부록 C

소프트웨어 성능평가 사례

1. 공개소프트웨어 성능 평가 사례 및 프레임워크 ····· 1

1. 공개소프트웨어 성능 평가 사례 및 프레임워크

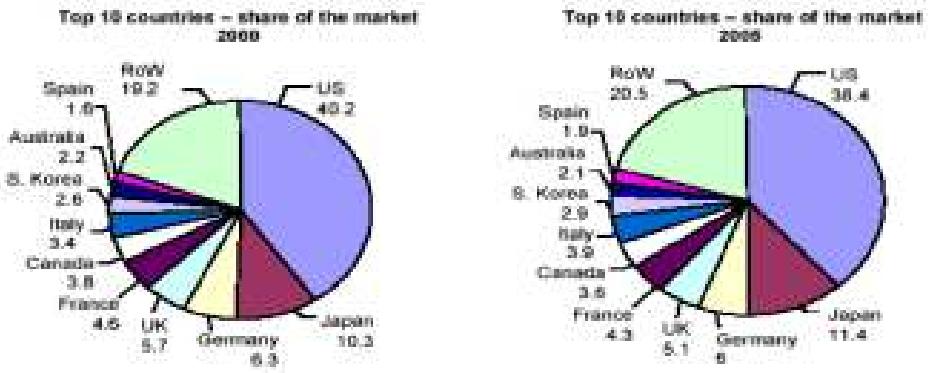
1.1. 공개소프트웨어 성능 평가 사례

1.1.1. 국내 사례

가. 소프트웨어 품질보증[정통기03]

1) 개 요

현재 우리나라는 전체 인구의3분의1이 넘는 국민이 초고속 인터넷을 사용할 정도로 세계적으로 인터넷 강국으로 알려져 있고 또한 정보통신에서는 없어서는 안될 반도체 및 모바일기기 등 H/W에서도 우수한 기술 및 품질을 자랑하고 있다. 하지만 기존의H/W 와 초고속 인터넷상에서 부가가치를 생산해 내는 S/W에서는 <그림1>과 <표1>에서 보는 바와 같이 세계시장 점유율과 수입대비 수출비율에 있어서 아직도 어려움을 겪고 있는 것이 현실이다.



<그림1> 세계 S/W시장 점유율

<표1> 국내 소프트웨어 산업현황 및 전망

구 분	2002	2003	2004	2005	2006	'01-'06 평균성장
생 산	30.53	37.72	46,294	55.87	66,605	22.0
내 수	38.40	45.69	54,458	64.42	75,950	20.0
수 출	121	223	345	481	611	46.0
수 입	721	831	968	1,134	1,324	17.7

자료 : KIPA, 2002. 4

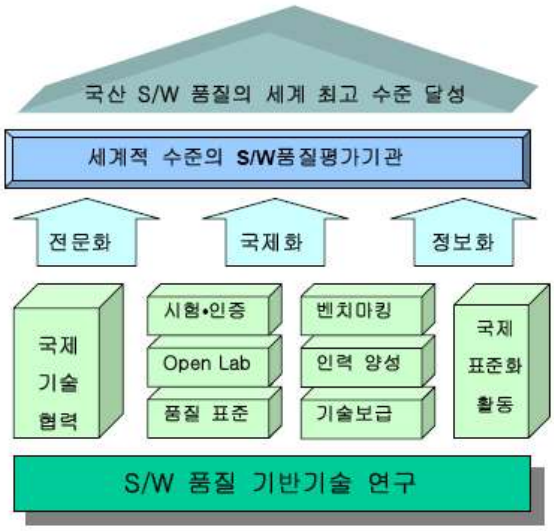
앞으로 2006년에는 400조원 규모 이상으로 성장이 기대되는 세계 S/W시장에서는 현재의 S/W산업 역조현상을 극복하고 보다 활발한 해외 수출의 길을 트기 위해서는 제품의 가격보다는 품질에 초점이 맞춰져야 한다는 것에는 그다지 큰 이견이 없을 것이다. 국내 대기업은 이러한 현안에 대비해서 자체적으로 품질향상을 위한 자구책을 마련해 나가고 있으나 국내 SW 개발업체의 대부분인 중소벤처 기업들에겐 아직까지 힘에 겨운 것이 현실이다.

이에 정부는 소프트웨어산업진흥법 제13조 ①항"소프트웨어의 품질확보 및 유통촉진을 위하여 소프트웨어에 관한 품질인증 실시"를 목적으로 TTA S/W시험인증센터(2001년 12월 ETRI S/W시험센터에서 TTA로 조직이관)를 S/W 품질인증기관으로 지정하였고, 정보통신부 고시 제 2000-81호 '소프트웨어 품질인증기준'에 따라 시험 및 인증을 실시하게 되었다.

2) TTA S/W시험인증센터 활동 현황

우리나라에는 S/W품질인증서비스를 시행하고 있는 기관이 몇 곳 있기는 하나 거의 대부분이 간단한 제품데모 및 문서심사 수준에 머무르고 있는 실정이다. 그 중에서 법적 근거에 의해 설립된 인증기관이면서 S/W 전반에 걸쳐 국제규격을 기준으로 전문적인 시험 및 인증을 하는 기관으로는

TTA S/W시험인증센터가 유일하다. 올해로 4년째를 맡고 있는 국내 S/W품질인증제도와 국내 소프트웨어 제품의 품질 향상 및 국제 경쟁력 확보를 위한 TTA S/W시험 인증센터의 서비스활동에 대해 알아보기로 하자.



<그림2> TTA S/W시험인증센터 서비스활동

<그림2>에서 보는 바와 같이 S/W시험 인증센터는 S/W품질 관련 기반기술 연구를 바탕으로 각종 시험.인증서비스,국제기술협력,국제표준화활동 등을 통하여 국산 S/W의 품질을 국제수준으로 끌어올려 국제경쟁력을 강화하는데 그 목적을 두고 있다.

⊙ S/W품질시험·인증 서비스

시장에 이미 출시되었거나 출시되기 전이라도 개발이 최종 완료된 S/W 제품에 대해서 S/W공급자가 제시하는 프로그램, 제품설명서 및 사용자매뉴얼 등을 대상으로 국제표준을 기준으로 테스트를 실시하고 인증을 하는 서비스이며, 패키지S/W 뿐만 아니라 S/W 전 분야를 대상으로

하고 있다. 시험결과 일정 기준을 통과하여 인증을 획득하게 되면 정통부 S/W산업진흥법에 명시되어 있는 인증마크인 GS마크(Good Software)를 부여하고 인증서를 수여하게 된다.

S/W시험인증센터에서 수행되는 S/W시험은 S/W제품 자체뿐만이 아니라 제품설명서와 사용자 매뉴얼의 충실성 정도, 문서와 프로그램의 일관성,프로그램의 기능 충실성 및 성능,최종 사용자 측면에서의 사용 편의성 등 S/W의 주된 기능 이외 사용자 편의성을 고려한 비기능적 측면들도 매우 중요한 품질평가 대상으로 삼고 있다.

내부적으로 수행되는 평가 및 시험 지침은 ISO/IEC 9126 (S/W품질특성과 매트릭에 관한 국제표준)과 ISO/IEC 12119(S/W패키지의 품질요구사항과 시험에 관한 국제표준)을 따르고 있다. 평가기준이 되는 품질특성과 부특성은 <표2>에서 보는 바와 같다.

인증획득에 대한 효과는 여러 가지가 있으나 그 중에서 주요내용 몇 가지만 열거해 본다면 제3자 시험과정을 통한 품질완성도의 획기적 개선, 각종 평가 및 선정 심사 시 가산점 부여 혜택(조달청 구매,유망중소기업 선정 심사, 중앙정부 S/W개발 프로젝트 발주 시 등), 행정자치부 행정업무용S/W 적합성 시험 등에 반영이 되고 있다.마케팅지원에 있어서는 전자신문,통신저널, TTA저널, IT Standard Weekly 등 각종 일/주/월간지 등에 정통부 품질인증획득 사실을 보도하고, SoftExpo에 부스를 마련하여 인증획득 제품을 전시홍보하며, 그 외에도 기타 국가 및 공공기관 우선 구매 지원 등을 추진중에 있다.

<표2> SW품질인증 서비스

품질특성	내 용 부 특 성
기능성 (Functionality)	일련의 기능 존재와 이들의 명세된 특성과 관련된 일련의 속성들의 집합 명시적 또는 묵시적 필요를 만족하는 것이 기능이다.
	적합성, , , , 준수성
신뢰성 (Reliability)	명시된 기간동안 명시된 조건에서 S/W 성능 수준을 유지하는 능력과 관련된 속성들의 집합
	성숙성, , , 수성
사용성 (Usability)	사용자( ) 사용을 위해 요구하는 노력과 그러한 사용에 대한 개개인의 판단과 관련된 속성들의 집합
	이해가능성, , , , 준수성
효율성 (Efficiency)	명시된 조건하에서 S/W 능 수준과 사용된 자원의 양 사이에 관계된 속성들의 집합
	시간효율성, , 준수성
유지보수성 (Maintainability)	규정된 수정을 수행하기 위하여 필요한 노력과 관련된 속성들의 집합
	분석성, , , , 준수성
이식성 (Portability)	S/W 다른 환경으로 이전되는 능력과 관련된 속성들의 집합
	적응성, , , , 준수성

⊙ 벤치마킹테스트 서비스

국내 중소 S/W개발 업체들은 우수한 기술과 제품을 보유하고 있음에도 불구하고 타 경쟁 제품에 비해 낮은 제품인지도, 경쟁제품과의 비교우위 분석자료 부족 및 제품의 시장규모에 대한 예측 부족 등으로 효과적인 시장개척을 하지 못하고 있다.따라서 정보통신부와TTA S/W시험인증센터에서는 소비자에게는 고품질의 우수제품 구매를 위한 정보 제공, 개발자에게는 경쟁제품과의 비교분석 자료 제공을 통한 제품 보완 및 경쟁력 확보전략 수립 기회 제공,그리고 시장 개척에 어려움을 겪고 있는 무명의 우수제품 발굴을 통한 마켓팅 지원 및 국내 S/W시장 활성화 등을 목적으로 벤치마킹테스트 서비스를 실시하게 되었다.

2002년 상반기에 BMT 평가모델 개발 및 테스트베드 구축, 그리고 선행시험 등을 실시한 후 하반기부터 본격적으로 BMT 서비스를 실시하고 있다.

⊙ 국제시험.인증 서비스

2001년 12월 미국 VeriTest와 TTA간에 기술 협력계약을 체결한 이후 VeriTest의 국제시험·인증서비스를 국내에서 받을 수 있도록 하고 있다. VeriTest는 1987년에 설립된 국제적인 제3자 시험·인증기관으로서 MS사의 Windows Logo Certification 독점 시험을 하고 있고, 시험 전문 인력만도 약 400여명이 넘는 미국 최대의 사설 시험기관 중의 하나이다.

국제 인증을 획득하게 되면 제품품질의 국제수준화를 통한 제품의 신뢰도 및 국제 경쟁력확보, 국제인증 획득 홍보를 통한 해외시장 진출 용이, 그리고 선진국의 비관세 무역장벽(품질인증 요구)등에 대응 가능한 이점이 있다.

또한 시험과정을 통한 선진 시험기술 습득과 VeriTest 현지에서 시험.인증 서비스를 받을 경우에 소요되는 막대한 비용과 시간도 절감하게 되는 효과가 있다.

⊙ S/W품질 컨설팅 서비스

S/W시험인증센터에서는 지금까지 수백개의 S/W제품을 접수 받아본 결과 거의 대부분의 제품에서 버그가 발견되는 등 수없이 많은 유형의 문제들이 내포되어 있다는 사실을 발견하고 이의 해결책으로 품질컨설팅 서비스를 제공하고 있다.

S/W 품질인증과 연계하여 시험(인증을 신청한 제품의 품질향상을 위한 상담 및 문제점 개선을 위한 기술 지도를 하게 된다. 내재된 결함 발견시 버그리포트 제공, S/W 패키징 기법, 제품설명서 및 사용자 매뉴얼 등

사용자 문서개발을 위한 문서작성 지침서(표준안)등의 제공을 통해 비숙련자라도 프로그램의 보완과 문서개발 등이 가능하도록 지원하고 있다.

⊙ S/W품질평가 기반기술 연구

현재 S/W시험인증센터에서는 패키지S/W 22개 분야 모바일 S/W, 임베디드 S/W, 컴포넌트S/W, e-Biz S/W, Web-based S/W,게임 S/W, GIS S/W, ERP, CRM, KMS, Groupware, 보안S/W, 주문형S/W, 행정업무용S/W 등 거의 모든 S/W분야를 대상으로 시험.인증 서비스를 제공하고 있다.

따라서 다양한 종류의 S/W제품을 시험하기 위해서는 각각의 S/W제품 특성에 맞는 평가모델이 필요하게 되므로 평가모델 개발 등 지속적인 기반기술 연구와 선행시험을 실시하고 있다.그 주요 내용을 살펴보면 Component,Mobile, Web-Based, e-Biz, Embedded S/W에 대한 평가기술, ebXML 표준적합성 및 상호 운용성 시험(인증 모델,생체인식, GIS 및 주문형 S/W의 품질평가 방법, 차세대 Web 시험기술 등을 연구하고 있으며, 이러한 분야에 대한 국제표준 및 해외기술 동향에 대한 조사도 병행하고 있다.

⊙ Open Lab. 구축 및 운영

자금, 기술, 시간, 인력 등의 부족으로 자체에서 시험을 진행할 수 없는 중소 S/W개발업체를 지원하기 위하여 Open Lab을 구축하여 서비스를 제공하고 있다. 대부분의 중소업체에서는 고가의 시험장비 및 시험자동화 도구, 시험전문기술의 부족 등으로 인해 제대로 된 시험을 할 수 없는 게 사실이다. 그로 인해 품질 완성도가 떨어지는 S/W를 시장에 출시하고 있고, 그 이후 제품의 수정단계에서 많은 유지보수 비용을 지불하고 있는 실정이다.이에S/W 시험인증센터에서는 고가의 시험장비와 시험 자동화 도

구, 각종OS와 시험 전문 인력 등을 갖춰놓고 테스트베드 환경구축 여력이 없는 중소벤처기업에서 언제든지 활용할 수 있도록 시험실을 Open해 놓고 있다.

⊙ 시험 전문 인력 양성

국내에는 S/W품질 또는 테스팅 전문 교육기관이 전무한 실정이다.

일부 대기업을 중심으로 품질보증 및 경영활동을 하고 있으나 국내에는 전문 교육훈련 과정이 없다 보니 비싼 비용에도 불구하고 매년 국외에 나가서 부분적으로나마 교육훈련을 고 오는 형편이다.

따라서 TTA S/W시험인증센터에서는 그 동안 축적된 기술의 산업계 전수를 통한 시험문 인력양성 사업을 추진하고 있으며, 상반기 중에 교재개발과 교육훈련프로그램 개발 등을 완료하고 금년 9-10월 경부터 교육시작을 목표로 하고 있다.

⊙ Testing Tool개발

시험의 정확성, 객관성, 효율성 등을 기하기 위해 시험 자동화 도구를 개발하고 있다. 주요 내용으로는 시험 결과 DB 분석 시스템, test coverage 분석 도구, 테스트 시나리오 작성 도구, 시험 DB & Reporting tool, SYS Analyzer, Stress Tester, 시험 리소스 관리 도구, 통합 시험 자동화 도구 등이 있다.

3) 향후 추진 계획

올해로4년째를 맡고 있고 국내 S/W 품질인증제도는 패키지S/W 시험·인증을 시작으로 전 S/W 분야로의 확대, 품질컨설팅, 국제시험·인증, 벤 치마킹테스트 등 현재는 S/W 품질향상을 위한 종합적인 서비스를 제

공하고 있다. 앞으로 이러한 서비스를 더욱 발전시키고 국내 S/W제품의 품질향상을 위해 우선은 시험범위 확대에 따른 평가모델 개발에 역점을 두고자 한다.

그리고 수년 내에 국산 S/W의 해외시장 진출지원을 위하여 국제 시험·인증제도를 더욱 발전시켜 VeriTest와 상호인증제도(MRA) 도입을 추진하고자 한다. 또한 어느 정도 여건이 성숙되면 독자브랜드(GS인증마크)로 해외시장에 진출하기 위하여 해외 시험소를 설치 운영하는 한편 국제적으로 GS마크의 공신력을 확보하여 국내에서 GS인증마크를 받은 제품이 국제시장에서도 품질보증이 통용되도록 할 계획이다.

나. 소프트웨어 벤치마크테스트(BMT) 현황[정통기b05]

1) 개 요

정보산업이 발달하여 유사한 제품의 개발이 많이 이루어짐에 따라 사용자는 업무의 목적, 용도 및 비용 등을 고려해서 최적의 제품을 선택하게 되었다. 이를 위해서는 유사 제품들 간의 기능이나 성능을 비교해서 보다 우위의 제품을 선택할 수 있는 정보가 제공되어야 한다.

또한 개발 업체들은 자신들의 제품이 타 제품보다 우수하다는 것을 증명할 수 있어야 했다. 이러한 필요성에 의해 벤치마크테스트(Benchmark Test)를 수행하기 시작하였다.

벤치마크테스트란 성능을 측정하기 위하여 사용되는 절차, 도구 또는 프로그램을 나타내며, 주로 비교 평가를 위한 기준으로 사용되고 있다. 시스템, 디스크, 메모리 등에 대한 성능, 각종 어플리케이션을 실행하는 컴퓨터의 성능, 파일 서버 또는 네트워크 상의 어플리케이션 서버의 성능 등 하드웨어 및 네트워크 제품에 대한 벤치마크테스트는 국·내외에서 비교적 많이 수행되고 있다.

CPU, 메모리, 그래픽 카드 등의 제품들은 사용자가 테스트 프로그램을 이용하여 쉽게 벤치마크테스트가 가능하며, 보다 복잡하거나 국제적인 경쟁력을 가지고자 하는 IT 제품은 미국의 NSTL, VeriTest, SPEC,TPC 등의 시험 전문기관을 이용하여 벤치마크테스트가 가능하다. 그러나 정확한 수치가 나오는 하드웨어 및 네트워크 제품과는 달리 소프트웨어는 객관적인 평가가 어려워 벤치마크테스트가 활발하게 진행되지 못하고 있다.

전 세계 패키지 S/W 수익의 90%를 차지하고 있는 미국시장에서만 S/W 벤치마크테스트가 활발하게 진행되고 있는 형편이다.

S/W 벤치마크테스트는 소비자에게는 우수 제품을 선택할 수 있는 비교 정보를 제공하고, S/W 개발 업체에게는 자사 제품의 강점을 파악하고, 취약점 보완 및 향후 개선 방안을 제시해 줌으로써 경쟁력을 제고하는데 크게 도움을 줄 수 있다. 특히 중소 S/W 개발 업체들이 우수한 기술과 제품을 보유하고 있음에도 불구하고 효과적인 매출 신장을 이루지 못하고 있는 국내 환경에서 벤치마크테스트를 통해 우수 S/W를 발굴하고 S/W 시장을 활성화 하는데 크게 효과가 있다. 그러나 현재 대부분의 국내 S/W 개발 업체들은 벤치마크테스트를 위한 투자 여력 및 기술력을 갖추고 있지 못하며, 국내에는 소프트웨어분야의 전문 벤치마크테스트 업체가 존재하지 않는 것이 현실이다. 이에 따라 한국정보통신기술협회(TTA)와 한국소프트웨어진흥원(KIPA)에서는 국내 S/W 벤치마크테스트를 활성화시키기 위하여 시장에서 임의 수거한 제품에 대해 벤치마크테스트를 수행하고 그 결과를 공표하는 프로세스를 마련하였다.

2) 국내 현황 및 문제점

⊙ 현황

국내에서는 초기에 소수의 S/W 개발 업체 및 대학교에서 적은 인력을 투입하여 S/W 벤치마크테스트를 수행하였으나, 현재 TTA S/W시험인증

센터를 제외하고는 모두 하드웨어 벤치마크테스트만 수행할 뿐 S/W 벤치마크테스트는 중단된 상태이다.

IT 관련 정보를 제공하는 벤처인 K-Bench(http://www.kbench.com)에서 2000년부터 S/W 리뷰를 시작하고, 2001년부터 2002년 3월까지 압축 유틸리티, 바이러스 백신, 동영상 재생 프로그램 등 3종에 대해 S/W 벤치마크테스트를 수행하였으나 현재는 S/W 리뷰만 제공하고 있다.

보물섬(http://www.bomul.com)에서는 2000년 10월부터 2002년 3월까지 전자신문 특집기사 게재를 위해 벤치마크테스트를 수행하였고, PC사랑 등의 잡지에서 요청이 올 경우 일부 수행하였다. 일반적으로 1명의 시험원이 벤치마크테스트를 수행하였고, 프로그램의 크기가 클 경우 최대 4명까지 벤치마크테스트를 수행하였으나, 벤치마크테스트를 수행하는데 대한 업무부담이 너무 과중하고 벤치마크테스트를 수행하던 주요 시험원의 이직이 많아지면서 벤치마크테스트는 중단된 상태이다.

PcBee(http://www.pcbee.co.kr)에서도 전자신문의 특집기사 게재를 위해 동영상 플레이어에 대한 벤치마크테스트를 1회 수행하였으나 현재는 하드웨어 벤치마크테스트만 수행하고 있다.

정보보호학회(http:..www.kiisc.or.kr)가 2002년9월에 일부 업체의 지원을 받아 7종의 백신 S/W에 대한 벤치마크테스트를 수행하였으나, 지원업체가 월등한 성적을 나타내는 등 공정성에 논란을 빚은 적이 있다.

한국정보통신기술협회(TTA, http://www.tta.or.kr) S/W 시험인증센터에서는 2002년부터 개발자 및 구매자로부터 의뢰를 받아 벤치마크테스트 결과를 당사자에게만 통보하는 형태로 벤치마크테스트를 수행하고 있다.

2002년에 모바일 S/W, 리눅스 OS, KT 소프트스위치 등 26개 제품,

2003년에 보안 S/W, Thin Client 제품, 자료관 시스템, 신전자문서 시스템, 백신 S/W, 운영체제 시스템 등 63개 제품, 2004년에 신전자문서 시스템과 자료관 시스템의 연동 API, 초고속 인터넷 복구 S/W, 리눅스 서버 운영체제 S/W 등 43제품 등 총 132 제품에 대하여 벤치마크테스트를 실시하였다.

⊙ 문제점

국내 시장의 경우 TTA S/W시험인증센터를 제외하고는 벤치마크테스트 전문기관도 부재하고, 벤치마크테스트 수요도 활발하지 못한 상황이다.

벤치마크테스트를 전문적으로 수행하기 위해서는 벤치마크테스트의 특성상 초기 투자비용이 많이 필요하다.

다양한 유형의 S/W에 대한 벤치마크테스트를 수행하기 위해 S/W 분야별 벤치마크테스트 항목 추출, 평가 기준 및 평가 방법을 개발해야 하고, 실제 벤치마크테스트를 수행할 전문인력이 필요하다. 그러나 대부분의 국내 S/W 관련 업체들이 벤치마크를 위한 투자 여력 및 기술력을 갖추지 못한 관계로 현재는 한국정보통신기술협회(TTA) S/W 시험인증센터를 제외하고는 S/W 벤치마크테스트 전문기관이 존재하지 않는다. S/W 벤치마크 테스트를 수행하였던 기관 중에서도 보물섬의 경우처럼 주요 시험원 한 명의 이직으로 S/W 벤치마크테스트 업무가 중단되는 등 전문인력의 부재로 지속적인 수행이 어려운 것이 현실이다. 또한 K-bench의 경우는 이윤을 추구하는 사업자 입장에서 수익 모델이 존재하지 않아 S/W 벤치마크테스트를 포기하고 있는 형편이다.

한편 S/W 개발관련 업체들은 독립된 제 3자 시험기관에서 S/W 벤치마크테스트를 수행하여 자사 제품의 취약점을 보완하고, 우수성을 부각시

킬 필요성은 인식하고 있으나 비용 등의 문제로 벤치마크테스트의 활성화가 이루어지지 못하고 있는 형편이다. 오히려 일부 회사에서 자체적으로 벤치마크테스트를 실시하여 자사 제품에 유리한 테스트 결과를 공표하여 소비자의 판단을 흐리게 하는 등 벤치마크테스트에 대한 부정적인 인식이 팽배해있는 상황이다.

