESS 기술을 구현하였다.

제안된 PESS 기술은 응용 소프트웨어를 서버에 위치해 놓고 스트리밍 방식을 통해 순간적으로 전송해 클라이언트의 가상 레지스터리와 가상 파일 시스템을 통해 사용자가 필요할 때 마다 아주 작은 팩 단위로 전송을 함으로써 프로세싱 할 수 있는 방법을 채택하고 있다.

본 논문에서는 제안된 PESS 기술의 특징을 분석하고, 알고리즘, 프로토콜, 서비스 방법 등을 제시하고, 구현된 시스템의 성능 시험과 평가 결과를 제시하였다. 제안된 기술이 보다 다양하고 편리하게 활용될 수 있도록 하기 위하여, 기존 패킹 제작 방법을 시스템화 시켜 자동적인 패킹 시스템화 방안 및 간편화가 필요하고, 네트워크 단절 중에서도 동작하는 기술을 개발할 필요가 있다. 이를 위하여 스트리밍 기술과 클라우드 컴퓨팅 기술을 접목하는 기술의 개발이 필요하다고 판단된다.

참 고 문 헌

[1] 한국정보화진흥원, *국가정보화백서*(National Information White Paper), 2008.

[2] 전진환, 김종기, "지적재산권의 중요성과 소프트웨어 품질이 불법복제에 미치는 영향에 관한연구", *정보화정책*, 제16권, 제1호, pp.54-75, 2009.

[3] Empas, http://itdic.empas.com/view.tsp

[4] http://www.tilon.co.kr/Tuscan/Professonal/infor.

[5] Naver, http://blog.naver.com/yongcoms

[6] Google, http://www.google.co.kr/search/complete.

[7] (주)한국소프트웨어개발원, 소프트온넷(주), "Stream Guide Book", 2003.

[8] 영진출판사, *IT용어사전*, pp.720-730.

[9] 남기혁, 금영섭, 강성주, 허성진, 최완, 김명준, "이동형 소프트웨어 플랫폼 기술 특허 동향 분석 (Anlysis on Patent for Portable Software Platforms)", *전자통신동향분석*, 제23권, 제2호, 2008. 4. 2007-S-015-01(SaaS기반 이동형 개인맞춤 사무환경 구축 기술개발).

[10] A. Dan, B. Levine, B. Lyles, H. Kassan, and D. Balsiefien, "Deployment Issues for the IP Multicast Service and Architecture", *IEEE Network*, 14(1)10-20, Feb. 2000.

[11] AppStream, http://www.appstream.com.

[12] Softricity, Inc. http://www.softricity.com.

[13] Stream Theory, http://www.streamtheory.

[14] Contanvalley, http://www.contantvalley.co.kr/xpaper/document Stream.

[15] Exent Technologies, http://www.exent.com.

[16] http://www.ca.com/Files/SupportingPieces/uni_vision_wp.pdf

[17] D. Eylon, A. Ramon, Y. Volk, U. Raz, and S. Melamed, "Method and System for Executing Network Streamed Application", U.S Patent 6, 574,618, June, 2003. 3.

[18] 한국소프트웨어진흥원, Standardization Roadmap for IT839 Strategy, Ver. 2004-Ver. 2006.

오 창 훈 (Chang-hun O) 정회원

1990년 2월 고려대학교 학사

1990년~1995년 (주)한성

7월 한국소프트웨어개발원 연구소장

2001년 5월~현재 한국소프트웨어개발원 원장

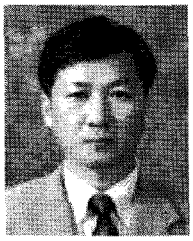
2003년 2월 대구가톨릭대학교 컴퓨터정보통신공학과 공학석사

2010년 2월 대구가톨릭대학교 컴퓨터정보통신공학과 공학박사

2010년 2월~현재 계명문화대학 외래교수

<관심분야> 소프트웨어 스트리밍, 웹 콘텐츠 개발

전 용 희 (Yong-Hee Jeon) 종신회원



1978년 2월 고려대학교 공학사
1989년 8월 미국 노스캐롤라이나주립대 공학석사
1992년 12월 미국 노스캐롤라이나주립대 공학박사
1978년 1월~1978년 11월 삼성중공업㈜
1978년 11월~1985년 8월 한국전력기술㈜
1979년 6월~1980년 6월 벨기에 벨가톰사
1992년 9월~1994년 2월 한국전자통신연구원 선임연구원
1994년 3월~현재 대구가톨릭대학교 교수
2001년 3월~2003년 2월 대구가톨릭대학교 공과대학장
2001년 1월~2006년 12월 한국통신학회 학회지 편집위원
2008년 1월~현재 한국정보보호학회 부회장
<관심분야> 네트워크 보안, 통신망 성능 분석