비영리기관인 TTA S/W 시험인증센터에서가 지난 4년 동안의 S/W 시험·인증을 통해 축적된 시험·평가 기술과 경험을 바탕으로 객관적인 벤치마크테스트를 수행함으로써 그 동안 소수의 S/W 개발사들이 자사 제품 홍보를 목적으로 주관적인 벤치마크테스트를 실시하여 제기되었던 벤치마크테스트 무용론을 제거하는 효과를 가져 오고 있다

3) 벤치마크테스트 결과 공표

한국소프트웨어진흥원(KIPA)은 SW산업진흥법 제14조 및 동법 시행령 제 11조에 의해 정보관리전문기관으로 지정되면서, 우수 S/W를 발굴하고 국내 S/W 산업을 활성화시키기 위한 방안으로 소비자에게 우수 제품을 선택할 수 있는 비교 정보를 제공하고, S/W 개발 업체에게는 자사 제품의 품질향상 및 홍보 기회를 제공하는 S/W 벤치마크테스트를 활용하기로 하였다. 이에 따라 TTA S/W시험인증센터와 KIPA에서는 시장에서 임의 수거한 제품에 대해 S/W 벤치마크테스트를 수행하고, 그 결과를 공표하는 프로세스를 개발하였다. TTA S/W시험인증센터는 전반적인 벤치마크테스트를 수행하고, KIPA는 수행 의뢰 및 결과를 공표하는 역할을 분담하였다. 또한 S/W 벤치마크테스트가 공정하고 투명하게 수행되고, 객관성을 유지할 수 있도록 하기 위해 외부 전문가로 이루어진 BMT 협의회를 구성하였다.

⊙ BMT 협의회 구성

S/W 사용자들에게 제품별 비교 정보를 제공하는 S/W 벤치마크테스트 결과 공표로 인해 해당 업체들이 매출이나 제품 신뢰성 등에 막대한 영향을 받을 수 있으므로, 기술적 내용 검토 및 분쟁 대응을 위한 심의기구의 운영이 필요하여 BMT 협의회를 구성하게 되었다.

BMT 협의회는 S/W 벤치마크테스트 및 품질평가에 필요한 전문지식을 갖춘 전문가와 결과 공표에 대한 법적 대응을 위한 변호사 등 15인 이내의 민간 위원으로 구성되어 있다.

BMT 협의회의 주요 역할을 살펴보면 다음과 같다.

- S/W 벤치마크테스트 대상 제품 선정
  - 벤치마크테스트를 수행할 S/W 제품 선정
- S/W 벤치마크테스트 수행 검토
  - 벤치마크테스트 평가항목 및 시나리오 심의
- S/W 벤치마크테스트 결과 검증
  - 벤치마크테스트 과정 중에서 발생된 문제점 검증
  - 결과에 대한 해당 업체의 이의 제기 타당성 검토
  - 벤치마크테스트 결과에 대한 최종심사 및 결과 공표

⊙ 세부 수행 내용

벤치마크테스트를 수행하는 3개 주체인 BMT 협의회, TTA S/W시험인증센터 및 KIPA의 수행 내용은 다음과 같다.

<표3> 세부 수행 내용

수행주체	수행항목	각 항목 작업
BMT 협회	BMT 대상 제품 선정	▪ SW BMT 대상 후보 선정

		▪ SW BMT 제품 확정
	BMT 기본, 수행 계획서 심의	▪ 대상 SW 자료 분석 ▪ 도출한 평가항목 심의 ▪ BMT 시나리오 검토
	BMT 결과 검증 및 공표 여부 심의	▪ BMT 과정에서 발생된 문제점 및 결과공표 등 심의 ▪ 이의 제기 항목에 대한 재수행 계획 검토
	이의제기 검토	▪ 결과에 대한 해당업체의 이의제기의 타당성 검토
	분쟁시	▪ 법적대응 관련한 논리적 자료 마련
TTA	SW 분석	▪ SW 제품의 프로그램, 매뉴얼 등 관련자료 분석
	일정 및 자원투입 방안	▪ 인력 및 자원 투입 계획 ▪ 전문가 활용 검토
	시험환경 구축	▪ 시험환경 구축 ▪ 시험도구 및 SW 설치
	BMT 항목도출	▪ 해당업체의 의견수렴 ▪ BMT 항목 확정:주요항목, 부가항목
	테스트 방법 개발	▪ BMT 기법, 시나리오, 벤치마크 프로그램, 스크립트, 테스트 케이스 작성
	시험 및 검토	▪ 기능 및 성능시험 ▪ 결과 정리
	보고서 작성	▪ BMT 절차, 방법 및 비교분석 결과
	분쟁시	▪ 법적대응 관련한 SW BMT 수행 내역자료 마

		련
KIPA	BMT 수행 의뢰	▪ TTA에 SW BMT 수행 의뢰
	결과공표	▪ 고객대상 BMT 결과공표
	설문조사	▪ BMT 결과에 대한 설문조사
	모니터링	▪ 결과공표에 대한 지속적인 모니터링
	분쟁시	▪ 법적대응에 필요한 전반적인 자료 마련

⊙ 결과 공표 프로세스

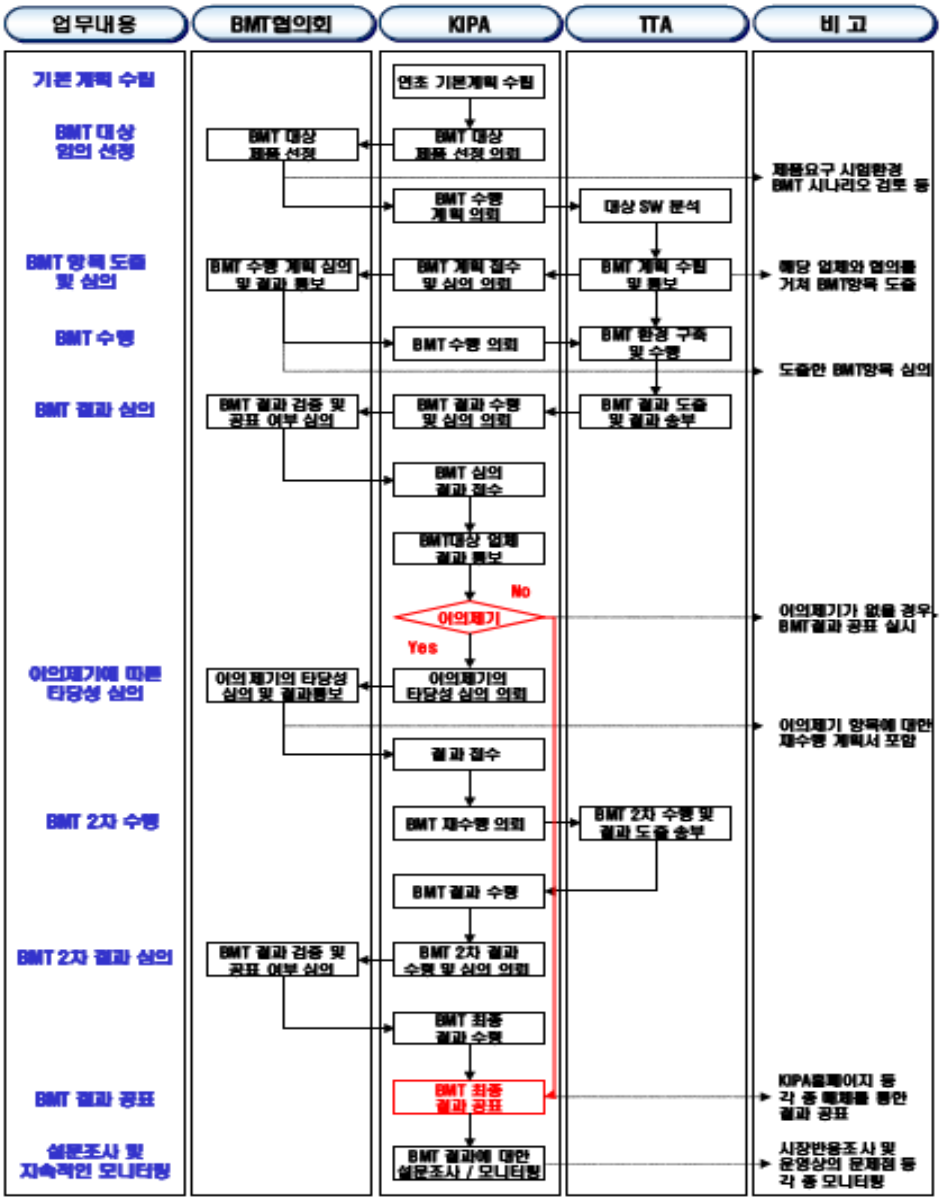
벤치마크테스트 결과 공표를 위하여 KIPA는 BMT협의회에 S/W 벤치마크테스트를 수행할 S/W 제품 선정을 의뢰하고, BMT 협의회는 분야별 S/W 대상군 및 S/W 대상 제품을 심의한다. TTA S/W시험인증센터는 선정된 S/W 제품을 임의 구매하여 각 S/W 벤치마크테스트 평가 항목에 따라 벤치마크테스트를 수행하여 결과를 도출한다. BMT 협의회는 한국정보통신기술협회(TTA)가 도출한 S/W 벤치마크테스트 결과를 심의한다. 심의를 통과한 벤치마크테스트 결과에 대해 KIPA는 해당 업체에게 송부한 후 해당업체의 이의제기가 없을 경우에는 한국소프트웨어진흥원(KIPA)이 홈페이지 및 각종 언론매체 등을 통해 결과를 공표함으로써 소비자에게 제품 비교 정보를 제공한다. 만약 해당 업체의 이의제기가 있을 경우, BMT 협의회의 심의를 통해 이의 제기가 타당성이 있다고 인정될 경우에는 벤치마크테스트를 재수행하게 된다. 재수행 하였을 경우에도 1차 테스트 결과와 오차 범위안의 결과가 발생하였을 경우에는 BMT 협의회에서 심의한 후 결과 공표를 추진한다.

4) 결론 및 향후 추진 방향

국내외 S/W 벤치마크테스트 현황과 결과 공표 프로세스에 대하여 살펴보았다. 2004년도에는 BMT 협의회 구성 및 전문가 활용을 통해 전문화된 체계를 구축하고, 리눅스 서버 운영체제 S/W에 대한 벤치마크테스트를 실시하여 2005년 2월 중에 결과를 공표할 예정이다.

향후에는 KIPA가 매년 초 S/W 벤치마크테스트 결과 공표 계획을 작성한 후 BMT 협의회 심의를 통해 연간 계획을 수립할 예정이다. 국가 전략 S/W 제품에 대한 벤치마크테스트를 중점적으로 실시하고, 언론매체를 통한 결과 공표를 원칙으로 한다. 시장에 출시된 국내외 S/W 제품들 중 시장 지배력이 있는 주요 제품을 임의 수거하여 실시하되, 임의 수거 방식을 통한 벤치마크테스트가 불가능한 제품에 대해서는 해당 기업의 협조를 요청하는 방안도 고려중이다.

S/W 벤치마크테스트 결과 공표에 대한 법적 분쟁을 사전에 예방하기 위해 해당 업체와 사전 협의를 도출할수 있도록 최선을 다할 것이며, 외산 S/W 업체의 이의 제기로 발생할 수 있는 통상 마찰도 대비하고 있다.



<그림3> S/W 벤치테스트 결과 공표 프로세스

다. SW제품의 품질정보 제공을 위한 BMT 시행[진홍원05]

일반적으로 구매자들이 SW제품 구매시 정보의 비대칭성으로 인하여, 제품의 품질보다는 브랜드 중심의 외산 SW제품을 막연히 선택하는 경향이 많다. 또한, 일부 회사의 경우 자체적으로 BMT를 실시하고, 자사 제품에 유리한 테스트 결과를 공표하여 소비자의 판단을 흐리게 하는 행태가 많은 것이 현실이다.

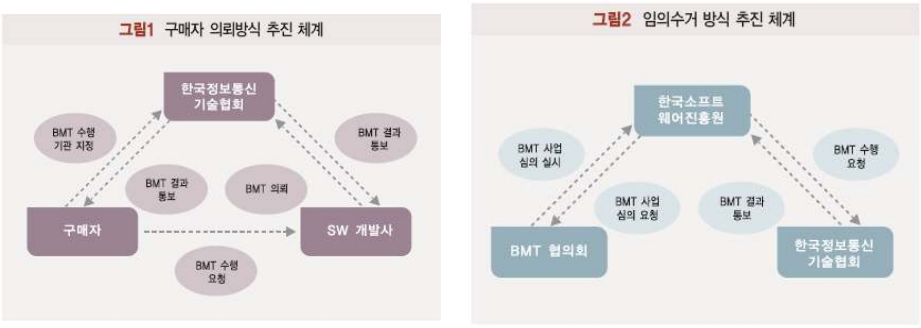
이러한 문제점에 대해서, 본원에서는 독립된 제3의 시험기관에서 수행하는 공신력 있는 테스트(BMT)를 실시함으로써 구매자를 대상으로 객관적인 SW제품의 품질 정보에 기반한 구매 의사결정을 할 수 있도록 지원하고, 또한 기업을 대상으로 자사의 제품품질 수준에 대한 정확한 정보를 제공함으로써 제품 품질수준 향상에 대한 계기를 제공하고 있다.

BMT 추진 방식은 구매자 의뢰에 의한 추진방식과 임의 수거에 의한 추진 방식으로 나눌 수 있다.

우선, 구매자 의뢰에 의한 BMT 수행 방식은 구매자의 의뢰에 따라 BMT를 수행하는 방법이며, 이 경우 의뢰자가 원하는 특정한 제품에 대해 테스트 후 그 결과를 구매자 및 개별 공급사에게 전달하는 방식이다. 이러한 방식은 주로 구매자가 신규 분야 SW제품에 대한 정확한 품질 정보를 알아야 할 필요가 있거나, 또는 제품 도입시 너무 많은 경쟁

SW제품들이 있어 각 SW제품의 품질정보를 정확히 알아야 할 필요성이 있는 경우에 주로 활용되고 있다.

구매자 의뢰방식은 구매자가 한국정보통신기술협회(TTA)를 BMT 수행기관으로 지정하고, 제품 공급사가 TTA에 BMT를 의뢰하여 TTA에서 제품을 테스트한 후 그 결과를 구매자 및 개별 제품 공급사에게 전달하게 되는 프로세스로 진행된다.



<그림4> 추진 체계

구매자 의뢰 방식 BMT를 활용하는 경우에는 중소기업은 60%, 대기업은 20%의 테스트 비용을 정부에서 지원받을 수 있다. 표1은 SW제품 규모별 평균지원 금액을 나타낸다.

다음으로, 임의수거 방식의 BMT는 BMT의 필요성이 높은 SW분야를 선정하여 해당 분야의 제품을 기업의 동의를 받아 제품을 제공받거나, 임의적으로 수거한 후 해당 SW 제품을 객관적인 평가지표에 의해 비교·평가하여 그 결과를 대외에 공표하는 방식이다.

BMT 운영 방식은 본원이 전반적으로 BMT 협의회(민간 교수 및 전문가로 구성)의 심의를 받아 사업을 추진하고 있으며 실제 BMT 테스트 수행은 한국정보통신기술협회(TTA)에서 전담하고 있다.

구매자 의뢰 방식의 BMT 활용은 2002년부터 다양한 구매자들의 요구에 부응하여 2005년 9월말 기준으로 총 19회(135개 제품)를 수행해오고 있다. 조달청의 Thin Client SW BMT 사례에 따르면, 조달청이 LG 전자를 포함한 10개 제품을 BMT 의뢰하여 수행한 결과, 젬텍이라는 무명의 중소기업의 제품이 가장 우수한 품질을 보유한 제품으로 판명되어 그 중소기업의 제품을 조달하기로 결정하였다. 이러한 사례에서 나타나듯이 구매

자들이 BMT를 활용하여 구매하고자 하는 제품의 정확한 품질정보를 확보하는 것이 구매결정에 중요함을 알려주고 있다.

<표5> 조달등록 BMT 평균 지원금액

SW제품규모	평균 지원금액(백만원)	
	60%(중소기업)	20%(대기업)
대(20백만원)	12	4
중(11백만원)	6.6	2.2
소(3.7백만원)	2.2	0.7

<표6> 2004년 임의 수거방식의 BMT 수행제품

구분	회사 및 제품명	커널	비고
1	리눅스 원의 NuxOne 2.0 Maru	2.6	국산
2	와우리눅스의 WOWLinux 7.3 Paran R2	2.4	국산
3	한컴 리눅스의 Hancom Linux Advanced Server 3.1	2.4	국산
4	노벨의 SUSE Linux Enterprise Server 9	2.6	미국
5	레드햇의 RedHat Enterprise Linux AS 3	2.4	이스라엘

또한 본원에서는 작년 처음으로 리눅스 서버 OS 제품군에 대하여 임의수거 방식의 BMT를 실시하였으며 금년에 그 결과를 한국소프트웨어진흥원 홈페이지(2005. 5. 18)에 공표하였다. BMT에 참여한 회사의 한 관계자에 의하면, 기존에는 자사 제품의 품질을 대외에 공식적으로 인정받을 수 있는 객관적인 자료가 없어 구매자를 대상으로 제품의 품질을 설명하는데 어려움을 겪었으나, 금번 BMT 수행 결과를 통해 구매자에게 제품의 품질에 대한 신뢰성을 높일 수 있는 계기가 되었다라는 평가를 내린 바 있다.

금년에도 4개 제품 분야(BPM, 리눅스 서버 OS, CAD 및 이미지 편집

SW 분야)에 대하여 임의수거 방식의 BMT를 추진하고 있으며, 금년 말에 한국소프트웨어진흥원 홈페이지(www.kipa.or.kr)에 BMT 결과를 공표할 예정이다.

라. BPM SW 벤치마크테스트[정통기06]

1) 서론

BPM은 엔터프라이즈 환경의 IT솔루션일 뿐 아니라 기업경영 개념을 구현하고 있다. CRM, SCM, ERP 등과 같은 이전의 다른 기업용 SW의 경우 기업 업무의 특정 프로세스를 지원하기 위해 SW가 특정 프로세스를 선구현하여 제공한 반면, BPM SW는 업무 프로세스 자체의 모델링과 관리가 목적인 SW이다.

BPM SW을 설치하였다 하더라도 운영하고자 하는 프로세스를 모델링하고 각 프로세스의 단위업무를 구현하지 않는다면 BPM SW의 운영을 가시적으로 확인하지 못하는 등의 특성으로 인해 BPM SW에 대한 품질 시험 항목은 일반적인 방법으로 정형화시키기 어렵다.

BPM에 대한 개요와 BPM SW BMT 시험 항목 및 시험 방법을 소개하고자 한다.

2) BPM 개요

BPM(Business Process Management)은 급변하는 경영 환경의 변화에 적응할 수 있는 비즈니스 프로세스 수행을 위한 해결책으로서, 최근 급격하게 개념이 확산되어 가고 있는 분야이다. BPM이란 조직 내부에서 또는 복수 조직간에 걸쳐서 일어나는, 사람 또는 시스템과 상호작용하는 비즈니스 프로세스를 식별하고 이해하고 관리하는 것을 말한다.

오늘날 업무 생산성 향상을 위해 새로운 IT 기술도입을 고려하고 있는 기

업들은 비즈니스 절차들을 긴밀히 결합해 업무 운영을 효율화하는 BPM 소프트웨어를 눈여겨보고 있다.

이들의 목표는 서로 다른 공간에서 운영되는 ERP(Enterprise Resource Planing)나 CRM(Customer Relationship Management), 그 밖에 다른 기업용 애플리케이션 등을 프로세스를 통해 긴밀하게 연계시키는데 있다. 기업의 요구에 의해 개발된 기존의 기업용 솔루션과 BPM의 가장 큰 차이점은 BPM의 관심대상이 프로세스이며 어떻게 프로세스를 관리할 수 있도록 지원하는가에 있다.

1990년대에 BPR(Business Process Reengineering)을 창시한 마이클해머는"프로세스가 곧 제품이다! 제품이나 서비스는 단지 프로세스의 부산물일 뿐이다"라는 말로써 프로세스의 가치를 강조했다. 그 이후 많은 IT 기술들이 경영환경에서 기업 내의 프로세스를 전산화하기 위한 도구로써 사용되어 왔다. 그룹웨어는 기업 내의 결재를 위한 시스템으로 개발되었고 CRM은 기업 내의 고객관리 업무를 위한 시스템으로 개발되었으며, SCM은 구매 관리 업무를 위한 시스템으로, KMS는 기업내 지식 관리를 위한 시스템으로 개발되었다. 이러한 기존의 시스템과 BPM SW의 가장 큰 차이는 프로세스의 관리를 위한 시스템이라는 것이다. 기존의 기업용 IT 솔루션이 대부분 미리 정해진 모범사례(Best Practices)를 기반으로 시스템에 프로세스를 구현하여 특정목적에 맞도록 개발되었다면, BPM은 미리 정해진 특정한 최적의 프로세스를 구현한 시스템이 아니라 각각의 기업이 원하는 목적에 따라 기업의 환경에 맞는 최적의 프로세스를 구현할 수 있도록 지원하는 솔루션이다.

업무의 전문가가 각자의 환경에 맞는 프로세스를 모델링 하고 그 위에 시스템을 구축하면, 프로세스는 가시적으로 관리된다. 이렇게 관리되는 프로세스는 급변하는 기업환경에 맞춰서 최소의 비용으로 가장 빠르게 환경에 적응하는 다른 프로세스로의 변화가 가능하게 된다. 또한 여타의 시스

템에 비해 프로그램을 수정하는 작업이 줄어들게 되어 유지보수 비용이 절감된다. 업무가 확장될 때 관련 애플리케이션의 프로그램을 수정하는 작업을 줄일 수 있다면 업무프로세스는 보다 효율적으로 관리될 수 있다.

최근 BPM 기술의 움직임은 프로세스 실행 기능과 프로세스 관리 기능을 연계하는 것이다. 프로세스 관리 기능은 기업들이 기업 내 운영 중인 프로세스를 자유롭게 모니터링하고 측정할 수 있도록 해 준다. 이러한 기능의 제공으로 BPM은 경영환경에서 RTE를 가능하게 할 수 있는 가장 주목 받는 도구가 되고 있다.

3) BPM SW BMT 시험 항목

2005년 8월부터 2006년 1월까지 6개월간 BPM BMT 참여사 4개 사와 501개 항목에 대해 협의를 하였다. BMT참여사의 요구 및 합의에 따라 성능 평가는 제외하고 기능평가만 하기로 합의되었으며, 합의된 결과에 의거 기능 중심의 평가항목만이 선정되었다.

501개 평가항목은 크게 8개의 평가항목으로 구분된다.

8개의 평가항목과 선정근거는 <표7>과 같다.

BPM SW의 운영은 비즈니스 프로세스 분석, 프로세스 설계, 프로세스 실행, 운영 중인 프로세스의 모니터링의 4단계로 구분했다.

전체 시험항목은 이러한 시스템 운영의 4단계를 기본으로 하여, 시스템이 지원하는 운영환경에 대한 확인 항목과 실제 BPM SW를 이용해 시스템을 구축할 때 필요한 개발 환경(EAI, 애플리케이션 개발환경) 항목, 그리고 제품의 표준 지원여부 확인 항목 등 총 8개의 분야로 구성된다.

<표7> 평가항목 구분 및 선정근거

일련번호	평가항목 구분	선정근거
1	운영환경	• 플랫폼 지원 여부, 다국어 지원 여부 등의 시스템 운영 환경에 대한 전반적인 기능 확인
2	비즈니스 프로세스 분석	• 프로세스 시뮬레이션, KPI 설정 등의 비즈니스 프로세스 분석을 위한 주요 기능 확인
3	프로세스 설계	• 플로우 설계, 프로세스 속성, 단위업무 속성 등의 프로세스 설계를 위한 주요 기능 확인
4	프로세스 실행	• 업무 회수, 반려 등의 프로세스 실행시 주요 기능 확인
5	비즈니스 프로세스 모니터링	• 실시간 모니터링, 대쉬보드 등의 프로세스 현황 파악 및 정보 분석을 위한 주요 기능 확인
6	EAI	• 레거시 어댑터 제공, 데이터 매핑 등의 시스템 통합과 관련된 주요 기능 확인
7	애플리케이션 개발 환경	• 사용자 화면 개발 지원, Rule 개발 지원 등의 BPMS 기반의 애플리케이션 개발을 위한 주요 기능 확인
8	표준지원	• BPEL, XPDL등의 BPM 관련 표준 준수 여부 확인

⊙ 운영환경

운영환경의 시험목적은 시스템 운영 환경과 클라이언트 운영환경, 국제화, 정보보안 등의 지원 여부를 확인하는데 있다. 각 평가항목별로 운영환경을 구성하여 제품기능의 정상동작 여부를 확인하는 방법으로 시험한다. 운영환경은 7개의 주요 평가항목과 41개의 평가항목으로 구성되며, 주요 평가항목과 평가내용은 <표8>와 같다.

<표8> 운영환경의 주요 평가항목 및 평가내용

평가항목 구분	일련번호	주요 평가항목	평가내용
운영환경	1	플랫폼 지원	• 다양한 시스템 운영환경 지원여부 확인
	2	사용자 환경	• 클라이언트 운영환경 지원여부 확인
	3	국제화	• 다국어 지원여부 확인
	4	신뢰성	• 시스템 신뢰성 확인
	5	접근제한	• 시스템 접근제한 기능 확인
	6	정보보안	• 사용자 정보보안 기능 확인
	7	달력지원	• 시스템 운영에 필요한 달력 제공여부 확인

⊙ 비즈니스 프로세스 분석

비즈니스 프로세스 분석의 시험목적은 프로세스 분석도구와 프로세스 정보 제공 및 조회 기능, 프로세스 시뮬레이션 등 프로세스 분석을 위한

기능을 확인하는데 있다. 각 평가항목별로 관련 메뉴(아이콘, 단축키)를 이용하여 기능 존재유무 확인 및 정상동작 여부를 확인하는 방법으로 시험한다.

비즈니스 프로세스 분석은 5개의 주요 평가항목과 64개의 평가항목으로 구성되며, 주요 평가항목과 평가내용은 <표9>과 같다.

<표9> 비즈니스 프로세스 분석의 주요 평가항목 및 평가내용

평가항목 구분	일련번호	주요 평가항목	평가내용
비즈니스 프로세스 분석	1	프로세스 분석 도구	• 프로세스 분석을 위한 기능 확인
	2	KPI 관리	• KPI 관리 기능 확인
	3	스코어 카드 관리	• 스코어 카드 관리기능 확인
	4	프로세스 조회	• 프로세스 정보검색 및 관리기능 확인
	5	프로세스 시뮬레이션	• 프로세스 시뮬레이션 기능 확인

⊙ 프로세스 설계

프로세스 설계의 시험목적은 프로세스 분류 기능, 프로세스 모델 관리 기능, 조직 및 권한 관리의 기능, 프로세스 속성 설정과 단위업무의 속성 설정 등 프로세스 설계의 주요 기능을 확인하는데 있다. 각 평가항목별로 관련 메뉴(아이콘, 단축키)를 이용하여 기능 존재 유무 확인 및 정상동작 여부를 확인하는 방법으로 시험한다. 프로세스 설계는 13개의 주요 평가항목과 183개 평가항목으로 구성되며, 주요 평가항목과 평가내용은 <표10>과 같다.

<표10> 프로세스 설계의 주요 평가항목 및 평가내용

평가항목 구분	일련번호	주요 평가항목	평가내용
프로세스 설계	1	프로세스 플로우 설계	• 프로세스 모델링 역량 확인
	2	프로세스 분류	• 프로세스 분류 관리기능 확인
	3	조직 및 권한 관리	• 조직 및 권한 관리기능 확인
	4	프로세스 속성	• 모델링시 프로세스 속성 설정기능 확인
	5	단위업무의 속성	• 모델링시 단위업무 속성 설정기능 확인
	6	비즈니스 속성	• 프로세스 진행시 운영환경 설정기능 확인
	7	Automation	• 참여자 설정기능 확인
	8	프로세스 라이브러리	• 모델링시 사용할 템플릿 제공여부 확인
	9	프로세스 모델 관리	• 작성된 프로세스 관리기능 확인
	10	모델러의 GUI	• 모델러의 유저 인터페이스 기능 확인
	11	편집 기능	• 모델러의 편집기능 확인
	12	외부 모델링 도구	• 외부 제품과 정보 공유기능 확인
	13	테스트 클라이언트	• 테스트 클라이언트 생성기능 확인

⊙ 프로세스 실행

프로세스 실행의 시험목적은 시스템 보안 기능, 프로세스 운영 플로우 관리, 사용자 환경기능 등 프로세스가 실행중에 필요한 기능을 확인하는데 있다. 각 평가항목별로 관련 메뉴(아이콘, 단축키)를 이용하여 기능존재 유무확인 및 정상동작 여부를 확인하는 방법으로 시험한다. 프로세스 실행은 8개의 주요 평가항목과 59개의 평가항목으로 구성되며, 주요 평가항목과 평가내용은 <표11>와 같다.

<표11> 프로세스 실행의 주요 평가항목 및 평가내용

평가항목 구분	일련번호	주요 평가항목	평가내용
프로세스 실행	1	운영	• 표준 프로세스 모델 실행여부 확인
	2	프로세스 실행 보안	• 프로세스 실행 권한 관리기능 확인
	3	신뢰성	• 운영 중 프로세스 정보 복원기능 확인
	4	동적 변경	• 실행 중 프로세스 변경기능 확인
	5	플로우 관리	• 실행 플로우 관리기능 확인
	6	프로세스 관리	• 프로세스 모델 업데이트 관리기능 확인
	7	시스템 관리	• 로그 관리, 프로세스 대버거 기능 확인
	8	업무 포털	• 사용자 참여 애플리케이션 기능 확인

⊙ 비즈니스 프로세스 모니터링

비즈니스 프로세스 모니터링의 시험목적은 리포팅 기능과 실시간 모니터링 기능, 대쉬보드 등 프로세스 실행 상태 정보를 모니터링 하기 위한 기능을 확인하는데 있다. 각 평가 항목별로 관련 메뉴(아이콘, 단축키)를 이용하여 기능 존재 유무 확인 및 정상동작 여부를 확인하는 방법으로 시험한다.

비즈니스 프로세스 모니터링은 5개의 주요 평가항목과 85개의 평가항목으로 구성되며, 주요 평가항목과 평가내용은 <표12>과 같다.

<표12> 비즈니스 프로세스 모니터링의 주요 평가항목 및 평가내용

평가항목 구분	일련번호	주요 평가항목	평가내용
비즈니스 프로세스 모니터링	1	데이터 관리	• 모니터링시 데이터 관리기능 확인
	2	모니터링 GUI	• 모니터링시 사용자 인터페이스 기능 확인
	3	실시간 모니터링	• 실시간 모니터링 기능 확인
	4	프로세스 분석	• 운영 프로세스 분석기능 확인
	5	대쉬보드	• 대쉬보드 제공여부 확인

⊙ EAI

EAI의 시험목적은 시스템 통합을 위한 EAI 기능을 확인하는데 있다. 각 평가항목별로 관련 메뉴(아이콘, 단축를 이용하여 기능 존재유무 확인 및 정상동작 여부를 확인하는 방법으로 시험한다.

EAI는 2개의 주요 평가항목과 38개의 평가항목으로 구성되며, 주요 평가항목과 평가내용은 <표13>과 같다.

<표13> EAI의 주요 평가항목 및 평가내용

평가항목 구분	일련번호	주요 평가항목	평가내용
EAI	1	UDDI 관리	• UDDI 관리기능 확인
	2	EAI 모듈	• 시스템 통합 기능 확인

⊙ 애플리케이션 개발 환경

애플리케이션 개발 환경의 시험목적은 Form과 Rule 그리고 웹서비스 등을 통해 BPMS 운영에 필요한 애플리케이션 개발 지원여부를 확인하는데 있다. 각 평가항목별로 관련 메뉴(아이콘, 단축키)를 이용하여 기능 존재유무 확인 및 정상동작 여부를 확인하는 방법으로 시험한다.

애플리케이션 개발 환경은 3개의 주요 평가항목과 19개의 평가항목으로 구성되며, 주요 평가항목과 평가내용은 <표14>과 같다.

<표14> 애플리케이션 개발 환경의 주요 평가항목 및 평가내용

평가항목 구분	일련번호	주요 평가항목	평가내용
애플리케이션 개발 환경	1	Form 저작도구	• Form 기반 애플리케이션 개발도구 기능 확인
	2	Rule 개발도구	• Rule 기반 애플리케이션 개발도구 기능 확인
	3	웹 서비스 지원 도구	• 웹서비스 지원도구 기능 확인

⊙ 표준 지원

표준 지원의 시험목적은 BPM 관련 표준 지원여부를 확인하는데 있다. 각 평가항목별로 관련 표준 명세서를 기준으로 표준 적합성 검사 도구를 이용해 표준 지원여부를 확인하는 방법으로 시험한다. 표준 지원은 12개의 평가항목으로 구성되며, 주요 평가항목과 평가내용은 <표15>와 같다.

<표15> 표준 지원의 주요 평가항목 및 평가내용

평가항목 구분	일련번호	주요 평가항목	평가내용
표준 지원	1	프로세스 정의 언어	• 프로세스 모델링 관련 표준 지원여부 확인

마. S/W BMT 기술동향 및 활성화 방안 [진홍원03]

1) 도입유형

공개소프트웨어를 도입하는 방법에는 크게 3가지 방법이 있다. 첫째 기관이 직접 선택하여 도입하는 직접선택, 둘째 민간기업 등 외부 의 추천에 의한 간접공급, 셋째 기관내부에서 공개소프트웨어를 개발 및 수정하는 내부개발 등이다.

이들 3가지 도입방법들에 대한 자세한 설명을 하기에 앞서 공개소프트웨어를 도입함으로써 얻을 수 있는 기대효과와 장점은 어떤 것들 이 있는가를 살펴보면 다음과 같다.

첫째, 소프트웨어의 유지관리의 지속성(영속성)을 보장 받는다. 원공급자가 폐업하여 사라지거나 지원을 중단하더라도 공개된 소스를 보유하고 있으므로 기관은 해당코드를 새로운 공급자에게 양도함 으로써 정보시스템 유지관리의 지속성(영속성)을 보장받는다.

둘째, 소프트웨어 도입 및 정보시스템 구축비용을 절감할 수 있다. 독점소프트웨어는 라이센스 수수료와 함께 서비스 및 기술지원 비용을 발생시키지만 공개소프트웨어는 라이센스 수수료에 대한 비용 대신에 서비스 및 기술지원비용만을 발생시킨다. 따라서 공개소프트웨어를 도입하면 도입비용을 현저하게 낮춘다. 또한, 유지보수 측면으로 보면 공개소프트웨어는 소스코드에 바로 접근하여 수정할 수 있는 권한이 확보되므로 유지보수 및 업그레이드에 투여되는 예산을 줄일 수 있다.

셋째, 높은 호환성을 보장받는다. 공개소프트웨어는 다양한 형태의 소스코드를 배포하고 있으므로 상대적으로 많은 플랫폼에 이식가능하며 또한 이와 같은 플랫폼 독립성 (호환성)은 수요자에게 하드웨어 선택의 폭을 넓혀준다.

넷째, 패치와 업그레이드 주기 및 속도가 빠르다. 공개소프트웨어는 소스코드가 공개되어 있고 많은 개발자들이 함께 공동 작업을 하고 있으므로 보안취약성등과 같은 결함을 발견한 후 몇 시간 또는 몇 일내에 재빨리 패치 및 업그레이드가 이루어지므로 보다 안전하게 시스템을 운영할 수 있다.

다섯째, 독점소프트웨어를 공급하는 업체로 부터의 구속을 탈피할 수 있다. 공개소프트웨어는 동일한 제품에 대해서도 공급자가 다수업체가 될 수 있기 때문에 공급자의 불합리성에 보다 적극적으로 대처할 수 있으며 또한 독점소프트웨어에 대한 구속을 회피할 수도 있다. 즉, 독점 소프트웨어의 경우에는 선택의 여지없이 사용해야 만하는 불합리성이 존재할 수 있지만 공개소프트웨어는 공급업체와 제품이 다수이기 때문에 이와 같은 구속은 존재하지 않는다.

여섯째, 소스코드의 수정 및 확장권한을 확보할 수 있다. 독점소프트웨어는 컴파일된 실행파일 즉, 이진파일형태로 제공되기 때문에 소스코드를 수정하거나 확장을 위한 변경작업을 할 수가 없으나, 공개소프트웨어는 소스코드가 공개되어 있기 때문에 현재 시스템에 적용되어 있는 소프트웨어의 수정과 추가 확장을 위한 변경작업이 얼마든지 가능하다.

일곱째, 공공부문의 정보시스템 통합이 용이하다. 정부 및 공공부문의 정보시스템 구축 시에 핵심모듈을 공개소프트웨어로 구축할 경우 공공부문의 개방표준화가 용이하게 되어 정보화 예산을 절감하고 호환성 및 이식성을 확보하여 공공부문 전체의 정보 시스템 통합을 촉진시킬 수 있다.

여덟째, 수요자 권리를 확보할 수 있다. 공개소프트웨어 도입 확대는 소프트웨어산업이 공급자 위주의 시장으로부터 소프트웨어 수요자(사용자, 정부) 위주의 시장으로 전환하는 데 중대한 역할을 할 수 있다. 독점소프트웨어 업체들이 그들의 소프트웨어를 공급하면서 독점지위를 유지하기

위하여 소스코드가 아닌 실행파일 형태로 제공하고 있다. 하지만 이것은 지극히 공급자 위주 의 관점이며 소프트웨어를 구매하는 수요자의 입장에서는 소프트웨어 구매와 함께 소스코드를 요구할 수도 있다고 볼 수 있다. 단순히 소스 코드만을 요구하는 것이 아니라 공급업체가 없어지거나 사라지더라도 도입한 시스템의 유지보수와 패치 및 업그레이드 등을 보장받으려 면 소스코드가 공개되어 있지 아니하고서는 불가능한 일이기 때문이다. 따라서 소비자들이 이러한 유지보수, 패치, 업그레이드 등을 보장 받으려면 독점소프트웨어로는 그 한계가 있기 때문에 공개소프트웨어를 사용하여 그 권한을 보장받을 수 있는 것이다.

공개소프트웨어는 수요자 측면에서의 이러한 장점과 함께 국가산업 적 측면에서는 소프트웨어 개발 원천기술을 확보함으로써 지식기반경제 핵심산업인 소프트웨어산업의 경쟁력을 강화시키고, 최신기술의 공개, 개발자 커뮤니티 등을 통한 기술교육에 의하여 국내 소프트웨어제품의 선진제품과의 기술격차를 줄일 수 있으며, 빠른 신제품 개발 등 시장에 즉시 대응이 용이하며 개발의욕 고취를 통한 양질의 핵심소프트웨어 인력양성이 가능하다는 장점이 있다. 직접선택, 간접공급 내부개발 도입방법 이외에도 다양한 방법들이 존재할 수 있지만 공개소프트웨어를 도입하는 거의 모든 경우는 대부분 이 세 가지 범주 내에 속하기 때문에 이번 절에서는 이들 3가지 방법에 대해서 설명한다.

⊙ 직접선택

직접선택("내부조달"이라고도 함)이란 공개소프트웨어 리소스 사이트에서 사용자가 특정 공개소프트웨어를 직접 다운로드하여 활용하는 것을 의미한다. 물론 다운로드 행위자체가 기관 내부의 직원에 의해 수행되며 기관 내부에서 공개소프트웨어 관련 정보가 어느 정도 확보 되어 있어야 가능한 방법이다. 이 방법은 비용발생이 거의 없다는 장점이 있는 반면 위

험부담이 크다는 단점이 있다. 특정 공개소프트웨어를 직접 다운로드 한다는 것은 독점소프트웨어를 공급업체로 부터 구매하는 것에 비한다면 기술검증에 대한 책임을 스스로 부담한다는 단점이 있다. 즉, 다운로드하는 소프트웨어 내에 트로이목마와 같은 바이러스 코드가 들어있다거나 키로거(key-logger)와 같은 악성 소프트웨어에 의한 감염을 포함한 소프트웨어가 될 수 있기 때문이다. 그러나 세계적으로 사용빈도가 높은 공개소프트웨어를 공식 사이트로부터 다운로드받는다면 이러한 부담을 제거할 수 있으며, 무결성 체크를 선행하는 경우 대부분 해결이 가능하다. 그리고 또 다른 문제점은 다운로드한 공개소프트웨어의 보상과 보증이 보장되지 않는다는 것이다. 즉, 공개소프트웨어 솔루션을 기관내부에서 직접 선택하여 도입하려고하는 경우에는 소프트웨어의 보증과 보상을 보장받지 못하므로 이에 대한 위험감소 대책을 고려해야만 한다.

이러한 위험에 대한 대책으로는 지역 내에 기술지원이 가능한 개발 업체가 하나이상 존재하는 공개소프트웨어를 선택하는 것이다. 이와 같은 대책을 마련해 둔다면 공개소프트웨어를 도입한 기관이 직접 해 결하지 못하는 기술적 문제에 대해 안정성을 보장 받을 수 있다. 즉, 도입기관의 시스템 운영자는 대부분 매일의 주기적이고 반복적 인 업무는 수행할 수 있다 하더라도 기술지원업체를 미리 확인하고 점검해 두는 것은 2중 3중의 안전장치를 가지는 것이며 시스템 가용성을 더욱 높이게 되어 위험요소를 감소시킬 수 있다. 직접선택 방법으로 공개소프트웨어를 도입할 경우 다음의 절차를 따를 것을 권고한다.

<표16> 공개소프트웨어 도입 절차

직접선택 워크플로우	
1 단계	공개소프트웨어 솔루션 조사[36]
2 단계	유력한 공개소프트웨어 솔루션 선택
3 단계	솔루션의 목적에 맞는 적합성과 비용적 가치 검토
4 단계	솔루션의 품질과 보안성 분석
5 단계	예비 프로젝트 수행
6 단계	예비 프로젝트 결과를 통하여 기본적 수준의 예측사항 검토
7 단계	프로젝트 수행 계획 수립
8 단계	프로젝트 수행

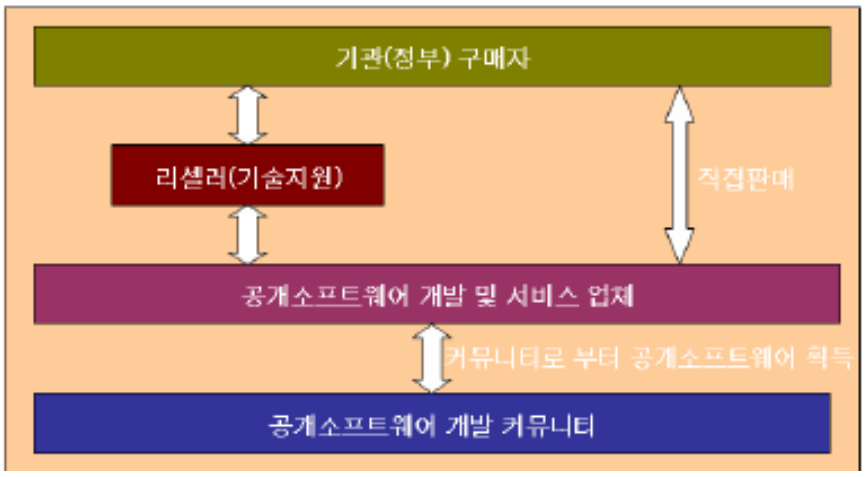
특정 공개소프트웨어 제품이 기관의 요구조건을 충족시킬 경우 도 입에 앞서 해당 기관에서 보다 상세한 검토를 수행하기로 결정하기로 하였다면 다음과 같은 체크리스트를 작성하고 평가과정을 거치도록 해야 한다.

<표17> 직접선택방식 도입 전 체크리스트

① 기관에서 사용되는 운영체제 플랫폼에서 해당 공개소프트웨어가 실행되는가?
② 해당 공개소프트웨어가 기존의 운영체제 플랫폼에 대해 획득, 시험, 배치해야 하는 추가적인 시스템 구성요소, 라이브러리 또는 모듈을 필요로 하는가?
③ 해당 공개소프트웨어의 설치 절차가 이해하기 쉽고 명확하게 정의되어 있는가?
④ 도입기관이 목적에 대한 적합성 여부를 결정할 수 있도록 공개소프트웨어를 설치, 배치, 시험할 수 있는 내부 전문지식을 갖추고 있는가?
⑤ 해당 공개소프트웨어의 설치 및 제거 절차가 명확하게 정의되어 있는가?

⊙ 간접공급

간접공급("외부조달"이라고도 함)이란 공개소프트웨어를 업체로 부 터 공급받는 것을 의미한다. 즉, 공개소프트웨어 공급 전문기업으로부터 특정 공개소프트웨어를 공급받는 것으로 기술지원과 서비스 비용 이 발생한다. 비용이 발생하는 대신 직접선택 방법에 비해 위험부담 을 줄일 수 있는 안전한 방법이라고 할 수 있다. 간접공급 방식으로 공개소프트웨어를 도입하였다면 라이센스에 관련된 법적인 위험요소에 대한 책임은 원칙적으로 공급업체에 있다. 하지만 이 경우에도 공급업체와의 계약서 내용에 따라서 법적인 책임 소재가 달라질 수 있으므로 이 부분에 대한 확인을 반드시 해야 한 다. 즉, 기관은 기술적인 위험요소와 변경관리 위험요소를 감소시키기 위한 적절한 실사조사를 수행해야 한다. 즉, 기관이 공개소프트웨어 솔루션을 위해 간접공급 방식을 선택한 경우, 해당 업체 즉, 공급업체 가 모든 기술 지원에 있어 책임을 져야하며 기관은 공급업체가 해당 공개소프트웨어에 대한 적절한 위험경감 절차를 수행함을 확인해야한 다는 의미이다. 다음은 간접공급 방식의 몇 가지 공급유형을 나타낸 것이다.



<그림5> 공개소프트웨어 간접공급 유형

이 외의 사항은 비공개소프트웨어를 도입할 때와 고려사항 및 절차 가 유사하다. 예를 들어 기관이 네트워크 및 시스템에 도입된 소프트웨어 보안표준에 대한 정책을 이미 보유하고 있을 경우에 개발업체는 위험분석의 일환으로 그러한 정책을 반드시 준수해야 한다. 또한 모든 절차가 끝난 후 검수작업 시에도 보안 표준과의 일치성 여부를 확인하도록 해야 하며 계약서 내에 이러한 모든 절차들이 포함되어 있어야한다. 또한, 기관은 개발업체에게 모든 소프트웨어 구성요소에 대한 공급 을 확인해야 하며 소프트웨어 출처와 라이센스 계약, 그리고 위험평가 문서를 확인해야 한다. 그리고 비공개소프트웨어를 도입할 때와 마찬가지로 도입된 공개소프트웨어의 전체적인 통합문제를 분석하고 이를 파악하여 반드시 문서화해야 하며 개발업체의 수행능력을 확인 할 수 있도록 자체평가를 수행할 수도 있다. 비공개소프트웨어를 업체로부터 도입할 때와 마찬가지로 간접공급 방법은 개발업체의 적격성, 확장성, 소프트웨어의 성숙도 등에 관련된 분석이 선행되어야 하며 이러한 문제는 소규모 개발업체와 거래할 때 특히 신중히 검토되어야 한다. 따라서 앞서 설명한 바와 같이 잠재적인 위험요소를 줄이기 위하여 개발업체에 대한 적절한 실사조사 및 평가는 반드시 이루어져야 하며 또한 평가 결과에 따른 조치가 이루어져야 한다. 다음은 이러한 평가에 반드시 포함되어야 할 내용들이다.

- 재정적 보장 및 안정성
- 위험관리의 적격성
- 내부 관리의 평가
- 업무 지속성 계획의 검토
- 일반적인 기능 개요
- 서비스 전달 및 관리
- 기타 기관의 특성에 맞는 평가요소

실제로 기관들은 공개소프트웨어 제품을 직접선택(내부조달)을 하기

위한 기술적인 능력을 갖추지 못한 경우가 대부분이므로 외부의 서비스 제공업체를 통해 공개소프트웨어 솔루션을 도입 받는 간접공급(외부조달)방법을 선택하는 것이 바람직하다. 그리고 잘 알려져 있고 인기 있는 대부분의 공개소프트웨어 제품들은 상용 솔루션 제공업체들을 통해 공급이 가능하다. 하지만, 외부의 어떤 업체를 통해 공급받더라도 공개소프트웨어와 비공개 소프트웨어 (독점소프트웨어)의 공급에는 차이점이 있다. 즉, 기관이 외부 서비스 업체를 통해 공개소프트웨어 솔루션 또는 제품을 공급받는 경우 일반적으로 관련 공개소프트웨어는 무료로 제공받고 이에 대한 서비스를 구입한다. 이것은 라이센스를 구입하고 부가가치 서비스로 지원을 제공받는 비공개 소프트웨어(사유소프트웨어)의 경우와 차이가 있다. 그리고 기관이 외부 공급업체를 선택하기에 앞서 후보 업체들에 대 한 실사로서 재정 상황, 안정성, 기술 능력 등을 평가해야 한다. 이러한 평가 작업들은 커뮤니티의 지원이 사라질 수 있는 위험한 제품선택에 대하여 위험을 줄여준다. 그리고 도입 후 어느 시점에 외부 공급업체가 사라지더라도 다른 공급업체에게서 지원을 계속 받을 수 있는 안정성과 유연성을 제공해 준다.

⊙ 내부개발

내부개발이란 기관내부에서 직접 개발하는 방법을 의미하며 공개되어 있는 수많은 공개소프트웨어들 가운데 도입하고자 하는 용도에 가장 적합한 공개소프트웨어를 선택하고 다운로드하여 기관내부의 개발자들에 의해 수정 개발하는 것을 의미한다. 아무것도 없는 상태에서 처음부터 개발하는 것이 아니라 공개되어 있는 수많은 공개소프트웨어들 가운데 도입하고자 하는 용도에 가장 적합한 공개소프트웨어를 선택하고 다운로드하여 기관내부의 개발자들에 의해 수정 개발하는 것을 의미하는 것이다. 기관내부 개발 방법으로 공개소프트웨어를 도입할 때에 고려해야 할 사항들을 간단히 정리해 보면 다음과 같다.

- 가장 적합한 공개소프트웨어 선택기준 마련
- 다운로드한 공개소프트웨어의 안정성 검토
- 적용되는 라이센스 정책마련
- 가장 적합한 개발방법론 마련
- 개발 직원의 개발능력 검토
- 사용직원의 사용법 교육 안 마련
- 개발소프트웨어의 문서화 작업
- 개발된 공개소프트웨어의 안정성과 영속성 대책 마련
- 기타 특정업무의 종속되는 대책 마련 등

다음은 실제 내부 개발을 추진한 경우 필요한시에 단계별 개발절차 이다.

- 서비스의 정의 및 적용업무 분석 가장 먼저 해야 할 작업이 어떤 업무에 적용하기 위하여 공개소프트웨어를 도입할 것인가를 정의해야하는 작업이다. 즉, 이러한 작업을 "서비스 정의"작업이라고 할 수 있으며 웹 기반으로 개발할 것인지 아니면 기관내부 솔루션으로 사용하기위한 개발인지 아니면 유관기관과의 정보공유 및 협동 네트워크를 위한 개발인지를 정확하게 구분하고 이를 정의해야 한다.

- 기반 기술 정의 다음 단계는 개발기술에 대한 분석과 정의이며, 공개소프트웨어의 개발에 필요한 기반기술들을 간단히 정리하면 다음과 같다.
  - 운영체제, 웹서버, DBMS, 사용 언어 등에 대한 기반기술 분석 및 정의
  - 공용 framework 개발 및 업무 서비스 요구 분석 및 설계 기술
  - 분산 DB 구축 설계 및 관리 기술
  - 그룹웨어의 확장 및 공유 수준 지정에 의한 access 범위 설정 기술
  - 유관 기관 간 정보 DB 표준화 (XML 기반)

이와 같이 관련 기반기술들이 정의가 되어 있어야 하며 개발 직원에 대한 교육과도 연계되어야 할 항목이다.

- 적용 업무에 대한 분석 개발하려는 공개소프트웨어가 적용될 업무에 대한 기본적인 분석이 이루어 져야한다. 즉, 공개소프트웨어를 직접 개발하기 이전에 다음과 같은 사항들이 이미 검토되어야 한다.
  - 활용성 : 개발 이후의 당기관 또는 유관기관에서의 활용성
  - 수요의 규모 : 개발한 공개소프트웨어의 수요정도에 대한 검토
  - 재활용성 : 개발한 공개소프트웨어가 타 업무에 재활용 가능 여부
  - 표준성 및 표준의 준수여부

- 개발기간 및 투입인력 분석 공개소프트웨어를 직접 개발하기 위해서는 개발기간과 투입인력에 대한 분석과 정의가 이루어져야 한다. 개발기간에 대한 정의는 다음 을 고려하여 작성하도록 한다.
  - 5단계 개발기간 분석 : 요구분석기간, 분석기간, 개발기간, 수정 및 검토기간, 적용 및 안정화기간에 대한 분석 및 각 기간별 위험관리 대책 마련(개발기간의 준수여부에 따라 기간초과에 따르는 위험관리)
  - 개발인력에 대한 대책
    * 기반 기술(주로 Language)에 가장 적합한 기술인력 활용
    * 개발경험이 있는 내부 개발 직원 활용
    * 개발책임자(PM)을 선정하여 프로젝트로 진행
    * 개발 분야에 따라 외부 개발 직원을 투입대책 마련
    * 개발자 교육 대책 마련

- 개발한 공개소프트웨어의 사후 관리문제 내부개발에 의해 개발 완료된

공개소프트웨어는 일회성 프로젝트로 끝나지 않도록 사후관리가 이루어져야 한다. 사후관리에는 개발 직원에 대한 사후관리를 포함한다. 사후관리는 개발한 공개소프트웨어를 관리대장에 등록하여 지속적인 관리가 이루어지도록 한고, 개발 시에 사용한 기술을 종합 정리하여 이를 문서화하고, 개발한 공개소프트웨어의 설치문서, 사용법문 서, 업그레이드 문서 등을 표준화하여 작성한다. 지금까지 설명한 내부개발 방법을 설명하였다. 기관 내부에 공개소프트웨어 개발자를 확보하기 어려울 뿐만 아니라 프로젝트 진행을 위한 관리자 확보가 어렵기 때문에 현실적으로 내부개발 방법으로 개발 하는 경우는 거의 없다. 개발자나 프로젝트 실무자를 확보하고 있다 하더라도 실제 진행을 위해서는 보직변경 및 장기간 동일업무 종사여건등과 같은 조직내부적인 환경이 갖추어져야 할 것이다. 결론적으로 내부개발의 방법으로 프로젝트를 진행하기란 현실적인 어려움들이 존재하는 것이 사실이고 이러한 현실적인 어려움을 극복하고 프로젝트를 진행한다 하더라도 개발 및 프로젝트 노하우의 축적이 지속되기란 더욱 어려운 실정이다.

⊙ 종합비교

지금까지 설명한 직접선택, 간접공급, 내부개발 3가지 방법에 대한 장단점을 종합비교하면 다음 표와 같다. 공공기관에서 3가지 방법 중 하나를 선택하여 공개소프트웨어를 도입할 수는 있지만 국내 공개소프트웨어 업계의 현실과 기관의 현실을 고려할 때에 가장 현실성이고, 일반적으로 사용할 수 있는 방법은 간접공급 방식이다.

2) 도입절차

정부기관 및 공공부문에서 시스템을 도입하거나 소프트웨어를 도입 하는 일반적인 절차는 다음 표와 같다. 각 단계에서 공개소프트웨어 도입을 위한 주요 검토사항은 다음과 같다.

<표18> 공개 SW 도입을 위한 주요 검토 사항

단 계 (절 차)	내 용
사업계획서 작성	ㅇ 과제 발굴(대부분 사업 전년도), 예산편성 ㅇ 사업 타당성, 적정성 검토 ㅇ 사업 추진계획서 작성(최종 사업계획서 작성)
제안요청서 작성	ㅇ 제안요청서 작성 (객관성 및 타당성 검토가 절실히 요구되는 단계)
사업공고 및 제안서 접수	ㅇ 입찰공고 ㅇ 제안설명회 ㅇ 사업제안서 접수
제안서평가 및 선정	ㅇ 제안서평가(평가위원회) ㅇ 원가분석 ㅇ 기술협상 및 가격협상
계약체결	ㅇ 최종계약체결
사업수행	ㅇ 사업수행(계약체결일로 부터 사업 진행) ㅇ 진도보고(착수, 주간, 중간보고 등) ㅇ 완료보고
결과보고	ㅇ 최종 결과보고서 평가(검수)

⊙ 사업계획서 작성 단계 추진 과제를 발굴하여 사업 타당성과 적정성을 검토한 후 사업계획 서를 작성한다. 주관기관은 사업계획 수립 시 비공개소프트웨어와 함께 공개소프트웨어 도입을 고려하되, 시스템 개발 시 개방 표준(Open Standard)을 지원하고 개방형 플랫폼(Open Platform)과 상호 호 환성을 가지는 제품을 우선 고려해야 한다.

⊙ 제안요청서 작성 단계 사업자 선정을 위한 제안요청서(RFP, Request For Proposal)를 작 성한다. 제안요청서는 사업 프로젝트 수행에 있어서 매우 중요한 역 할을 하는 만큼 제안요청서 작성 시 비표준 조건과 특정기술 등의 제 약을 두지 않도록 매우 주의해야하는 단계이다. 즉, 고의, 부주의 또 는 업무관습 등으로 인하여 주관기관에서 제안요청서 작성 시 운영체제와 하드웨어 플랫폼

등과 같이 매우 중요한 부분의 요구사항에서 특정 기업 제품 또는 기술표준만 제안 가능한 기술요건을 명시할 경우 불공정경쟁을 초래할 수 있으므로, 다음 표를 참고하여 비 표준적이거나 특정 기술조건을 명시하지 않도록 주의하여야 한다.

<표19> 비표준·특정기술 조건 예

구 분	요구조건	특정기술조건 예
공통	특정업체 특정제품만 가능한 사양 한정	. PCI Slot 48개 . 특정 locking 명시
하드웨어 (Server)	특정 운영체제 아키텍처기반의 CPU를 명시	. RISC, SPARC
	특정 제품만 가능한 성능, 품질등의 인증요구	. 특정업체 인증 tpmC요청
주변기기	특정 운영체제만 지원하는 주변기기 채택	. Windows XP 전용 스캐너
운영체제(OS)	특정 운영체제만 지원하는 소프트웨어 개발언어 및 기술사용	. ActiveX, ASP, WMV로 개발
	특정 운영체제를 명시	. UNIX 플랫폼지원 . Windows 플랫폼지원
WAS	특정 개발 툴에 종속된 개발환경 요구	. 특정 세션관리 기법제시
웹서버 (WebServer)	특정 운영체제상에서만 운용이 가능한 제품 채택	. IIS 도입
	특정 웹어플리케이션 개발 언어의 사용요구	. ASP 사용 개발
DBMS	특정 운영체제상에서만 운영이 가능한 제품 채택	. 특정 Locking 기법 제시
어플리케이션	특정업체의 기술 종속적이며 사용자의 선택권을 제한하는 소프트웨어 지정	. Windows Media Player

그리고 제안요청서 작성 시에 기관은 시스템 계층별 독립성을 확보하기 위하여 제안요청서상의 '도입대상 장비 내역 및 구성요건' 작성 시 다음 사항을 요구할 필요가 있다. 첫째, 특정 하드웨어에 종속된 특정 운영체

제를 선택할 수밖에 없는 상황을 초래하지 않도록 하기 위하여 하드웨어와 운영체제를 별도의 항목으로 명시하고, 별도의 비용으로 계상한다. 둘째, 특정 운영체제에 종속된 특정 응용 소프트웨어를 선택해야만 하는 상황을 초래하지 않도록 하기 위하여 응용 소프트웨어와 운영체제를 별도의 항목으로 명시하고, 별도의비용으로 계상한다. 그리고 포털사이트 구축 사업 또는 웹 기반의 서비스 구축 사업을 진행할 경우 주관 기관은 제안요청서 작성 시 국민의 "정보 접근권 보장"을 위하여 "다양한 컴퓨팅 환경에서 접근 가능하도록 국제표준 을 준수하는 제품도입 및 개발"에 대한 항목을 명시하도록 한다. 그리고 BPR/ISP 사업의 제안요청서 작성 시에는 공개소프트웨어 적용 가능성 분석을 포함시킴으로써 사업 초기단계부터 공개소프트웨어기반의 시스템 구축 가능성을 검토함으로써 시스템의 상호 운용성을 확보할 필요가 있다. 마지막으로 공개소프트웨어를 제안하는 사업자는 사업완료 이후 '공개소프트웨어 유지보수 방안 제시'에 대한 항목을 반드시 명시하도록 함으로써 공개소프트웨어에 대한 기술지원 및 유지보수 서비스를 보장 받아야 한다.

⊙ 제안서(제안업체) 평가 및 선정 단계 일반적으로 사용하는 제안서 평가항목은 다음과 같다.

- 유사분야에서의 사업수행 경험
- 개발대상 업무의 이해도
- 개발전략
- 기능 및 성능
- 개발방법론
- 기타 주관기관의 평가항목

각 항목별 세부 평가요소들은 주관기관의 사업추진 방향에 따라서 다소 달라질 수 있다. 특히, 소프트웨어 평가 시 동등 성능일 경우 공개소프트웨어를 제안한 업체를 우선 고려할 경우 정보화예산을 절

감 할 수 있는 장점이 있다.

⊙ 사업 수행 및 결과보고 단계 공개소프트웨어를 적용하여 시스템을 구축하였을 경우 제공하는 소프트웨어의 소스코드를 확보하여야 한다. 특히, 제안업체에서 소스코드를 일부 수정하였을 경우에는 수정부분에 대한 문서화와 필요한 경우 해당 소프트웨어 개발 커뮤니티에 정보를 제공함으로써 공개소프트웨어 발전에 기여할 필요가 있다.

3) 적용시스템

전자정부 등 공공 정보시스템 구현시 목표로 하는 시스템을 서비스 측면으로 보면 크게 정책결정 시스템, 거래 처리 시스템, 지식정보 시스템 등 세 가지 유형으로 구분할 수 있다. 다음에서는 정보시스템 구축 유형별 고려사항과 공개소프트웨어 기반의 정보시스템 구축시 필요한 시나리오별 구현방법을 소개한다.

⊙ 시스템 유형

▪ 정책결정 시스템

정책 결정 시스템은 어떤 하나의 안건에 대해 다수의 의견을 모아 이를 정책에 반영하고 스로인해 원만하고 활발한 정책 적용 및 이에 영향을 받는 구성원들의 참여를 이끌어 내는 것을 목표로 한다.

따라서 정책 결정 시스템에서 제공하는 서비스는 구성원들의 참여를 이끌어내기 위한 것과 공정하게 의견을 반영하는 것, 그리고 결정된 안건이 구성원들의 의견을 받아들여 투명하게 수행되고 있다는 것을 보여 주는 데에 목적을 둔다.

따라서 아래와 같은 세부 서비스를 정의할 수 있다.

첫째, 구성원들의 참여도 향상 서비스로써 구성원들의 참여도를 증가시키기 위해서는 언제 어디서나 구성원들이 자신의 의사를 전자정부 시스템에 반영할 수 있는 인터페이스가 갖추어져야 한다. 전자정부에서는 웹 기반의 의견 수렴 도구를 구축하여 사용자들의 접근이 쉽고 수많은 접속자들로부터의 의견을 반영할 수 있도록 공개된 웹서버를 통한 서비스를 제공해야 한다. 세부적인 서비스로는 선거나 토론을 위한 온라인 선거 서비스, 온라인 토론방외에도 각종 건의사항 을 빠르게 반영할 수 있도록 온라인 건의사항 수렴서비스 등이 존재 할 수 있다.

둘째, 의견 반영 서비스로써 구성원들의 참여도가 향상되어 인터넷 을 통한 구성원들의 의견을 수렴하면 이를 실제 정책에 적용하고 평가하는 과정에 대해 구성원들에게 신뢰감을 심어줄 수 있는 서비스가 제공되어야 한다. 이러한 서비스의 세부 내역으로는 정책을 적용하고 이를 수행하는 관리자나 혹은 관련 단체에서 보다 빠르게 의견을 모니터링 할 수 있는 서비스와 필요 없는 정보를 줄일 수 있는 필터링 서비스, 의견 분석을 위한 자료수집과 수집된 자료를 일련의 지표로 삼을 수 있는 자동 데이터 분석 서비스가 있다.

셋째, 정책 집행 서비스로써 사용자들의 요구에 부합하고자 정책이 집행되는 모든 과정을 투명하게 사용자에게 제공할 수 있고 신속한 처리과정을 위해 관련 부서간의 유기적인 협력 체제 및 인터페이스를 갖추어야 한다.

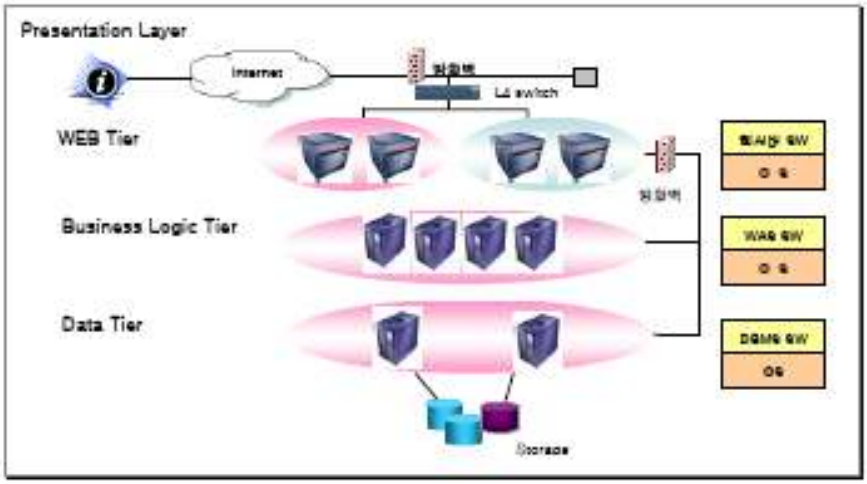
- 거래처리 시스템

거래 처리의 관점에서 보면 전자정부는 네트워크를 통하여 G2G, G2B, G2C, C2G, B2B, B2C 등의 거래를 제공할 수 있어야 하며 이를 위해서

범용적이고 향상된 보안대책이 필요하다.

▪ 지식 정보 시스템

공공부문과 민간부문에 대한 서비스를 위해 다양한 데이터가 사용된다. 이 데이터들은 종류도 다양할뿐더러 그 양이 너무 방대해 파일관리 위주의 일반적인 데이터 관리로써는 원할한 정보의 공유가 힘들 다. 더구나 효율성 면에 대한 고려도 필요하기 때문에 이 데이터를 이용한 지식 정보 베이스를 구축해야 할 필요성이 있다. 나. 시나리오별 도입 방법 웹 기반의 업무환경은 Web Tier, Business Logic Tier, Data Tier 의 3계층으로 구성되며 각 각의 계층은 웹브라우저/웹서버, 웹 어플리케이션 서버, 데이터베이스 서버의 순서로 처리된다.



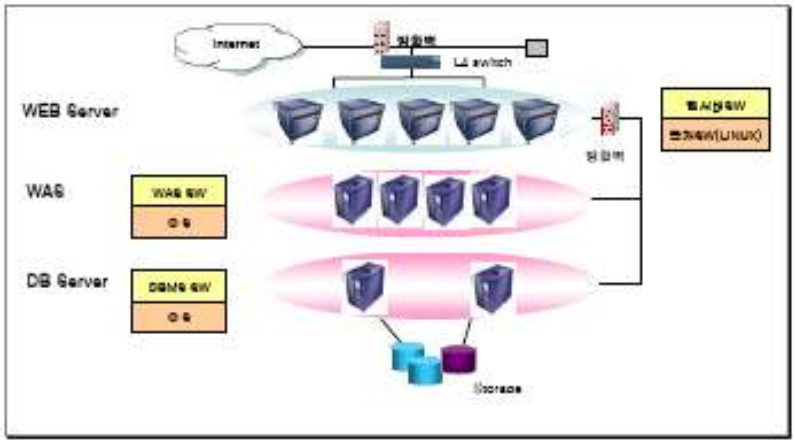
<그림6> 웹기반 정보시스템 구조

웹 기반의 전자정부 등 공공 정보시스템을 공개소프트웨어기반으로 도입시 다양한 시나리오를 적용할 수 있는데, 그 가운데 대표적인 세 가지 시나리오별 예시하면 다음과 같다.

▪ 시나리오 1 :

웹 서버 운영체제 공개소프트웨어 도입 기존 3계층으로 구성된 환경에서 웹 서버 운영체제를 공개SW로 대 체하는 경우로써, 공개SW의 안정성 및 성능에 대해 PoC(Proof of Concept)를 실시하고자 하는 경우 적합하다.

웹 서버에 공개소프트웨어 를 도입하는 경우는 단순히 서버만을 교체하는 수준의 작업이 요구되므로 전환이 용이하다는 장점이 있다. 웹서버 소프트웨어는 공개소프트웨어(Apache, Tomcat 등)와 비공개소프트웨어(WebtoB, SunOne 등)와 모두 적용할 수 있다.



<그림7>웹서버 운영체제 공개SW 도입

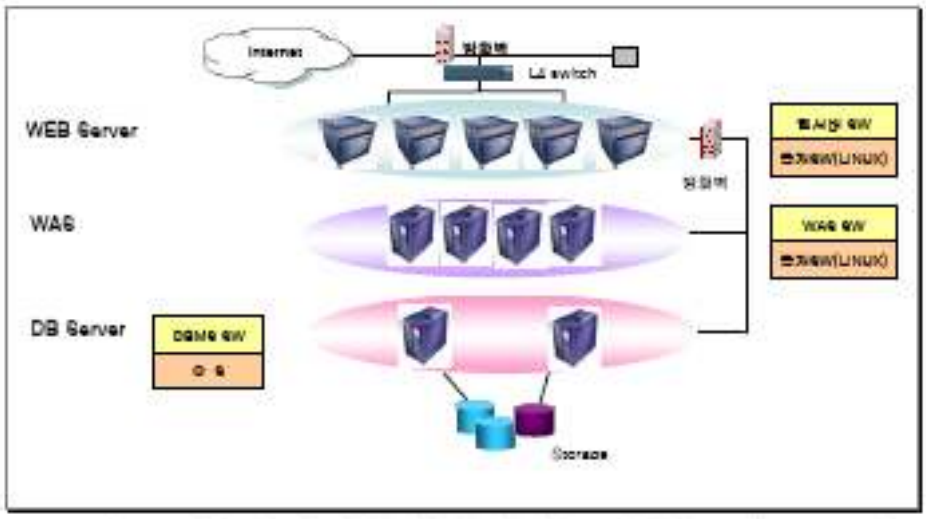
▪ 시나리오 2 :

웹서버 및 웹 어플리케이션 서버 운영체제 공개 소프트웨어 도입 기존의 3계층으로 구성된 환경에서 웹서버, 웹 어플리케이션 서버 운영 체제를 공개SW로 도입하는 경우로써, 비용절감 효과를 원하며, 비교적 안전하게

공개SW를 도입하고자 하는 경우에 적합하다.

어플리케이션 서버의 경우 J2EE 표준준수로 비교적 추가 작업이 적고, 비용절감효과 크다.

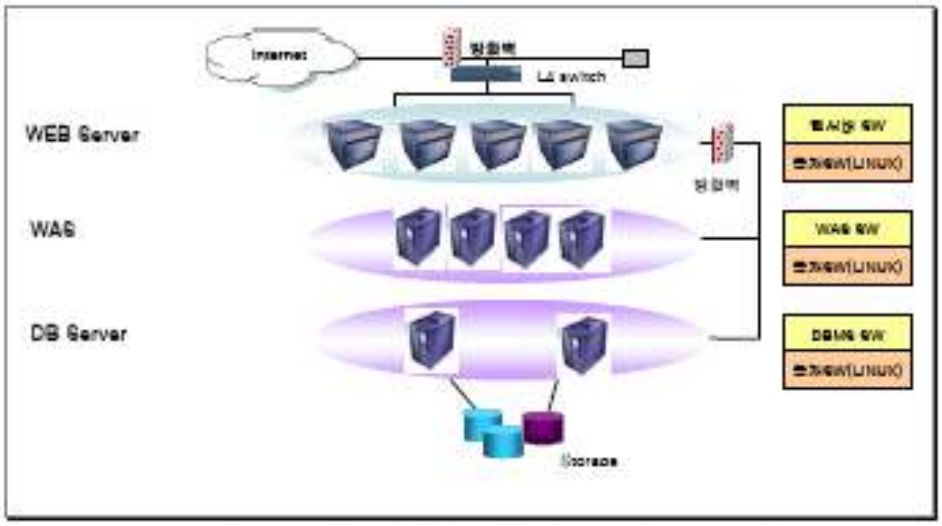
WAS는 공개소프트웨어(JBoss, Tomcat, JOnAS 등) 또는 비공개 소프트웨어 (JEUS, Weblogic, WebShpere 등) 모두 적용 가능하다.



<그림8> 웹서버, 웹 어플리케이션 서버 운영체제 공개SW 도입

- 시나리오3 :

적극적 공개SW 도입 기존의 3계층으로 구성된 환경 전체 운영체제에 공개 SW를 도입하는 경우로써, 신규로 구축되는 사이트에 가장 적합하고, 비용절감 효과가 가장 크다. DBMS는 공개소프트웨어(MySQL, PostgreSQL 등) 또는 비공개소프트웨어(UniSQL, Oracle 등) 모두 적용 가능하다.



<그림9> 3계층 서버 운영체제 공개SW 도입

⊙ 적용 가능분야

대표적인 공개소프트웨어인 리눅스는 웹서버, 메일서버 등 인프라서버와 1way ~ 4way 엔트리급 서버에서는 이미 기술적 안정성을 입증 받고 있으며, 리눅스 커널 2.6 출시 이후 DB서버, AP서버 등 기관 핵심 업무와 8 way 이상의 엔터프라이즈급 서버 운영체제로 폭넓게 활용되고 있는 추세이며, 특히 리눅스가 일반적으로 선택되고 있는 환경은 다음과 같다.

- 네트워크 인프라 : 도메인 네임 서버 (DNS), IP 주소 할당 (DHCP), 웹 서비스, 어플리케이션 서비스, 프록시 서버, 디렉토리 패킷 쉐이핑 및 통신 최적화를 위한 소프트웨어 포함
- 데이터베이스 서버 : 탁월한 오픈 소스 데이터베이스 서버로는 MySQL, PostgreSQL, MaxDB Firebird SQL(이전에는 Interbase), Ingres(이전에는 Adabas)가 있으며, 다수의 비공개 데이터베이스 서버도 리눅스에서 이용 가능
- 보안 시스템: 방화벽, 침입 탐지 및 분석, 허니포트(honeypots), IPSEC 및 기타 가상 사설망(VPN) 시스템, 패킷스니핑 (packet-sniffing) 및

분석, 바이러스 백신 소프트웨어 및 스팸 방지 필터링 포함

- 인터넷 및 인트라넷 출판: 웹 서버, 컨텐츠 관리 시스템(CMS) 플랫폼 및 워크플로우 관리 도구 포함
- 문서 관리: 자동 전자 문서 캡처 시스템, 개정 관리 시스템, 데이터 캡처 기술, 문서보관 시스템 포함
- 이메일 및 통신: 이메일, 일반 그룹웨어(그룹 캘린더 관리, 주소 록 공유, 알리미 서비스, 공유 폴더) 및 인스턴트 메시지 교환 서버를 위한 다수의 솔루션 포함
- 어플리케이션 서버: Java, PHP, Perl, Python 및 ZOPE 스크립트 툴, (Java 및) Java 2 Enterprise Edition(J2EE) 서버(예: JBoss), dotGNU .NET 공개소프트웨어 어플리케이션 서버에 기반한 광범위하게 사용되고 있는 웹 어플리케이션 서버 포함. 또한, 현재 다수의 비공개 어플리케이션 서버가 리눅스에서 이용 가능
- 파일 및 프린트 서버: Unix NFS, Microsoft SMB/CIFS 및 Novell Netware NCP와 같은 대부분의 주요 파일 공유 프로토콜을 포함하는 툴
- 저장: 몇몇 네트워크에 부착된 저장 장치들은 주로 공개소프트웨어 플랫폼에 구축되어 있음
- 제한된 기능의 워크스테이션: 기본 웹, 이메일, 터미널 액세스 그리고 통화 센터, 키오스크 및 이와 유사한 용도를 위한 사무 실 생산성 관련 기능을 제공하는 고정 용도의 워크스테이션
- 고성능 컴퓨팅: 여기에는 다수의 마이크로프로세서 (수직 평가), 다수의 저비용 시스템(수평 평가) 및 기타 유형의 수퍼컴퓨터에 기반한 클러스터를 장착한 단일 영상 시스템 포함
- 고성능 기술 워크스테이션: 과학적 분석, 계량, 모델링, 3D 컴퓨터로 생성된 이미지 (CGI) 및 비디오 처리 기능과 같은 컴퓨터 의 처리 기능을 강화한 어플리케이션용 멀티 프로세서, 64-비트 및 대용량 메모리 시스템

4) 고려사항

공개소프트웨어를 도입 방법, 절차 그리고 적용 가능한 시스템에 대하여 살펴보았으며, 공개소프트웨어를 도입할 경우 다양한 요구사항들이 도출되는 것을 확인했다. 따라서 이번 절에서는 공개소프트웨어를 도입함에 있어서 고려해야 할 사항들을 검토하면 다음과 같다.

첫째, 도입하고자 하는 공개소프트웨어에 대한 라이센스를 정확하게 확인하고 이해해야 한다. 공개소프트웨어로서 완전한 권리를 획득하기로 하였다면 해당 공개 소프트웨어는 소스코드를 모두 공개하고 해당권리를 사용자에게 양도해야 한다. 하지만 대형 프로젝트 등에서는 공급자가 모든 소스에 대 한 권리가 없는 경우가 많으므로 소스의 일부는 비공개 소프트웨어거나 타인의 저작권에 속하는 기술로 개발된 부분일 수 있으므로, 이러 한 부문의 검토도 간과하지 말아야 한다.

둘째, 시스템 호환성 확보를 위해 개방표준을 지원하는 제품을 우 선적으로 고려한다. 정보시스템을 설계할 때에는 유연한 상호호환성 확보를 위해 개방 표준을 지원하는 제품을 우선적으로 고려해야 한다. 모든 사람들이 읽어야 하는 자료를 웹을 통하여 제공할 경우에는 가능한 모든 정보 통신 환경 사용자가 접근 가능하도록 제공하여야 한다.

셋째, 도입하는 공개소프트웨어에 대한 기술지원 및 유지보수에 대 하여 고려해야 한다. 만약 간접공급 방식으로 공개소프트웨어를 도입하기로 결정하였다면 사용자지침서, 온라인 튜토리얼, 교육훈련, 헬프 데스크 및 유지 보수가 원활해야하며 또한 여러 플랫폼이나 사용 환경에 맞추어 맞추어 버전관리가 잘 이루어질 수 있는지 확인해야 한다. 직접선택 방식으로 공개소프트웨어를 도입한다면 위 사항에 대한 구체적인 대안이 마련되어야 한다.

넷째, 사업의 취지에 가장 적합한 솔루션인가를 깊이 있게 고려해 보아야한다. 제도적 또는 절차적 공정성을 확보하여 독점소프트웨어와 공개소

프트웨어의 장단점을 고려하여 가장 적합한 솔루션을 선택해야 한다. 독점 소프트웨어라고 하여 소프트웨어의 완성도가 반드시 높은 것은 아니며 공개소프트웨어가 완성도가 높은 경우도 흔히 있다. 따라서 사업의 취지에 맞는 소프트웨어의 장단점을 검토하여 가장 적합한 솔루션을 선택해야 한다.

다섯째, 도입되는 솔루션에 대한 합리성을 확보해야 한다. 소프트웨어 구매 시 같은 성능일 경우에는 가격이 낮은 제품을 선택하고, 같은 가격일 경우에는 소스코드의 확보 및 확보된 소스코드 의 자유로운 변경이 가능한 제품을 우선적으로 고려해야 한다. 성능과 가격이 모두 같은 경우라면 소스코드의 확보가 가능한 것이 향후 프로그램의 유지보수에 대한 보장성과 사업의 영속성을 위하여 보다 합리적이므로 소스코드의 확보가 가능한 것을 선택하도록 한다. 단, 소스코드의 확보는 독점소프트웨어이든 공개소프트웨어이든 구분 하지 않고 확보가능성 여부를 고려해야 한다. 여기서 주의할 것은 소스코드의 확보가 가능하다고 하더라도 이것의 자유로운 변경이 허락 되어 있지 않은 경우에 추가비용이 필요할 수 있으므로 이러한 경우 에도 소스코드의 자유로운 변경이 가능한 제품을 선택해야 한다. 여기서 주의할 것은 소스코드의 확보가 가능하다고 하더라도 한 번 수정이 가해지면 원래의 공개소프트웨어 공급처(또는 커뮤니티)의 소프트웨어와 호환성이 떨어져 향후 업그레이드가 어려워지므로 소스코드의 수정작업은 신중히 이루어져야 한다.

여섯째, 도입되는 공개소프트웨어의 소스코드에 대한 완전한 권리를 획득하였는가를 확인해야 한다. 도입하는 공개소프트웨어에 대한 소스코드의 확보는 매우 민감한 문제이다. 소스코드를 확보하지 못한다면 향후 수정, 패치, 업데이트 등과 같은 작업뿐만 아니라 시스템 증설작업이 필요할 경우에 사업자 의 폐업과 같은 극한 경우에도 사업의 영속성을 보장받을 수 있기 때 문이다. 따라서 공개소프트웨어 도입 시에는 소스코드의 확보와

함께 확보한 소스코드의 변경 가능여부와 변경한 소스코드를 적용가능한가를 반드시 확인해야 한다.

일곱째, 비공개소프트웨어를 선택하였을 경우와 마찬가지로 선택한 소프트웨어에 대한 기술적인 분석이 이루어져야 한다. 공개소프트웨어를 선택하든 독점소프트웨어를 선택하든 도입하기로 선택한 소프트웨어에 대해서는 기능성, 성능, 효율성, 그리고 확장가능성 및 시스템의 운용에 따르는 부가적인 문제로 구분하여 분석해야 한다.

여덟째, 도입하는 공개소프트웨어에 대한 법적인 분석이 이루어져야 한다. 어떤 무형의 권리를 "지적재산권"으로 총칭하여 언급하는 것이 편리하기는 하지만, 세부적인 내용을 살펴보면 이것은 많은 관점에서 서로 다른 법률들을 포함하는 집합적인 개념(collective term)이다. 따라서 실제로 무엇이 쓰여 지거나, 창조되거나, 개발되었는지, 그리고 어떤 종류의 지적재산권법이 관련되는지를 개별적으로 파악할 필요가 있다.

마지막으로, 웹 클라이언트 측면에서의 접근성에 대한 준수여부를 고려해야한다. 현재의 웹 기반 개발 환경이 MS Internet Explorer 를 기준으로 개발되고 있어 리눅스, 맥킨토시 등 비 윈도즈 운영체제 및 모질라, 사파리 등 비 IE 환경 사용자들의 인터넷 사용에 제약을 가하고 있다. 따라서, 웹을 통한 정보공개, 민원서비스 등의 시스템을 구축할 경 우 다음의 기준을 준수하는 것이 필요하다.

- HTML 문서 작성 시 표준 태그만을 이용하여 작성
- 자바스크립트 작성 시 표준을 따르는 코드만 이용
- 사용자 인증 등을 위하여 Plug-in 기술을 사용해야 하는 경우 다양한 운영체제 및 웹브라우저에서 접근 가능하도록 설계
- 기타 특정기업에 의존성 있는 코드 사용 배제

위에 언급된 대표적인 브라우저(MS IE, 모질라, 오페라, 넷스케이프)에서 동일하게 접근 가능한가 테스트를 실시(W3C의 Validation

Test(validator.w3c.org) 참조)

5) 공개 소프트웨어 도입 가이드라인

⊙ 정보시스템 구축을 위한IT솔루션을 구매할 경우 절차적 공정성을 확보하여 독점소프트웨어와 공개소프트웨어의 장단점을 고려하여 가장 적합한 솔루션을 선택하여야 한다.

※ 절차적 공정성 확보란 특정 운영체제에서 구현 가능한 기술이나 표준을 강제하지 않는 것을 의미한다.

- 독점소프트웨어라고 하여 소프트웨어의 완성도가 높은 것은 아니며 공개소프트웨어가 완성도가 높은 경우 또한 많다. 따라 서, 소프트웨어의 장단점을 검토하여 적합한 솔루션을 선택하도록 해야 한다. 예를 들어, 현재 Sendmail은 가장 보편적으로 사용되는 소프트웨어로 소스를 공개하고 있는 공개 소프트웨어이다. 이 소프트웨어는 오랜 기간동안 검증된 것으로 그 탁월한 기능성을 인정받고 있다.
- 특정 운영체제에서 구현 가능한 기술이나 표준을 강제할 경우 기타 운영체제 소프트웨어의 선택 범위를 줄이고, 같은 운영체제 내에서도 특정 소프트웨어의 특정한 기술이나 표준을 강제 하는 경우에는 해당 운영체제 내에서 소프트웨어에 따라 동종 의 서비스를 구현하지 못할 수도 있어 특정 소프트웨어를 사 용하도록 유도할 수 있으므로 소프트웨어의 독점을 초래할 수 있다.

⊙ 소프트웨어 구매시 동등 성능일 경우 낮은 가격, 동등 가격일 경우 소스코드의 확보 및 확보된 소스코드의 자유로운 변경이 가능한 제품을 우선적으로 고려하여야 한다.

- 소프트웨어가 동등 성능일 경우 낮은 가격을 선택하는 것이 비 용 효율성면에서 합리적이다. 동등 성능일 경우 굳이 높은 가격의 소프트웨어를 구매해야할 이유가 없다. 더욱이 공개 소프트웨어이고 상대적으로 저가의 소프트웨어라고 해서 이 소프트웨어에 대한 서비스가 현저하게 떨어지지는 않는다. 오히려 공개 소프트웨어에 대한 공개 포럼이 독점

소프트웨어보다 활성화되어 있기 때문에 특정 기업의 서비스 외에도 문제 해결 방 법 면에서 가능성이 더 다양하다. 특히, 본사가 어디에 위치하고 있느냐에 따라서 서비스의 질이나 시간적인 차이가 날 수밖에 없는 상황이라면 벤더 이미지에 비해 상대적으로 낮은 서비스와 책임회피 등의 부작용이 있을 수 있다.

- 동등 성능이면서 동등 가격일 경우 소스코드의 확보가 가능한 것이 향후 프로그램의 유지 보수를 위해 합리적이므로 소스코드의 확보가 가능한 것을 선택하도록 한다. 소스코드의 확보는 독점 소프트웨어이든 공개 소프트웨어이든 구분하지 않고 확 보 가능성 여부를 고려해야 한다.
- 소스코드의 확보가 가능하다고 하더라도 이것의 자유로운 변경 이 허락되어 있지 않은 경우에 추가 비용이 필요할 수 있으므로, 이러한 경우에도 소스코드의 자유로운 변경이 가능한 제품 을 선택하도록 한다.
- 만약 성능과 가격의 차이가 각각 소규모일 경우에는 소프트웨어의 비용대비 성능과 소스코드의 확보 여부를 고려하여 향후 소프트웨어의 도입시에 보다 비용 효율적이고 합리적인 선택을 하도록 한다.

⊙ 정보시스템을 설계할 때, 유연한 상호호환성 확보를 위해 개방표준을 지원하는 제품을 우선적으로 고려하여야 한다.

※ 개방표준이란 다음과 같은 컴퓨터 자료의 암호화 및 전송에 대한 규격을 의미한다.

- 모든 사람이 읽고 구현할 수 있는 자료
- 특정한 공급자 또는 그룹에 사용자를 구속하지 않는 자료
- 적합성 인증을 위해 표준화 기관에서 요구하는 수수료를 제외하고 로열티 또는 기타 비용을 지불하지 않고 모든 사람이 자유롭게 구현할 수 있는 자료
- 구현 기술 표준 적합성 외에는 어떤 사유로도 특정 구현자에게 특권을 부여하지 않는 자료
- 표준 위반을 금지하는 라이센스 규정을 사용할 수 있는 자료

- 모든 사람이 읽고 구현할 수 있는 자료의 경우에는 특정인이나 특정 기업의 영향력으로부터 자유로울 수 있으므로, 관리와 개발 모든 측면에서 합리적이다.
- 사용자가 특정한 공급자 또는 그룹에 구속된다면 이에 따른 추가 적인 부담이나 비용 발생시 유연한 대처가 어려워질 수 있다.
- 소프트웨어가 상호호환성을 지니지 아니한 경우에는 향후 다른 시스템으로의 전환이 어려워지고 많은 비용이 추가로 필요할 수 있으므로 이에 따라 특정 벤더에 종속될 수 있는 가능성을 가지게 되므로 상호호환성을 확보할 수 있도록 한다.
- 개방표준을 지원하지 않는 경우 처음 개발자 및 소프트웨어 제공자의 개발 지속 여부에 상당한 특권을 부여하게 될 우려가 있다. 그러므로 이를 방지하고 해당 소프트웨어의 안정적인 개발을 위해서는 개방표준을 지원하는 소프트웨어를 채택하도 록 해야 한다.

⊙ 주문형 소프트웨어 구매 또는 정보화를 위한 시스템통합(SI)사 업 발주처는 소프트웨어에 대하여 소스코드의 확보 등 완전한 권리를 획득하여야 한다.

▪ 시스템 도입에 있어 수요자의 요구에 맞추어 개발된 제품의 소유권이 확보가 되어야 한다. 상용 제품을 활용할 수 있지만,상용 제품을 활용하여도 소프트웨어에 대한 완전한 소유권 및 관련 지적재산권을 인정받아 소스코드의 사용을 확보하여야 한다.

▪ 시스템통합(SI)사업의 발주처는 향후의 안정된 시스템 관리 및 유지보수를 위해 소프트웨어에 대한 완전한 권리를 획득하고 있어야 한다. 이것은 소스코드의 확보가 포함된 것으로 소스코드의 확보는 향후의 시스템의 유지"E보수를 위해 필요한 것이므로 이에 대한 신중한 고려를 해야 한다.

- 소스코드의 확보가 되지 않은 경우 소프트웨어에 대한 유지보수시 추가적인 비용을 부담하거나 관련 신소프트웨어를 기 타 소프트웨어의 선택의 여지없이 구매해야 하는 경우가 발생 할 수 있다.
- 소프트웨어의 구매 또는 시스템의 개발이 새롭게 필요한 경우 특정 시스템에 종속되지 않고 확장이 가능하여야 하며 타 시스템과의 연동이 원활하게 되어야 한다. 필요시 시스템의 연동 과 확장을 위하여 소프트웨어의 수정이 가능하도록 소스코드를 공개하여야 한다.
- 현재 세계적인 검색 사이트인 Google은 FreeBSD 운영체제를 사용하고 있으며, FreeBSD는 소스가 공개되어 있어 자유롭게 이 소스들을 해당 기업 시스템에 맞게 수정하여 최상의 서비스가 이루어지도록 하고 있다. 이는 Google의 자체 기술력에 의해 이루어지고 있는 것으로 소스를 공개한 소프트웨어를 사 용한 성공적인 공개소프트웨어의 활용사례이다.

⊙ 정부 및 공공기관이 인터넷을 통하여 정보 제공시 가능한 모든 환경 사용자가 접근 가능하도록 제공하여야 한다.

- 기관이 보유한 정보 및 서비스를 인터넷으로 제공하고자 웹시스템을 구축할 경우, 서비스 수혜자가 컴퓨팅 환경(Windows, Linux, Mac 등)에 관계없이 이용 가능하도록 설계"E구축해야 한다. 예를 들어, 웹프로그래밍에 있어서 웹브라우저의 특정 기능에 종속시키지 않도록 하거나 웹브라우저별로 웹페이지를 따로 구축하도록 하고, 특정 운영체제에서만 사용이 가능한 웹 브라우저 플러그인 소프트웨어를 사용하지 않도록 한다. 예를 들어, Internet Explore와 Netscape에서 지원하는 웹프로그래밍 문법에 차이가 있고, ActiveX 플러그인은 특정 운영체제에 서만 제대로 사용된다. 그리고 웹브라우저별로 웹페이지를 따 로 구축하는 경우에도 각 웹페이지의 기능상 차이가 없어야 한다.
- 정부 및 공공기관의 정보제공은 일반인들에게 일반적인 정보들과는 달리 정보에 대한 접근의 선택이 불가능한 경우가 많으므로 정보제공 시스템에의 접속 가능성이 일반인들의 컴퓨터의 소프트웨어에 영향을 줄 수 있다. 따라서 이러한 경우에는 일반인들에게 불필요한 시간의 낭비

나 새로운 소프트웨어의 구입 을 위한 불필요한 비용을 발생케 할 수 있다. 예를 들어, 특정 운영체제에서만 접근이 가능한 웹서비스의 경우 사용자들은 해 당 운영체제를 구입해서 설치하고 사용해야 하므로 동류의 소프트웨어에 대한 독점적 지위를 부여하거나 사용자들에게 추가적인 비용을 부담하게 하는 결과를 초래하게 된다.

- 시스템에 연결된 클라이언트 소프트웨어의 사용의 제한은 클라이언트 시스템을 특정 벤더에 종속시켜 국가 전반적으로 소프트웨어 시장의 불균형을 초래할 수 있다. 뿐만 아니라, 인터넷 을 통한 정보제공의 사용자 접근의 제한은 결국 다시 정부 및 공공기관의 데스크탑 환경을 제한할 것이고, 이것이 전체 정부 및 공공기관의 시스템을 특정 벤더로 종속시키는 것을 유도할 수도 있으므로, 이에 대한 심각한 검토를 해야 한다.
- 모든 환경 사용자가 접근가능하도록 제공되지 않을 경우, 해당 정보를 얻거나 특정인의 편의를 위해 특정 정보를 제공하기 위해서 소프트웨어를 구입해야 할 수 있다. 그러나, 이 경우 해 당 소프트웨어를 구입할 수 없는 사용자들의 접근이 원천적으로 불가능하게 될 수 있으므로, 정부의 행정에 대한 헌법상의 국민의 알권리나 엑세스권의 보장에 문제가 발생할 수 있다. 바. 독점 소프트웨어를 선정하였을 경우에는 정당함을 설명할 수 있어야 한다.
- 기본적인 타당성은 기능성, 성능, 효율성과 함께 차후 확장 및 시스템의 운용에 따르는 부가적인 문제로 구분한다. 그리고, 차후 확장에 있어 공개된 소스코드의 수정이 가능한지를 검토한다.
- 소프트웨어 선정시 선정에 대한 선택 기준이 명확해야 한다. 일반적인 명성이나 편견에 따라 소프트웨어를 선택하는 것은 비용의 낭비와 관리상의 어려움을 초래할 수 있을 뿐만 아니 라 불합리하게 특정 벤더에 종속되는 결과를 낳을 수 있다.
- 소프트웨어의 선정이 불법적인 것에 의한 선택이어서는 안된다. 특정 기업의 로비활동이나 특정인에 대한 뇌물수수 등을 통한 소프트웨어의 구매가 이루어져서는 안된다.
- 소프트웨어의 선정에 있어서 위 조항들의 기준에 맞는지 충분 한 검토

를 해야 한다. 그리고, 충분한 검토가 이루어졌는지에 대한 보고서를 작성하도록 한다.

- 소프트웨어의 구매시 제조사와 협의하여 라이센스를 체결하도 록 한다. 단, 본 가이드라인의 조항들을 충분히 충족할 수 있는 라이센스를 체결하고 소프트웨어의 유지보수 및 재사용 을 위해 소스코드의 사용 허락에 대한 내용을 삽입하고, 리버스 엔지니어링의 제한을 규정하지 않도록 한다.
- 독점 소프트웨어는 소프트웨어에 대한 새로운 라이센스 방식을 도입하고 있으나, 이것은 구매자에게 불리한 조항들 또한 삽입 되어 있을 수 있다. 예를 들어, 현재 소프트웨어에 대해서 클릭랩 라이센스를 통해 소프트웨어 구입 이후의 사용에 대한 제 한을 하고 있고, 미국에서는 판례를 통해 이러한 라이센스의 효력이 점차 확대되어 인정되고 있다.
- 라이센스는 따로 체결할 수 있는 사안이므로, 가이드라인의 요구사항들을 충족할 수 있도록 새로운 라이센스를 체결하도록 하여 향후 소프트웨어로 인해 추가비용이나 특정 기업의 제품 에 종속되는 것을 방지하도록 해야 한다. 특히 OEM 방식으로 하드웨어와 함께 구매된 소프트웨어들에 대해서는 해당 소프트웨어 사용시 먼저 라이센스에 대한 제한 사항을 확인하고 이러한 제한 사항에 대해서 해당 소프트웨어의 제조사와 협의하여 사용 여부를 판단하도록 한다.
- 소스코드의 사용 허락이나 리버스 엔지니어링의 허용은 차후의 불합리한 추가 비용을 줄여주고, 해당 소프트웨어 이외의 시스템 도입시 필수적인 요소로 작용할 수 있다. 소스코드의 확보 와 리버스 엔지니어링을 통한 소프트웨어에 대한 분석을 통해 소프트웨어의 유지보수뿐만 아니라 다른 소프트웨어에 기존 소프트웨어에 사용된 데이터를 이전하는데에도 필요할 수 있다. 아. 독점 소프트웨어를 구매하는 경우 소스코드의 획득과 활용은 가능하지만 공개를 제한하는 라이센스를 하는 것이 불가피한 경우, 해당 소프트웨어를 대체할 수 있는 공개 소프트웨어의 도입 가능성 여부를 먼저 충분히 검토하고 해당 독점 소프트웨어를 구매하도록 한다.
- 소프트웨어의 소스코드 공개를 통하여 컴퓨터 기술의 보급과 확산에

기여할 수 있어 소프트웨어 기술과 산업 발전을 이끌 수 있다.

- 소스코드의 공개가 이루어지는 것이 국내 기술 발전에 보다 바람직하므로 소스코드 공개 여부를 소프트웨어 체택시에 우선적으로 고려한다. 자주문형 소프트웨어 구매 또는 정보화를 위한 시스템통합(SI)사업에 대한 적절하고 공정한 보상이 이루어지도록 한다.
- 적절하고 공정한 보상이 이루어지지 않는 경우 기업간의 입찰 을 위한 과도한 경쟁으로 인해 납품하는 소프트웨어 가격이나 서비스 비용이 턱없이 내려가는 경우가 발생할 수 있고 이것 은 해당 기업의 손실로 이어질 수 있다.
- 이러한 손실은 단순히 해당 사업에만 한정되지 않고 향후 해당 기업의 사업에도 영향을 미칠 수 있고 이에 따라 관련 산업의 서비스 가격에 기형적인 영향을 줄 가능성이 있다. 특히 국내 현실이 대기업에 많은 부분이 종속된 시장 구조를 가진 상황에 서는 더욱 그러하다.
- 이런 상황은 소프트웨어 시장이나SI 시장의 불안정을 야기할 수 있다. 정부나 공공기관의 구매가 도리어 시장에 심각한 타 격을 주지 않도록 하기 위해 구매 또는 사업 발주시 이에 대 한 적절하고 공정한 비용을 책정하여 지불하도록 해야 한다.
- 지불할 비용의 책정은 최저가의 발주 비용이 아니라 여러 가지 상황을 종합하여 합리적인 차원에서 이루어져야 한다. 비용에 는 소프트웨어 비용과 이에 대한 서비스 비용을 포함하도록 한다. 그리고 지급 비용에 대한 것 이외의 과도한 요구를 하지 않도록 한다.
- 만약 특정 소프트웨어나 서비스에 대해 일정한 가격을 정하는 경우에는 시장조사를 선행하여 일반적으로 인정될 수 있는 가격을 책정하도록 해야 한다.

바. 정보시스템 성능평가 모델[정보진06]

1) 연구개발의 목적 및 필요성

정보기술에 기반을 둔 현대 사회의 모든 조직에서 정보시스템은 추구하는 전략적 목표를 달성하기 위한 수단임과 동시에 필수 불가결의 지원도구라는 것은 주지의 사실이다. 즉, 정보시스템은 조직이 추구하는 목표를 효율적으로 달성하기 위한 전제조건이라 할 수 있다. 그러나 정보시스템은 목표를 달성하기 위해서 정의된 과업을 수행하기 위한 도구이기 때문에, 정의된 과업이 효율적으로 수행되어 조직이 추구하는 목표를 효율적으로 달성될 수 있느냐는 정보시스템이 논리적으로 어떻게 작동하느냐? 그리고 과업을 성공적, 효율적으로 수행하기 위해서 필요한 요건 – 물리적인 하드웨어 규모와 소프트웨어의 성능 – 을 갖추고 있는가? 에 달려 있다.

정부의 공부문의 정보화에 대한 투자 규모는 지속적으로 증대하고 있는데 최근 5년간 공공부문 정보화 예산의 연평균 증가율은 17.9%이고, '07년도 정보화 예산의 규모는 약4조5천억 원에 이르고 있다. 이러한 정보화 예산 중 60% 이상은 하드웨어 도입 비용으로 사용된다. 그러나 업무 특성 및 환경을 고려하는 적정한 시스템 도입 기준 등이 미비하여 예산의 낭비가 발생하고 이로 인한 정보화 투자 효율이 낮다는 지적이 일고 있다. 또한, H/W의 경우, 공공기관들이 시스템 도입 시 참고할 만한 객관적인 기준이 미비하여 실제 필요한 정보시스템 자원 보다 과다 책정되어 예산이 낭비되거나, 과소 책정되는 경우도 발생하고 있다.

더욱이 한국정보사회진흥원이 매년 시행하는 공공자원조사에 따르면 현재 공공기관에서 운영 중인 정보시스템 하드웨어의 평균 CPU 사용률이 30% 미만인 경우가 44.5%, 평균 디스크 사용률 50%미만인 경우가 58.4%인 것으로 조사되고 있으며, 대부분의 공공기관들은 SI업체나 하드웨어 제조사들이 제공하는 자체 산정 기준에 의존하고 있다.

따라서 정부에서는 하드웨어 도입자원의 효율적인 산정을 위한 필요성을 인식하고 정보통신부와 한국전산원을 중심으로 “H/W 용량산정 기법연구”(2002, 한국전산원), “정보시스템 용량산정 기술 및 프레임워크연구”(2003, 한국전산원), “정보시스템 규모별 용량산정 연구”(2004, 한국전산원) 등의 연구들을 수행한 결과, 정보시스템 하드웨어 규모산정 지침(2005, 한국전산원), 정보시스템 하드웨어 규모산정 가이드라인(2006, 한국전산원)에 관한 연구를 현재 진행하였다.

이들 H/W 규모산정 연구는 국제적인 공인인증기관인 TPC와 SPEC의 벤치마크를 토대로 하고 있다. 그러나 2005년 이후 이들 기관의 벤치마크 기준은 정보기술의 발전에 따라 다양하게 변화하고 있으며 따라서 기존 H/W 규모산정에 근간이 되고 있는 TPC와 SPEC의 TPC-E, TPC-APP, SPECjbb2005, SPECjappserver2004, SPECWeb 2005 등 신규기준의 적용이 필요한 상황이다.

⊙ 연구의 내용

기존 H/W 규모산정 기준 연구는 국제적인 공인인증기관인 TPC와 SPEC의 벤치마크를 토대로 하고 있다. 2005년 이후 이들 기관의 벤치마크 기준은 정보기술의 발전에 따라 다양하게 변화하고 있으며 따라서 본 연구에서는 기존 H/W 규모산정에 근간이되고 있는 TPC와 SPEC의 TPC-E, TPC-APP, SPECjbb2005, SPECjappserver2004, SPECWeb2005 등 OLTP, WAS, WEB 서버에 대한 신규 벤치마크를 분석하고 이를 기존 기준에 반영하기 위한 방안을 제시하고자 하며, 세부적인 연구의 내용은 다음과 같다.

○ 정보시스템 성능평가 모델 관련 국내외 동향
  - 국외 동향 (TPC, SPEC 동향을 중심)
  - 국내 동향 (국내 하드웨어 규모산정 현황과 연계)
○ TPC 및 SPEC의 신규벤치마크 기준 분석
  - TPC-E, TPC-APP, SPECjbb2005, SPECjappserver2004, SPECWeb2005 모델분석
  - 모델의 시사점과 적용방안
  - 모델의 주요 현황 및 특성
  - 기존 관련 모델과의 비교
  - 적용방안
○ 기존 규모산정지침으로의 적용방안 제시
  - 적용 시 고려사항
  - 활용방안

정보시스템 성능평가모델 도입방안에 관한 연구는 정보시스템 성능평가 모델 관련 국내·외 동향을 파악한 후, 성능평가 국제기관(Transaction Processing Performance Council: TPC, Standard Performance Evaluation Corporation: SPEC)의 최신 성능평가기준을 분석하고, 기존의 기준과의 비교 검토를 거친 후, 성능평가기준에 반영할 요소를 추출하고 활용방안을 제시하기 위해서 5인으로 구성된 전문가위원회를 중심으로 수행되었다.

<그림10> TPC 벤치마크 기준

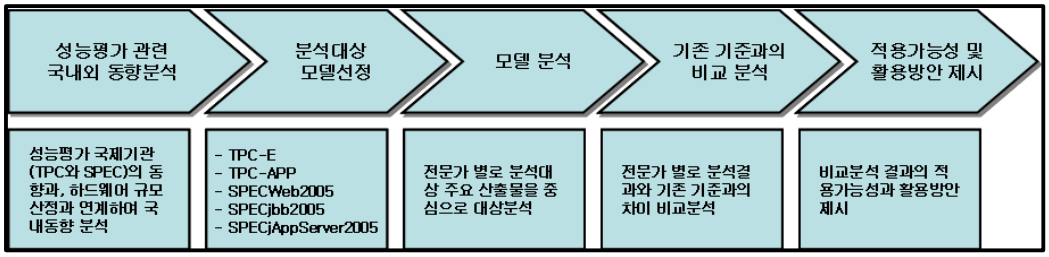
<그림11> SPEC 벤치마크 기준

정보시스템 성능평가모델 도입방안 연구를 위한 전문가위원회는 먼저 분석대상 모델을 선정하여 분석대상 모델을 분석하였으며, 기존 기준과의 비교분석을 통해서 새로이 추가되거나 혹은 변경된 내용을 추출하고, 적용

가능성을 파악하여 최신 정보시스템 성능평가 모델로의 적용방안을 제시하였다. 전문가위원회에서 선정한 분석대상 모델은 <표20>과 같으며, 성능평가모델 분석 작업은 <그림12> 과 같은 절차로 진행하였다.

<표20> 분석대상 성능평가모델

대 상	분석대상 산출물
TPC-E	• TPC-E V0.30 Draft Specification
TPC-APP	• TPC-App is Version 1.1.1.
SPECWeb2005	• SPECweb2005 Design Document[P] • SPECweb2005 Benchmark Run and Reporting Rules • SPECweb2005 User's Guide
SPECjbb2005	• SPECjbb2005 Design Document(P) • SPECjbb2005 User's Guide • SPECjbb2005 Run and Reporting Rules
SPECjApp Server 2004	• SPECjAppServer2004 Design Document(P] • SPECjAppServer2004 Run and Reporting Rules • SPECjAppServer2004 User's Guide



<그림12> 성능평가모델 분석 작업절차

분석대상 모델 및 주요 분석대상 산출물의 선정은 전문가위원회에서 정

보시스템 유형과 성능평가 국제기관의 최신 모델을 고려하여 선정하였으며, 분석 작업은 각 전문가별로 모델분석, 기존 모델과의 비교분석, 적용가능성과 활용 방안에 대한 연구를 진행하되, 단계마다 상호 교차 검증하는 방식으로 진행하여 최종 합의를 거친 후 새로운 성능평가기준으로의 적용가능성과 종합적인 활용방안을 제시하였다.

2) 정보시스템 성능평가 모델 국내외 동향

⊙ 국외 동향

▪ TPC의 벤치마크 동향

TPC는 트랜잭션 처리와 데이터베이스 성능 평가 기준을 정의하고, 객관적이고 입증할 수 있는 TPC 성능 데이터를 산업에 전파하기 위해 설립된 비영리 법인이다. 일반적으로 트랜잭션은 다양한 비즈니스와 컴퓨터 기능에 적용된다. 컴퓨터 기능으로 트랜잭션은 하나의 서브시스템에서 다른 서브시스템으로 디스크 판독/기록, 오퍼레이팅 시스템 호출, 또는 데이터 전송 형식을 포함하는 운영체계라 할 수 있다. TPC 성능평가 기준이 컴퓨터 기능과 운영의 측정과 평가에 관련되어 있지만, TPC는 일반적으로 트랜잭션을 비즈니스 세계에서 이해할 수 있는 것으로 여긴다: 상품, 서비스 또는 화폐의 상업적 교환. TPC에 의해 정의된 것처럼, 전형적인 트랜잭션은 재고 관리(상품), 항공 예약(서비스)또는 뱅킹(화폐)을 위해 데이터베이스 시스템의 갱신 처리를 포함할 것이다. 이러한 환경에서, 수많은 고객 또는 서비스 대리인은 데이터베이스에 연결된 단말기 또는 데스크톱 컴퓨터를 통해 트랜잭션을 입력하고, 관리한다. 전형적으로, TPC는 시스템과 데이터베이스가 단위 시간당 (예를 들면, 초당 또는 분당 트랜잭션) 얼마나 많은 트랜잭션을 수행할 수 있는지에 대한 트랜잭션 처리(TP)와 데이터베

이스(DB)성능을 측정하는 성능 평가 기준을 제공한다.

- 전체 평의회 (Full Council)

모든 주요한 결정은 Full Council에 의해 만들어진다. TPC를 구성하는 회원사는 1장의 투표권을 가진다. 안건이 통과되기 위해서는 회원 3분의 2이상의 지지를 얻어야 한다.

- 운영위원회 (Steering Committee)

매년 회원사로에서 뽑은 5명의 대표자로 구성된다. 운영위원회는 TPC 관리와 지원 활동을 감독하고, 전체적인 방향과 권고를 전체 평의회에 제공하는 책임이 있다. 그러나 전체 평의회가 모든 실재적인 TPC 문제를 결정한다. TPC의 작업을 촉진하기 위해, 전체 평의회는 2종류의 소위원회를 만들었다: 상임 소위원회와 기술 소위원회(Standing and technical Subcommittees). 상임 소위원회는 TPC를 위해 행정과 공공 관계 그리고 문서 작성을 감독하고 관리하는 상설 위원회다. 기술 소위원회는 개발 작업이 완료된 후에 성능 평가 기준 제안서를 개발하고, 성능 평가 기준을 유지, 전개하기 위해 만들어 졌다.

* 상임 소위원회(Standing Subcommittees)
* Steering Committee (SC): 운영위원회
* Technical Advisory Board (TAB): 기술자문위원회
* Public Relations Committee (PRC): 홍보위원회

- 기술 소위원회(Technical Subcommittees)

* TPC-C Subcommittee: TPC-C 소위원회 : 이 소위원회는 OLTP 환경에 언급하는 TPC-C 성능 평가 기준을 소유한다.
* TPC-H Subcommittee: TPC-H 소위원회 : 이 소위원회는 의사 결정 지원 어플리케이션(복합 질의 모델과 더불어)에 대한 TPC-H 성능 평가 기준을 소유한다.
* TPC-App Subcommittee: TPC-App 소위원회 : 이 소위원회는 웹기반 전자상거래 환경에 대한 TPC-App 성능 평가 기준을 소유한다.

▪ 성능벤치마크 기준

TPC는 1998년에 설립된 이후 2006년 까지 매년 5-6회의 성능 평가 기준 상태 보고서를 발간하고 있다. 1998년 이후 현재까지 개발된 벤치마크 중에서 5개를 폐기하였으며, 3개를 사용 중이며, 2개의 벤치마크를 개발 중에 있다. 성능평가 대상은 OLTP, 의사결정지원, 어플리케이션 서버 & 웹서비스 등 3개이다. 성능평가 대상을 첫째, 사용하지 않은 과거의 성능 평가 기준, 둘째, 현재 사용하는 성능 평가 기준, 셋째, 미래에 사용하기 위해 준비 중인 성능 평가 기준으로 구분하면 <표21>과 같다.

<표21> TCP의 성능평가 기준

성능 평가 대상	사용하지 않는 과거 성능 평가 기준	사용 중인 현재 성능 평가 기준	개발 중인 미래 성능 평가 기준
OLTP	○ TPC-A ○ TPC-B	○ TPC-C	○ TPC-E
의사결정지원	○ TPC-D ○ TPC-R	○ TPC-H	○ TPC-DS
어플리케이션 서버 & 웹서비스	○ TPC-W	○ TPC-App	

- 폐기된 벤치마크

과거에 개발되어 활용되었으나, 현재 폐기되어 사용하지 않는 벤치마크는 TPC-A, TPC-B, TPC-D, TPC-R, TPC-W 등 5개 다.

* TPC-A : TPC-A는 온라인 트랜잭션 처리(OLTP) 어플리케이션의 전형적인 업데이트 집약형 데이터베이스 환경에서 성능을 측정한다. 1989년 11월에 발표되어 1995년 6월 6일에 폐기되었다.
* TPC-B : TPC-B 시스템이 초당 얼마나 많은 트랜잭션을 수행할 수 있는가를 측정한다. 1990년 8월에, TPC는 두 번째 성능 평가 기준인 TPC-B을 승인했다. TPC-A과 대조적으로, TPC-B은 OLTP 성능 평가 기준이 아니다. 오히려, TPC-B은 데이터베이스 스트레스 시험으로서 보일 수 있다. 폐기된 날짜는 1995년 6월 6일이다.
* TPC-D : TPC-D는 크고 복잡한 데이터 구조에 대응대는 복잡하고 장기간 실행하는 질의를 요구하는 의사 결정 지원(DS)의 넓은 범위를 표현한다. TPC은 1995년 4월에 네 번째 성능 평가 기준을 승인했다. 폐기된 날짜는 1999년 4월 6일이다. TPC-D는 크고 복잡한 데이터 구조에 대응하는 복합적이고 오랜 기간의 수행 질의를 필요로 하는 의사 결정 지원(DS)어플리케이션을 표현한다.
* TPC-R : TPC-R은 비즈니스 보고서 작성, 의사 결정 지원 성능 평가 기준이다. 폐기된 날짜는 2005년 1월 1일 이다. TPC Benchmark™R(TPC-R)은 TPC-H과 유사한 의사 결정 지원 성능 평가 기준이다. 그러나 이것은 질의에 의한 고급 지식에 기초한 부가적인 최적화를 허용한다.

* TPC-W : TPC-W는 거래 처리용 웹 e-Commerce 성능 평가 기준이다. 폐기된 날짜는 2005년 4월 28일이다. TPC Benchmark™ W(TPC-W)는 업무 처리의 웹 성능 평가 기준이다. 평가 작업은 비즈니스 지향 업무 처리의 웹 서버 활동을 시뮬레이션 하는 통제된 인터넷 상거래 환경에서 수행된다. TPC-W의 마지막 버전은 Version 1.8이다. TPC-W Version 2는 지금 퍼블릭 리뷰에 이용할 수 있다.

- 현재 활용되는 벤치마크

2006년 현재 TPC는 TPC-App, TPC-C, TPC-H 등 3개의 벤치마크 기준을 운영하고 있다.

* TPC-App : TPC Benchmark™ App(TPC-App)은 어플리케이션 서버와 웹 서비스 성능 평가 기준이다. 성능 평가 작업은 24x7 환경에서 운영되는 B2B 업무 처리 어플리케이션 서버의 활동을 시뮬레이션 하는 관리 환경에서 수행된다. TPC-App은 어플리케이션 서버 시스템의 성능을 보여준다. 아래 특성과 같은 환경에서, 판매용 어플리케이션 서버 제품, 메시징 제품과 데이터베이스 등에 대한 성능 평가 작업을 수행한다.

✓ 다중 온라인 비즈니스 세션

✓ 상업용 어플리케이션 환경

✓ 데이터 교환을 위한 XML 문서와 SOAP의 사용

✓ B2B 어플리케이션 로직

✓ 분산 처리 관리

✓ 신뢰할 수 있고 내구성이 있는 메시징

✓ 데이터베이스 액세스와 업데이트가 있는 다이내믹 웹 서비

스 응답 생성

- ✓ 비즈니스 기능에 대한 다양한 처리 형태의 동시 실행
- ✓ 다양한 크기, 속성, 관계가 있는 많은 테이블로 이루어진 데이터베이스
- ✓ 트랜잭션 통합

* TPC-C : TPC-C는 데이터베이스에 대응하는 트랜잭션을 실행하는 완전한 컴퓨팅 환경을 시뮬레이션 한다. 성능 평가 기준은 주문 입력 환경의 주요한 활동(트랜잭션)과 관련되어 있다. 이 트랜잭션은 주문 입력과 전달, 지불 기록, 주문 상태 확인, 창고에서의 재고 수준 감시를 포함한다. TPC-C는 온라인 트랜잭션 처리 성능 평가 기준이며, TPC-C 성능 평가 기준의 현재 버전은 Version 5.7이다. 아래 특성과 같은 환경에서, 시스템 구성요소에 대한 성능 평가 작업을 수행한다.

- ✓ 복잡한 다중 트랜잭션 유형의 동시 실행
- ✓ 온라인과 보류중인 트랜잭션 실행 모드
- ✓ 다중 온라인 터미널 세션
- ✓ 적당한 시스템과 어플리케이션 실행 시간
- ✓ 중요한 디스크 입/출력
- ✓ 트랜잭션 무결성(ACID 속성)
- ✓ 주키와 보조키를 통한 데이터 접근의 비균등(Non-uniform) 분배
- ✓ 다양한 크기, 속성과 관계가 있는 많은 테이블로 이루어진 데이터베이스
- ✓ 데이터 접근과 업데이트의 경합

* TPC-H : TPC-H TPC Benchmark™H(TPC-H)은 의사 결정 지원 성능 평가 기준이다. 이것은 비즈니스 지향의 특별한 질의와 동시 병행의 데이터 변경으로 구성된다. 데이터베이스에서 생성되는 질의와 데이터는 산업 전반에 걸치는 의미를 갖고 있다. 이 성능 평가 기준은 많은 양의 데이터를 조사하고, 고도로 복잡한 질의를 실행하고, 중요한 비즈니스 질문에 응답해 주는 의사 결정 지원 시스템을 설명한다.

- 신규 성능 평가 기준

2006년 10월26일, TPC는 내슈빌에서 General Council 회의를 개최했다. OLTP 측면에서, TPC-E 성능 평가 기준은 OLTP의 최종 드래프트이다. 의사 결정 지원 측면에서, TPC-H 소위원회는 가격 부분에서 복구 로그와 누락에 대한 문제가 있었지만 TPC-H Version 2.6.0을 배포했다. DS 성능 평가 기준 리뷰도 1부터 100까지의 TB 범위에 대한 측정 요인을 가진 프로토타입 작업을 계속하고 있다. TPC-App 소위원회는 TPC-App 성능 평가 기준을 시각화하기 위한 작업을 하고 있다.

* TPC-E : 2006년 10월 이후 2달 동안, TPC-E 소위원회는 TPC-E 사양에 대한 상세한 리뷰를 하고, 논평에 대한 피드백을 하였다. 이 기간 동안 논평 결과를 받았고, 새로운 명세 드래프트와 소스 코드를 알리게 되었다. 피드백 논평은 소위원회에 의해 처리되었다.
* TPC-DS : TPC-DS 소위원회는 TPC-DS 성능 평가 기준을 프로토타핑 하는 것에 집중했다. 7개 기업이 1부터 100까지의 TB 범위를 갖는 척도 요인으로 프로토타이핑 작업을 하고 있다.

- 기존 모델의 최근 동향(2006년 12월 기준)

  * TPC-C : 2006년 10월 회의에서 TPC-C 유지보수 소위원회는 개최하지 않았다. 현 단계에서 성능 평가 기준은 기능적으로 안정화되었고, 소위원회는 이번에는 어떠한 공개된 행동을 하지 않았다. 소위원회는 TPC-C 사양의 미래 버전에 대한 토론과 종속물을 위해 기술자문위원회(TAB: Technical Advisory Board)의 해석과 판정과 더불어 성능 평가 기준 감시를 계속할 것이다.
  * TPC-H : TPC-H 소위원회는 TPC-H 사양을 조금 개선한 버전 2.6.0을 배포했고, 가격 부문(조항 7.2.3)에서 감독을 수정한다. 이 조항은 사양에서 잘못하여 생략되었고, 새로운 사양에서 재 소개 된다. 개정된 2.6.0은 최소 8시간의 복구 로그(조항 7.1.3.3)를 지원하는 요구 사항을 삭제한다. 견본의 실행 요약은 시스템 구성 요소의 합계와 노드에 따른 시스템 구성 요소의 수를 분명히 진술하기 위해 수정되었다. 버전 2.6.0의 유효 일자는 2007년 1월 1일이다. 성능 평가 기준 결과는 Version 2.6.0에 대응하여 즉시 출판된다. 2007년 1월 1일 부터는 의무 사항이다.
  * TPC-App : TPC-App 소위원회도 10월 회의는 개최하지 않았다. TPC-App 소위원회는 성능 평가 기준으로 기술을 수용하기 위해 시각화의 정의에 대한 작업을 계속하고 있다.

▪ SPEC의 벤치마크 동향

SPEC(Standard Performance Evaluation Corporation)이라고 이름 붙여진 시스템 성능 평가 기업은 시장에서 실제적이고 표준화

된 성능 테스트가 반드시 필요하다고 느낀 몇몇의 워크스테이션 판매자들에 의해 1988년에 설립되었다. SPEC는 60개 이상의 회원사를 가진 성공적인 성능 표준 회사의 하나로 성장했다. SPEC은 우리의 목표를 지원하는 모든 회사 또는 조직들에게 개방하는 비영리 법인회사이다. SPEC는 원래 CPU 매트릭스를 고안하는 워크스테이션 벤더의 몇 명의 사람들과 함께, 3개의 다양한 그룹들로 이루어진 조직으로 발전하였다.

- The Open Systems Group (OSG)

OSG는 최초의 SPEC 위원회이다. 이 그룹은 개방 시스템 환경에서 운영되는 데스크톱 시스템, 최고급 워크스테이션과 서버들을 위한 벤치마크에 초점을 둔다. 6개의 워킹그룹으로 구성되었다.

* CPU : SPECmarks와 다른 CPU 벤치마크들 (SPECint, SPECfp, SPECrates, 등등)을 수행하는 사람들
* JAVA : JVM98, JBB2000와 JBB2005, 자바 클라이언트 서버 사이드 벤치마크, jAppServer 자바 엔터프라이즈 어플리케이션 서버 벤치마크를 수행하는 사람들 .
* MAIL : SPECmail2001, 소비자 인터넷 서비스 제공자(ISP), 메일 서버 벤치마크를 수행하는 사람들
* POWER-PERFORMANCE : Power-Performance 위원회는 서버 클래스 컴퓨터들을 위한 에너지 효율을 평가하기 위해 첫 번째 세대 SPEC 벤치마크를 개발한다.
* SFS : SFS93(LADDIS), SFS97, SFS97_R1을 가져다 준 사람들로, 다른 파일 서버 벤치마크를 수행한다.
* WEB : WEB96, WEB99, WEB99_SSL, WEB2005, 웹서버 벤치마크를 수행하는 사람들

- The High-Performance Group (HPG)

HPG는 표준화된 크로스 플랫폼 성능 평가를 위한 고성능 컴퓨팅 어플리케이션을 표현하는 벤치마크를 만들고 유지하기 위한 포럼이다. 이 벤치마크는 체계적인 멀티프로세서 시스템, 워크스테이션 클러스터, 분산 기억 병렬 시스템, 전통적인 벡터와 벡터 병렬 슈퍼컴퓨터와 같은 고성능 시스템 구조들을 대상으로 한다.

- The Graphics Performance Characterization Group (GPC)

SPEC/GPC 그룹은 일관되고 반복되는 그래픽 벤치마크와 성능 보고 절차들을 개발하는 프로젝트 그룹을 위한 조직이다. SPEC/GPC 성능 평가 기준은 대중적인 그래픽 어플리케이션의 사용자 경험을 반영하는 방법으로 성능을 평가하는 세계적인 표준이다.

▪ 벤치마크 기준

SPEC의 벤치마크 중 우리 연구의 대상인 개방 시스템 그룹(OSG)의 성과를 중심으로 살펴본다. 성능평가 대상은 CPU, 자바 클라이언트/서버, 메일 서버, 네트워크 파일 시스템, 전원과 성능, 웹서버 등 6개이다. 성능평가 대상을 첫째, 사용하지 않은 과거의 성능 평가 기준, 둘째, 현재 사용하는 성능 평가 기준, 셋째, 미래에 사용하기 위해 준비 중인 성능 평가 기준으로 구분하면 <표22>와 같다.

<표22> SPCE의 성능 평가 기준

성능 평가 대상	사용하지 않는 과거 성능 평가 기준	사용 중인 현재 성능 평가 기준	개발 중인 미래 성능 평가 기준
CPU	○ CPU92 (obsolete benchmark) ○ CPU95	○ CPU2006 ○ CPU2000	
자바 클라이언트/서버	○ jAppServer2002 ○ jAppServer2001 ○ JBB2000	○ jAppServer2004 ○ JBB2005 ○ JVM98	
메일 서버		○ MAIL2001	○ SPECimap
네트워크 파일 시스템	○ SFS97(2.0) ○ SFS93(LADDIS)	○ SFS97_R1 V3.0	
전원과 성능			○ SPECpower
웹서버	○ SPECweb96 ○ SPECweb99 ○ SPECweb99_SSL	○ SPECweb2005	

- CPU

  * CPU2006 : 서로 다른 컴퓨터 시스템 상에서 계산-집약적인 (compute-intensive) 작업을 비교하기 위해 사용할 수 있는 성능 측정도구를 제공하기 위해 설계되었다. SPEC CPU2006은 2종의 성능 평가 기준이 있다.

  * CPU2000 : 서로 다른 컴퓨터 시스템 상에서 계산-집약적인 (compute-intensive) 작업을 비교하기 위해 사용할 수 있는 성능 측정도구를 제공하기 위해 설계되었다. SPEC CPU2000은 2종의 성능 평가 기준이 있다. 현재의 버전은 CPU2000 V1.3이다.

- 자바 클라이언트/서버

  * jAppServer2004 : SPECjAppServer2004는 J2EE 1.3 어플리케이션 서버의 성능을 측정하기 위해 설계되었다. 이 성능 평가

기준은 웹 계층, JMS와 SPECjAppServer 2002의 변경 사항의 추가 작업을 포함한다.

* JBB2005 : 전형적인 자바 비즈니스 어플리케이션을 실행하는 서버의 성능을 평가하는 벤치마크이다. JBB2005는 도매 공급자를 위해 주문 처리 어플리케이션을 표현한다. 성능 평가 기준은 Java Virtual Machine(JVM)서버의 하드웨어와 소프트웨어 측면의 성능을 평가하기 위해 사용될 수 있다.

* JVM98 : SPECjvm98은 사용자가 JVM 클라이언트 플랫폼의 결합된 하드웨어와 소프트웨어 측면의 성능을 평가할 수 있도록 한다. 소프트웨어 측면에서, JVM98은 JVM, 저스트 인 타임의(JIT)컴파일러와 오퍼레이팅 시스템 실행의 효율성을 측정한다. 하드웨어 측면에서, CPU(정수와 부동 소수점), 캐시, 메모리와 다른 플랫폼에 특유한 성능을 포함한다.

- 메일 서버

* MAIL2001 : 표준화된 메일 서버 벤치마크는 인터넷 표준 프로토콜 SMTP과 POP3에 기반 하여 이메일 요구를 서비스하는 메일 서버의 역할을 하는 시스템의 능력을 측정하기 위해 설계되었다. 성능 평가 기준은 실제적인 네트워크 접속, 디스크 기억 장치와 클라이언트 작업에 대한 시험을 통해 메일 서버 시스템의 처리율과 응답 시간으로 간주한다.

- 네트워크 파일 시스템

* SFS97_R1(3.0) : SPEC의 성능 평가 기준은 NFS 프로토콜의 버전 2와 3을 사용하는 NFS(네트워크 파일 서버)컴퓨터의 속도와 요구-처리 능력을 평가하기 위해 설계되었다.

– 웹서버

* WEB2005 : SPECweb2005는 웹 서버의 광대역 인터넷 접속에 대한 사용자의 송신 브라우저 요구를 에뮬레이트한다. 이것은 3개의 새로운 작업을 제공한다: 뱅킹 사이트(HTTPS), 전자상거래 사이트(HTTP/HTTPS 혼합)과 지원 사이트(HTTP). 동적 콘텐츠는 PHP과 JSP에서 실행된다. 현재의 버전은 1.10이다.

▪ 새로운 성능 평가 기준

– 메일 서버

* SPECimap : 기업의 이메일 서버 성능을 측정하기 위해 설계된 산업 표준 성능 평가 기준은 SPEC에 의해 개발 중이다. 이 성능 평가 기준은 인터넷 표준 프로토콜 SMTP과 IMAP4에 기초하여, 이메일 요구를 처리하는 서버의 능력을 테스트할 것이다

– 전원과 성능

* SPECpower : SPEC Power-Performance 위원회는 서버 클래스 컴퓨터를 위한 에너지 효율성을 평가하기 제 1세대 SPEC 성능 평가 기준의 개발하기 시작했다. SPECpower 성능 평가 기준 목표는 여러 가지 사용 수준에서 시스템 에너지 사용을 공평하고 일관되게 보고할 수단을 제공하는 것이다.

⊙ 국내 동향

1990년대 중반이후 정부부처와 공공기관들이 대규모의 정보화사

업을 추진하고 있지만, 각 기관에서 발주하는 하드웨어의 경우, 규모와 성능에 대한 객관적인 근거나 자료 없이, 예산을 기준으로 하드웨어 도입을 추진하고 있다. 이러한 문제점을 해결하기 위해, 2002년 이후 정보통신부와 한국정보사회진흥원이 중심이 되어 학계와 업계 전문가들이 참여하는 국내의 정보시스템 하드웨어 규모산정 연구를 진행하고 있다. 하드웨어 규모산정과 하드웨어 성능평가는 하드웨어 도입과 운영에 있어 매우 중요한 2개의 요인이라 할 수 있다. 2002년 이후 지금까지의 연구가 하드웨어 규모산정을 중심으로 진행된 연구였다면, 2006년에 이후에는 하드웨어의 성능측정 중심으로 연구가 진행되고 있다. 이는 하드웨어 규모산정 연구 결과의 가시적인 성과이며 그동안 미진하였던 성능 측정 분야에 대한 추가적인 요구와 연구가 진행되고 있음을 의미한다. 현재까지의 하드웨어 규모산정 연구 결과는 <표23>과 같다.

<표23> 국내 공공부문 대상 하드웨어 규모산정 방식 설정에 관한 선행연구

연구과제	주요 연구내용	수행방식
H/W 용량산정기법 연구(2002)	○ 공공부문 정보화사업과 관련된 하드웨어 용량산정 기법 및 고려사항 ○ 용량산정이 개념 정의 및 절차 제시	한국전산원 자체연구
정보시스템 용량산정 기술 및 프레임워크 연구(2003)	○ 시스템 유형에 따른 CPU, 메모리, 디스크에 대한 용량 산정식 제시 ○ 산정항목의 개념 정의 및 절차 제시	한국전산원 위탁연구
정보시스템규모별 용량산정 기준 연구(2004)	○ 설문조사를 통한 기존 용량 산정식 보완 ○ 항목별 입력값의 재산정 및 주전산기 규모분류	한국전산원 위탁연구
정보시스템 H/W 규모산정 지침 연구(2005)	○ 공공부문 정보화사업에 대한 하드웨어 규모산정 사례수집 및 분석 ○ 하드웨어 규모산정 방식의 보완 및 산정항목별 적용기준 제시 ○ 하드웨어 규모산정 방식 적용을 위한 지침작성	한국전산원 위탁연구
정보시스템 H/W 규모산정 포털 시스템 구축(2005)	○ 하드웨어 규모산정 지침에 기반한 포털 시스템 구축	한국전산원 위탁연구

3) 최신 정보시스템 성능평가 모델분석

⊙ 분석 개요 및 방법

정보기술에 기반을 둔 현대 사회의 모든 조직에서 정보시스템은 추구하는 전략적 목표를 달성하기 위한 수단임과 동시에 필수 불가결의 지원도구라는 것은 주지의 사실이다. 즉, 정보시스템은 조직이 추구하는 목표를 효율적으로 달성하기 위한 전제조건이라 할 수 있다. 그러나 정보시스템은 목표를 달성하기 위해서 정의된 과업을 수행하기 위한 도구이기 때문에, 정의된 과업이 효율적으로 수행되어 조직이 추구하는 목표를 효율적으로 달성될 수 있느냐는 정보시스템이 논리적으로 어떻게 작동하느냐? 그리고 과업을 성공적, 효율적으로 수행하기 위해서 필요한 요건 – 물리적인 하드웨어 규모와 소프트웨어의 성능 – 을 갖추고 있는가? 에 달려 있다.

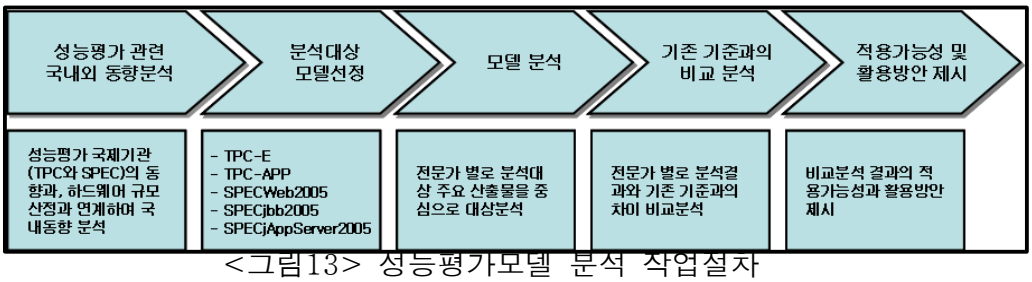
물리적인 하드웨어의 적정 규모산정에 관해서는 정보통신부와 한국전산원을 중심으로 “H/W 용량산정 기법연구”(2002, 한국전산원), “정보시스템 용량산정 기술 및 프레임워크연구”(2003, 한국전산원), “정보시스템 규모별 용량산정 연구”(2004, 한국전산원) 등의 연구들을 수행한 결과, 정보시스템 하드웨어 규모산정 지침(2005, 한국전산원), 정보시스템 하드웨어 규모산정 가이드라인(2006, 한국전산원)이 마련되어 있으나, 정보시스템의 성능에 관한 연구는 아직 체계적으로 진행되고 있지 않은 상황이다. 이에 정보시스템 성능기준을 마련하기 위한 목적으로 정보시스템 성능평가모델 도입방안에 관한 연구를 현재 진행하였다. 정보시스템 성능평가모델 도입방안에 관한 연구는 정보시스템 성능평가 모델 관련 국내•외 동향을 파악한 후, 성능평가 국제기관(Transaction Processing Performance Council: TPC, Standard Performance Evaluation Corporation: SPEC)의 최신 성능평가기준을 분석하고, 기존의 기

준과의 비교 검토를 거친 후, 성능평가기준에 반영할 요소를 추출하고 활용방안을 제시하기 위해서 5인으로 구성된 전문가위원회를 중심으로 수행되었다.

정보시스템 성능평가모델 도입방안 연구를 위한 전문가위원회는 먼저 분석대상 모델을 선정하여 분석대상 모델을 분석하였으며, 기존 기준과의 비교분석을 통해서 새로이 추가되거나 혹은 변경된 내용을 추출하고, 적용가능성을 파악하여 최신 정보시스템 성능평가 모델로의 적용방안을 제시하였다. 전문가위원회에서 선정한 분석대상 모델은 <표24>과 같으며, 성능평가모델 분석 작업은 <그림13> 과 같은 절차로 진행하였다.

<표24> 분석대상 성능평가모델

대 상	분석대상 산출물
TPC-E	• TPC-E V0.30 Draft Specification
TPC-APP	• TPC-App is Version 1.1.1.
SPECWeb2005	• SPECweb2005 Design Document[P] • SPECweb2005 Benchmark Run and Reporting Rules • SPECweb2005 User's Guide
SPECjbb2005	• SPECjbb2005 Design Document(P) • SPECjbb2005 User's Guide • SPECjbb2005 Run and Reporting Rules
SPECjApp Server 2004	• SPECjAppServer2004 Design Document(P] • SPECjAppServer2004 Run and Reporting Rules • SPECjAppServer2004 User's Guide



<그림13> 성능평가모델 분석 작업절차

분석대상 모델 및 주요 분석대상 산출물의 선정은 전문가위원회에서 정보시스템 유형과 성능평가 국제기관의 최신 모델을 고려하여 선정하였으며, 분석 작업은 각 전문가별로 모델분석, 기존 모델과의 비교분석, 적용가능성과 활용 방안에 대한 연구를 진행하되, 단계마다 상호 교차 검증하는 방식으로 진행하여 최종 합의를 거친 후 새로운 성능평가기준으로의 적용가능성과 종합적인 활용방안을 제시하였다.

4) 분석 대상 모델

⊙ TPC BenchmarkTM E (TPC-E)

▪ TPC-E의 목표

TPC-E는 현실 세계의 OLTP환경을 재구성한 가상의 시스템을 통해서 트랜잭션이 수행되는 동안 데이터베이스에 어느 정도의 작업부하가 걸리는지를 측정하여 적정 수준의 CPU, 메모리, 디스크 용량을 산정하려는 목적으로 Transaction Processing Performance Council(TPC)에서 개발한 성능평가 모델이다. 즉, TPC-E의 목적은 OLTP환경의 멀티 애플리케이션 시스템간의 복잡한 데이터흐름을 측정하려는 것이 아니라, OLTP환경에서 트랜잭션이 수행되는 동안 시스템에 영향을 미치는 다양한 요인들을

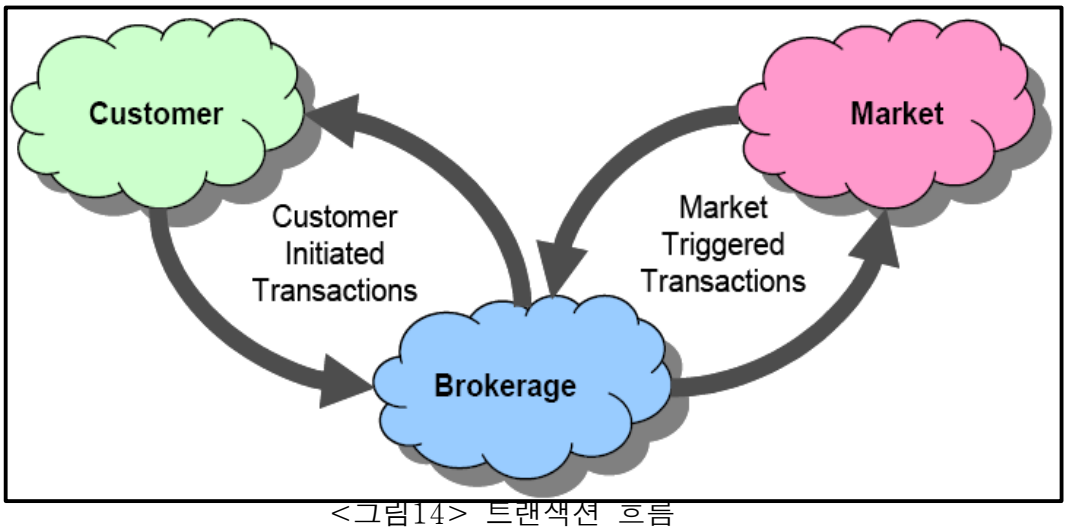
고려하여 정보시스템의 성능을 평가하려는 것이라 할 수 있다. 이 같은 TPC-E의 목적에 부합하여 실제 OLTP환경에서와 동등한 작업부하 수준을 측정하기 위해서 작업부하에 영향을 미치는 다양한 요인들과, 비정상적으로 발생될 수 있는 유지보수 세션을 제외하고 모든 세션의 데이터처리, 테이블에 저장된 데이터 변환처리과정 그리고 데이터베이스와 상관없이 발생할 수 있는 모든 가능한 트랜잭션을 포함하여 OLTP환경을 재구성하여 분석대상으로 하고 있다. 그러나 TPC-E가 다양한 OLTP 애플리케이션 환경을 내포하고 있기는 하지만, 전체 OLTP 요구사항을 반영하고 있지는 않다. 또한 작업부하는 개별 애플리케이션 상황과 환경요인에 따라 다를 수 있기 때문에, 모든 OLTP환경에 적용하는 것은 무리가 있는 것이 사실이지만, 하드웨어의 적정규모를 산정하려는 유저들에게 객관적이고 유용한 성능평가 데이터를 제공할 수 있는 벤치마크 모델이다.

Transaction Processing Performance Council(TPC)은 급변하는 정보기술과 새로운 OLTP환경 요인들을 고려하여 OLTP환경에서의 현실적인 작업부하를 측정할 모델을 개발하기 위해서 위원회를 구성하여 자체 개발 작업을 착수하여 2003년 1월에 TPC-C 후속모델인 TPC-E Draft Version0.15를 공포하였다. 이후 수정 보완 작업을 계속하여 2005년 8월에 Draft Version0.30을 발표하였으며, 2006년 5월 Draft Version0.32d를 발표하였다. Draft Version0.32d가 발표된 이후 현재까지 최종 TPC-E Version이 발표되지는 않고 있기 때문에, 본 분석결과 보고서는 최종 발표된 TPC-E Draft Version0.32d를 기준으로 작성한 것이다.

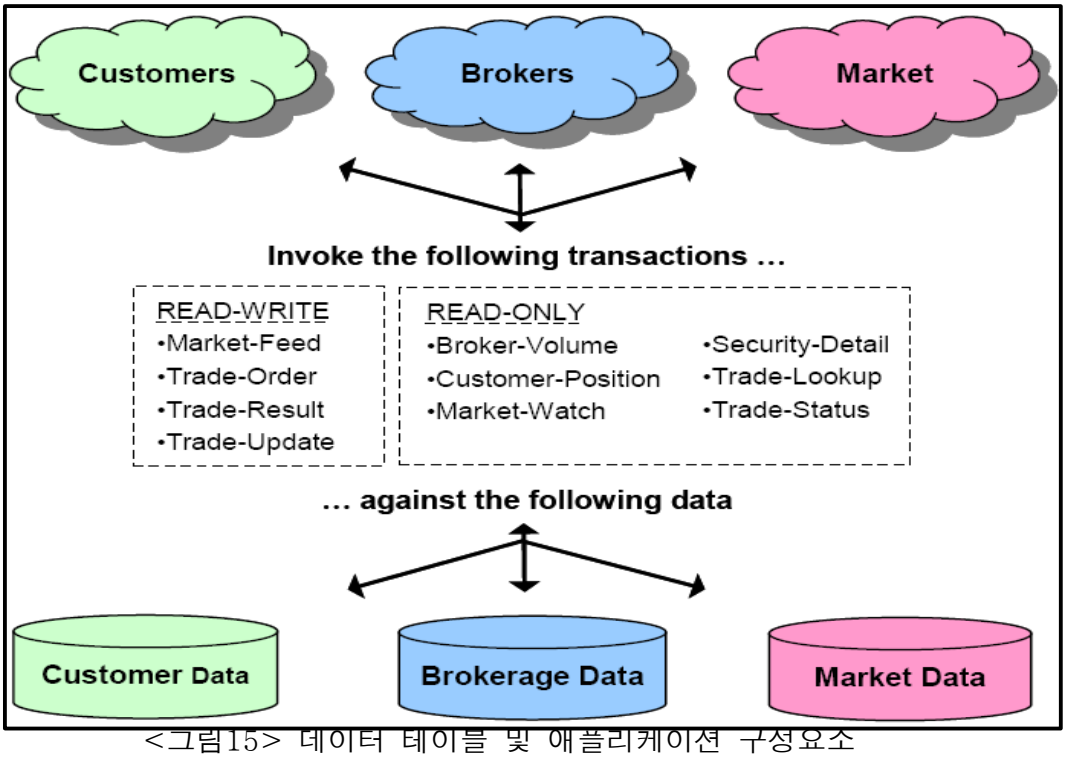
- TPC-E의 비즈니스 및 애플리케이션 환경

TPC-E 모델은 복잡한 OLTP 애플리케이션 환경의 가장 보편적인 형태의 트랜잭션을 수행할 수 있도록 설계된 모델이기 때문에, 사용자들로 하여금 벤치마크 구성요소를 쉽게 이해할 수 있도록 하기 위해서 실제와 같은 상황의 내용들로 구성되어 있으며, 주식•증권거래 중개회사의 구체적인 트랜잭션을 내용으로 하고 있다. 작업부하는 중개회사의 주식•증권거래 처리과정이 중심이며, 다음과 같은 CUSTOMER, BROKERAGE, MARKET, DIMENSION 테이블 세트로 이루어진 구조이다. 다음 <그림 3-2>는 주식 및 증권거래과정의 각 주체간의 트랜잭션 흐름을 나타내며, <그림 3-32>은 트랜잭션 처리과정의 데이터 테이블구조를 설명한다.

본 TPC-E 벤치마크 모델은 MARKET, CUSTOMER, BROKER를 포함하는 세 개의 데이터베이스 테이블에 대해서 실행되는 트랜잭션으로 이루어져 있으며, 네 번째 테이블은 DIMENSION 데이터를 포함하고 있다. 이러한 환경에서의 작업부하 즉, 성능평가를 위해서 DRIVER가 트랜잭션을 위한 데이터를 생성하여 SUT(System Under Test)에 보낸 후에 트랜잭션이 처리되어 다시 DRIVER에 돌아오는데 걸리는 시간, 처리과정에서 생성•추가•변경되는 데이터를 저장하기 위한 소요 공간 등에 의해서 평가된다.



<그림14> 트랜잭션 흐름



<그림15> 데이터 테이블 및 애플리케이션 구성요소

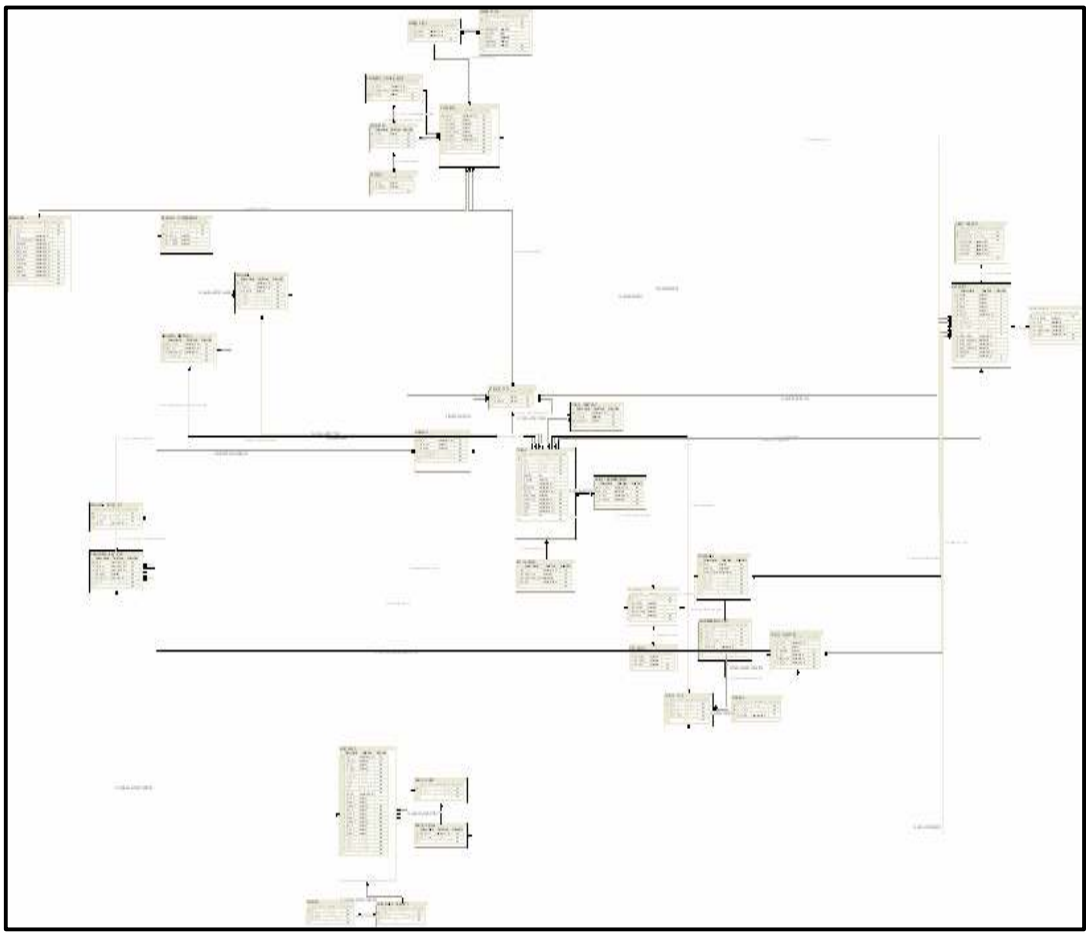
▪ TPC-E의 데이터베이스

- Database Tables and E/R-Diagram

TPC-E 데이터베이스는 다음과 같이 4개 영역 33개의 독립된 개별 테이블로 구성되어 있으며, 이들 테이블 사이의 관계는 <그림16>와 같다.

<표25> TPC-E 데이터베이스 스키마구조

Category	Number of Tables	Tables
MARKET	11	증권•주식, 회사, 거래에 관련 정보관리
CUSTOMER	9	고객관련 데이터 정보관리
BROKER	9	브로커 관련 데이터 관리
DIMENSION	4	여러 테이블들에서 참조되는 일반 정보를 담고 있는 테이블
	33	



<그림16> TPC-E 데이터베이스 E/R Diagram

- Database Size and Table Cardinality

설계된 데이터베이스에서 트랜잭션 로드가 발생하면, 고객서비스를 제공하기 위해서 관련 주체와 상호작용을 하면서 처리해야 할 데이터의 양이 증가하게 된다. 이때 CUSTOMER 테이블의 Cardinality는 TPC-E 벤치마크의 트랜잭션 처리 성능요구(Transaction-Per-Second-E)를 기준으로 하며, TPC-E 에서는 다음과 같은 세 가지 타입의 테이블 크기를 기준으로 한다.

* Fixed Tables : 데이터베이스 규모와 트랜잭션 처리와 관련해서 언제나 같은 테이블 행수를 갖는 고정 테이블(예: TRADE_TYPE 테이블)
* Scaling Tables : CUSTOMER 테이블의 차수와 항상 같은 행수를 갖는 테이블로 예를 들어, 트랜잭션 업데이트 테이블의 경우 행이 업데이트되지만, 테이블 사이즈는 언제나 동일한 테이블
* Growing Tables : CUSTOMER 테이블 차수와 일정 관계를 갖는 행수를 갖지만, 트랜잭션이 실행되면서 트랜잭션 처리율과 비례해서 차수가 증가하는 테이블

이 중에서 Fixed Table이나 Scaling Table의 경우에는 메모리 용량에 직접적으로 큰 영향을 미치지 않지만, Growing Table의 경우 트랜잭션이 수행되는 동안 지속적으로 테이블의 크기가 변화하기 때문에, 메모리 용량에 직접적으로 영향을 미치게 된다. Growing Table의 경우에 있어서 어느 정도 규모로 테이블의 규모가 증가하는지는 초당 처리되는 단일 트랜잭션(=>tpsE) 단위의 CUSTOMER 테이블 행수 – 이를 Scale Factor(SF)라 함 – 에 따라 다르며, TPC-E 에서는 SF=500으로 적용하고 있다.

Fixed Table과 Scaling Table 그리고 Growing Table의 초기 차수와 트랜잭션 처리 과정에서 증가하게 되는 차수 계산은 다음에 의한다.

<표26> Fixed Table의 초기 차수와 차수 계산식

TABLE TYPE: FIXED		
테이블 명	차수	비고
CHARGE	21	-> 7일 *3 customers tiers
COMMISSION_RATE	336	-> 3(customers tiers)*7(trade types)*4 (exchanges)*4(rates)
EXCHANGE	4	-> 4 exchanges
INDUSTRY	102	-> 102 industries
SECTOR	12	-> 12 sectors
STATUS_TYPE	10	-> 10 status types
TAXRATE	321	-> 321 tax rates
TRADE_TYPE	7	-> 7 trade types
ZIP-CODE	14,741	-> 14,741 zip codes

<표27> Scaling Table의 초기 차수와 차수 계산식

TABLE TYPE: SCALING		
테이블 명	최소 차수	차수 계산식
CUSTOMER	1,000	-> 트랜잭션 율 기준, 1000행 단위 추가
CUSTOMER_TAXRATE	2,000	-> customers*2
CUSTOMER_ACCOUNT	~5,000	-> avg of ~5*customers
ACCOUNT_PERMISSION	~7,000	-> accounts*~1.42(permissions per account)
ADDRESS	1,504	-> companies+EXCHANGE(4)+customers
BROKER	100	-> customers*0.1
COMPANY	500	-> 500*(customers/1,000)
COMPANY_COMPETITOR	1,500	-> companies*3
DAILY_MARKET	893,925	-> securities*1,305
FINANCIAL	10,000	-> companies*20
HOLDING_SUMMARY	~49,320	-> ~9,864*accounts
LAST_TRADE	685	-> securities*1
NEWS_ITEM	1,000	-> companies*2
NEWS_XREF	1,000	-> companies*2
SECURITY	685	-> 685*(customers/1,000)
WATCH_LIST	1,000	-> customers*1

WATCH_ITEM	~5,500	-> customers*(avg. of ~5.5 securities per watch list)

<표28> Growing Table의 초기 차수와 차수 계산식

TABLE TYPE: Growing		
테이블 명	최소 차수	계산식
CASH_TRANSACTION	~1,649,630	-> ~0.92*settled(84% of buys &100% of sells are cash)
HOLDING	~362,952	-> ~0.126*settled (가정: ITD=50 & SF=500)
HOLDING_HISTORY	~3,786,553	-> ~1.315*settled (가정: ITD=50 & SF=500)
SETTLEMENT	2,880,000	-> 1* settled
TRADE	2,880,000	-> ((ITD*8hr/day*3600sec/hr*customers/SF)
TRADE_HISTORY	6,911,974	-> ~2.4*trades

<표28>의 테이블 초기 차수를 기준으로 트랜잭션이 수행되는 동안 Growing Table은 초당 다음과 같은 규모로 증가하게 된다.

<표29> Growing Table의 초당 증가하는 규모의 크기 계산식

TABLE TYPE: Growing	
테이블 명	계산식
CASH_TRANSACTION	-> ~0.92*(customers/SF)
HOLDING	-> ~0.054*(customers/SF)
HOLDING_HISTORY	-> ~1.315*(customers/SF)
SETTLEMENT	-> 1*(customers/SF)
TRADE	-> 1*(customers/SF)
TRADE_HISTORY	-> ~2.4*(customers/SF)
TRADE_REQUEST	-> ~(60 to 70)*(customers/SF)

- TPC-E 데이터베이스 트랜잭션

TPC-E는 데이터베이스 내에서 수행되는 트랜잭션을 TPC가 제공하는 트랜잭션 로직인 EGenTxnHarness와, 스폰서가 제공하는 트랜잭션 로직인 Frame으로 이루어져 있으며, "start", "commit", "rollback" 사이의 데이터베이스 상호작용으로 정의하고 있다. TPC-E 데이터베이스 내에서 처리되는 이 같은 트랜잭션은 다음과 같이 12개로 구성되어 있으며, 이들 트랜잭션들은 <그림17>와 같은 Database-Footprint (=> 트랜잭션을 수행하는데 필요한 데이터베이스 인터랙션 조합)로 설명된다.

Example Transaction Database-Footprint		Frame Number		
		1	2*	3
Transaction		Start	Rollback*	Commit
Table Name	**Column**			
CUSTOMER_ACCOUNT	CA_BAL	Reference		
	CA_C_ID	Return		
	CA_TAX_ST	Return		
HOLDING	H_PRICE		Return	
	H_QTY		Modify	
	Row(s)		Remove *	
	1 row		Add *	
TRADE_HISTORY	1 row			Add

<그림17> Database-Footprint의 예

- 트랜잭션 구성 및 로직

TPC-E에서 데이터베이스 트랜잭션은 12개의 트랜잭션으로 구성되어 있으며, 그 중에서 11개의 트랜잭션은 합리적이고 균형 잡

힌 워크로드를 측정하기 위해서 다양한 시스템 기능들을 수행하도록 구성되어 있으며, Data Maintenance 트랜잭션은 트랜잭션에 의해서 수정되거나 변경되지 않는 트랜잭션이다. 12개의 개별 트랜잭션 비중과 액세스 방법과, 구성되어 있는 트랜잭션로직의 수는 다음 <표30>과 같다.

* Trade-Order Transaction : 거래주문을 처리하기 위한 트랜잭션
* Trade-Result Transaction : 주식시장의 거래가격을 평가가격으로 대체하고 실제 총액과 비슷한 거래량을 계산하며, 또 고객의 거래과정에 대한 정보를 나중에 참조할 수 있도록 업데이트하기 위한 트랜잭션

<표30> TPC-E 데이터베이스 트랜잭션의 특징

Transaction	Weight	Access	Category	Frames	Definition
Trade-Order	Heavy	Read-write	Customer initiated	6	Clause 3.3.1
Trade-Result	Heavy	Read-write	Market triggered	6	Clause 3.3.2
Trade-Lookup	Medium	Read-only	Customer initiated	4	Clause 3.3.3
Trade-Update	Medium	Read-write	Customer initiated	3	Clause 3.3.4
Trade-Status	Light	Read-only	Customer initiated	1	Clause 3.3.5
Customer Position	Mid to Heavy	Read-only	Customer initiated	3	Clause 3.3.6
Broker Volume	Mid to Heavy	Read-only	Broker initiated	1	Clause 3.3.7
Security Detail	Medium	Read-only	Customer initiated	1	Clause 3.3.8
Market Feed	Medium	Read-write	Market triggered	1	Clause 3.3.9
Market Watch	Medium	Read-only	Customer initiated	1	Clause 3.3.10
Data Maintenance	Light	Read-write	Time triggered	1	Clause 3.3.11
Trade-Cleanup	Medium	Read-write	Run once before Test Run	1	Clause 3.3.12

* Trade-Lookup Transaction : 고객과 브로커에 대한 정보를

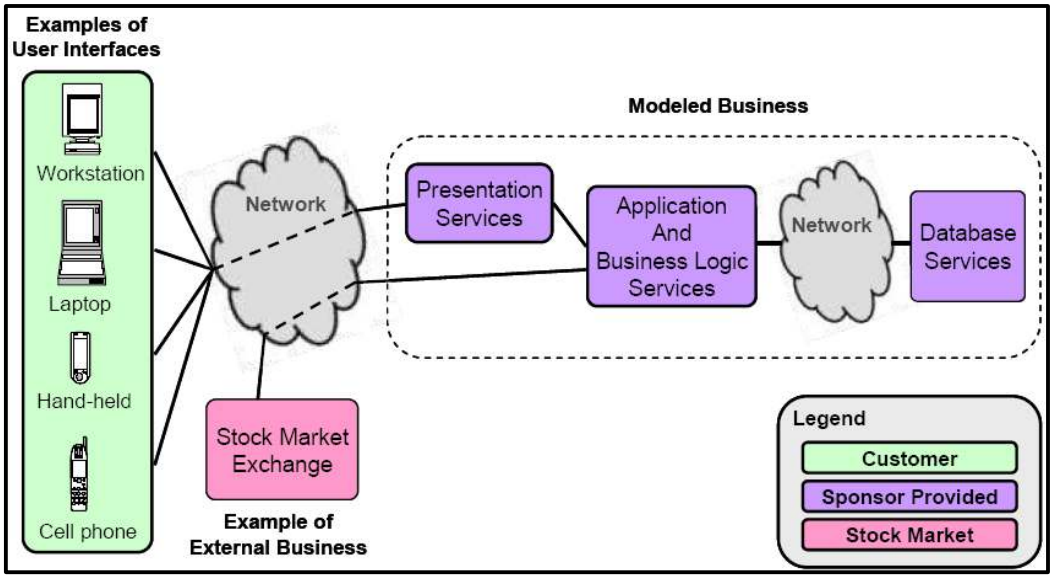
검색하는 트랜잭션

* Trade-Update Transaction : 거래계정/증권 그리고 거래 자체에 대한 문의를 충족시키기 위해서 정보를 검색하고 수정하기 위한 트랜잭션
* Trade-Status Transaction : 최근 거래 상태의 정보를 검색하기 위한 트랜잭션
* Customer-Position Transaction : 고객계정이 보유한 전체 자산의 가치를 계산하기 위한 트랜잭션
* Broker-Volume Transaction : 브로커가 문의한 모든 거래에 대해서 잠재적인 전체 볼륨(거래규모)을 계산하기 위한 트랜잭션
* Security-Detail Transaction : 증권관련 모든 가능한 정보를 검색하기 위한 트랜잭션
* Market-Feed Transaction : 증권시장에 나오는 모든 증권의 최근 가격을 받아서 증권표시기의 개별 아이템에 대한 최근 가격으로 데이터베이스를 업데이트하기 위한 트랜잭션
* Market-Watch Transaction : 전체 증권의 현재 시장가격 대비 전일 마감된 증권의 시장자본 가치의 퍼센티지를 계산하기 위한 트랜잭션
* Data-Maintenance Transaction : 12개의 테이블을 기준으로 12분마다 테이블을 수정하는 단일 로직의 트랜잭션

- TPC-E의 SUT, DRIVER, AND NETWORK

TPC-E는 기술한 바와 같이 다음과 같은 현실세계의 OLTP 환경을 재구성한 것이다: 클라이언트들이 여러 가지 장비들을 이용하여 브로커사의 거래서비스에 접속하면, 서비스제공자 측에서 가

능한 서비스를 사용자에게 제시하고, 클라이언트는 원하는 서비스를 선택하여 데이터를 입력하고 서비스를 제공받는다. 다음 <그림 3-6>은 이러한 환경요소를 나타낸 것이다.



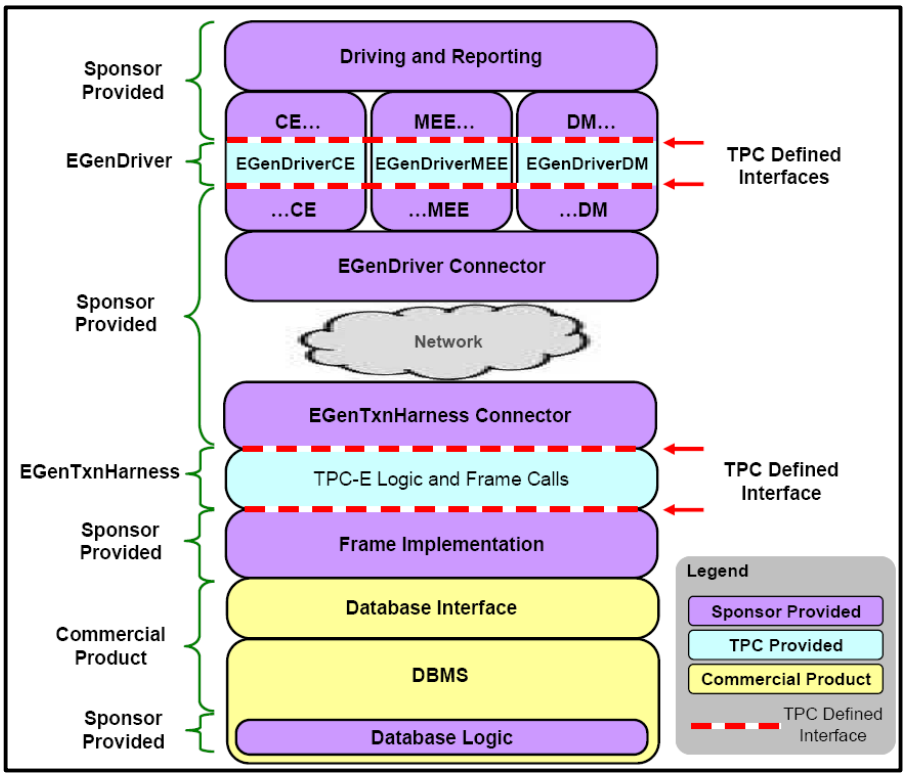
<그림18> 현실세계의 OLTP 환경

이러한 현실세계의 OLTP환경을 수행하기 위해서 TPC-E는 다음과 같은 기능적 구성요소를 정의하고 있다.

* Driving&Reporting : 테스트시스템 설치, 테스트실행 및 관리, 실행시간, 데이터 수집, 요약보고서를 생성하기 위해서 스폰서가 제공하는 기능이며, TPC가 정의한 인터페이스를 통해서 EGenDriver로 연결됨.
* CE : Customer-Emulator 설치, 관리 및 운영을 위해서 스폰서가 제공하는 기능이며, EGenDriverCE로 연결됨.
* MEE : Market-Exchange-Emulator 설치, 관리 및 운영을 위

해서 스폰서가 제공하는 기능으로 EGenDriverMEE로 연결됨.

* DM : 데이터유지보수 실행 및 관리를 위해서 스폰서가 제공하는 기능으로 EGenDriverDM으로 연결됨.
* EGenDriver : 프로그램 실행을 위해서 구현된 C++ 코드
* EGenDriver Connector : 내부의 EGenDriver가 생성한 데이터를 보내고 받기 위해서 스폰서가 제공하는 기능

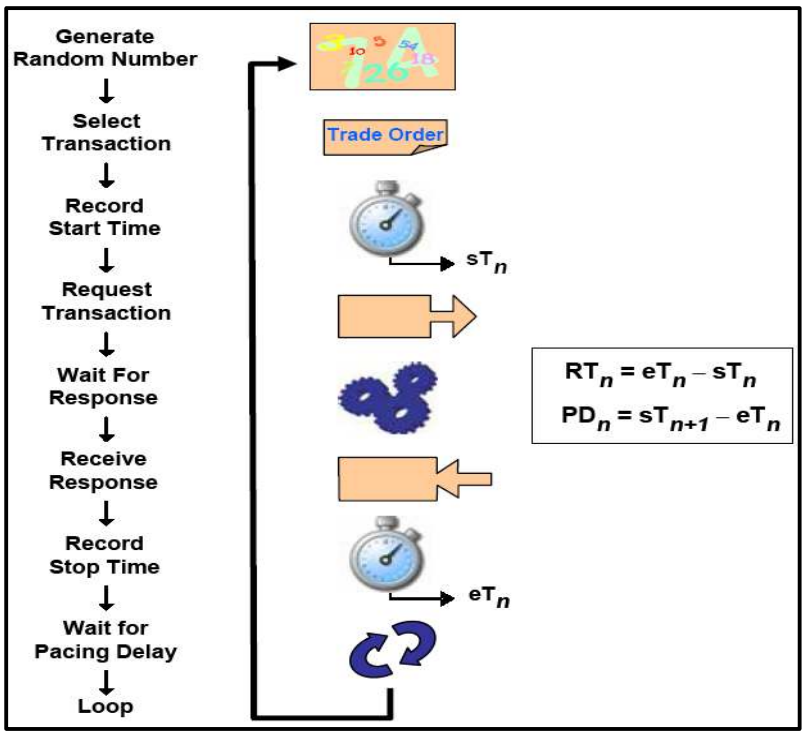


<그림18> TPC-E OLTP환경의 기능 구성요소

* EGenTxnHarness : 프로그램 실행을 위해서 구현된 C++ 코드

실제 프로그램 실행 측면에서 <그림18>의 기능구성요소를 요약하면 크게 테스트 실행을 위해서 필요한 데이터를 생성하여 전달하기 위한 DRIVER와, 전달받은 데이터를 처리하고 그 결과를 다시 DRIVER에 전달해 주는 역할을 담당하는 SUT(System Under Test) 로 나눌 수가 있다. EGenDriver와 EGenTxnHarness 를 포함하여 전체 TPC-E를 구현하기 위해서 TPC가 제공하는 소프트웨어 패키지를 EGen이라 한다.

- EGen 실행과 성능평가 기준



<그림19> 트랜잭션 처리과정과 시간계산 방법

EGen이라는 소프트웨어 패키지를 통해서 실행되며, EGenDriver에서 필요한 데이터를 생성하여 EGenTxnHarness 에 보내면, EGenTxnHarness에서 프로그램을 실행하여 그 결과를 다시 EGenDriver에 보내게 된다. EGenDriver에서 보내진 데이터가 EGenTxnHarness에서 실행되어 결과가 성공적으로 다시 EGenDriver에 전달되는 트랜잭션을 유효한 트랜잭션이라 하며, 이러한 유효한 트랜잭션이 1초에 몇 번이나 수행되는지를 성능평가 기준으로 삼으며, 유효한 트랜잭션의 처리과정을 기준으로 Response Time과 Pacing Delay Time을 측정한다. <그림19>는 트랜잭션이 처리되는 과정과, 이를 통해서 계산되는 Response Time과 Pacing Delay Time을 계산하는 방법을 나타낸 것이다.

* Response Time(RTn=eTn−sTn) : 유효한 트랜잭션을 기준으로 EGenDriver에서 데이터를 생성하여 EGenTxnHarness에 보내는 처음 순간부터 처리 결과가 EGenTxnHarness에서 EGenDriver로 돌아오는데 까지 걸리는 시간
* PacingDelay (PDn=sTn+1−eTn) : 트랜잭션의 처리결과가 EGenDriver에 도착된 이후 다음 트랜잭션 처리를 위해서 다음 데이터를 생성하여 EGenTxnHarness에 보낼 때 까지 소요되는 시간

단일 트랜잭션이 처리되는 과정은 위와 같으며, 개별 트랜잭션별로 정의된 믹스 율은 다음과 같다.

<표31> 트랜잭션별 믹스율

Security-Detail	14%	
Market-Feed	1%	Each Market-Feed contains entries for 10 completed Trade-Results.
Market-Watch	18%	
Total	**100%**	

- TPC-E 테스트

EGen이라는 소프트웨어 패키지를 통해서 실행되는 트랜잭션이 완전히 처리되기까지의 전체 시간을 Ramp-up, Steady-state, Ramp-down으로 구분하며,

* Ramp-Up Time : 테스트 시작에서 Steady-state까지의 시간
* Ramp-Down Time : Steady-state 종료에서 테스트 종료까지의 시간
* Steady-State Time : 트랜잭션이 지속적으로 처리되는 상태의 시간으로 최소 8시간 이상 지속적으로 성능을 제공할 수 있는 시간
* 구분된 시간 동안 증가 혹은 수정/변경된 데이터를 저장하기 위해서 전체 저장 공간을 다음과 같이 구분한다.
* Free Space : 저장 가능한 공간
* Growing Space : 트랜잭션이 처리됨에 따라 증가하는 데이터를 저장하기 위한 공간
* Fixed Space : 정적 정보와 인덱스를 저장하는데 사용되는 공간
* 테스트 데이터베이스는 성능에 영향을 주지 않고 8시간 동안 지속적으로 증가하는 객체를 처리할 수 있어야 하며, 하루 8

시간 증가한 내용을 저장할 수 있는 별도의 공간이 필요하다. 이를 구체적으로 정의하기 위해서 테스트가 시작되기 전의 저장 공간 상태와(Pre-Run Space), 테스트 이후의 Growing Space와 Free Space를 계산하여 테스트 후의 저장 공간의 상태를(Post-Run Space) 계산하고, 처리된 트랜잭션 수로 나눈 결과를 이용하여 그 차이(Run Space)를 계산한다.

* Daily Growth = Run - Space (per Trade - Results) * 3600 sec/hr*8hrs/day
* 8-Hour Log Growth = log Space(per Trade-Results) * 3600sec/hr * 8hrs/day

- TPC-E 트랜잭션 처리율과 데이터베이스 규모

데이터베이스 트랜잭션 처리는 CUSTOMER 테이블의 매 1,000 행을 기준으로 초당 처리되는 트랜잭션(=> Transaction-Per-Second-E: tpsE)으로 계산되며, Scale Factor(SF)에 따라서 다르게 정의할 수도 있다. 트랜잭션 처리율(=> Throughput Rating)은 TPC-E가 기준으로 제시한 성능평가 기준으로 사용된다. 따라서 최종 reported throughput rating의 단위는 tpsE로 한다.

이 때 Nominal Throughput의 80~100%가 유효하게 처리되면, 트랜잭션 처리율을 Measured Throughput으로 보며, 유효한 트랜잭션 처리수가 Nominal Throughput을 초과하는 경우(2% 이상 초과할 수 없음) Nominal Throughput을 트랜잭션 처리율로 본다. 따라서 유효한 트랜잭션 처리는 트랜잭션 처리율보다 2% 이상 클 수 없음을 의미한다.

- TPC-E 성능평가 기준

TPC-E 표준, TPC's Use Policies, Guideline 등 모든 TPC-E 관련 문헌에서 사용하는 성능평가 기준(Primary Metrics)은 다음에 따른다.

* Performance Metric : TPC-E Throughput Rating (단위: tpsE)
* Price/Performance Metric : 전체 3년 동안의 가격을 Throughput Rating으로 나눈 값 (단위: price/tpsE)
* Price/Performance Metric의 기준이 되는 Pricing 요소는 다음과 같다.
* Price of SUT
* 60-day period 동안의 모든 트랜잭션으로 생성되는 데이터, 인덱스를 저장하고 유지하는데 충분한 공간 확보를 위한 장비 가격
* SUT가 60일간의 필요 공간 요구를 충족시키기 위해서 필요한 Free Space를 확보하지 못한 경우, 추가적인 저장매체와 관련된 장비 가격
* 시스템 운영, 관리 및 유지보수를 위해서 필요한 부가제품 가격
* 애플리케이션을 개발하는데 필요한 부가제품에 대한 가격
* 기타 부가적인 운영요소 및 부가 소프트웨어에 대한 가격

5) Full Disclosure Report and sample

⊙ 벤치마크 실행결과 보고서

TPC-E 벤치마크 실행결과 보고서는 TP-E 표준에서 정하는 절차와 내용을 포함하여 작성하며, 보고서에 포함되어야할 내용은 다음과 같다.

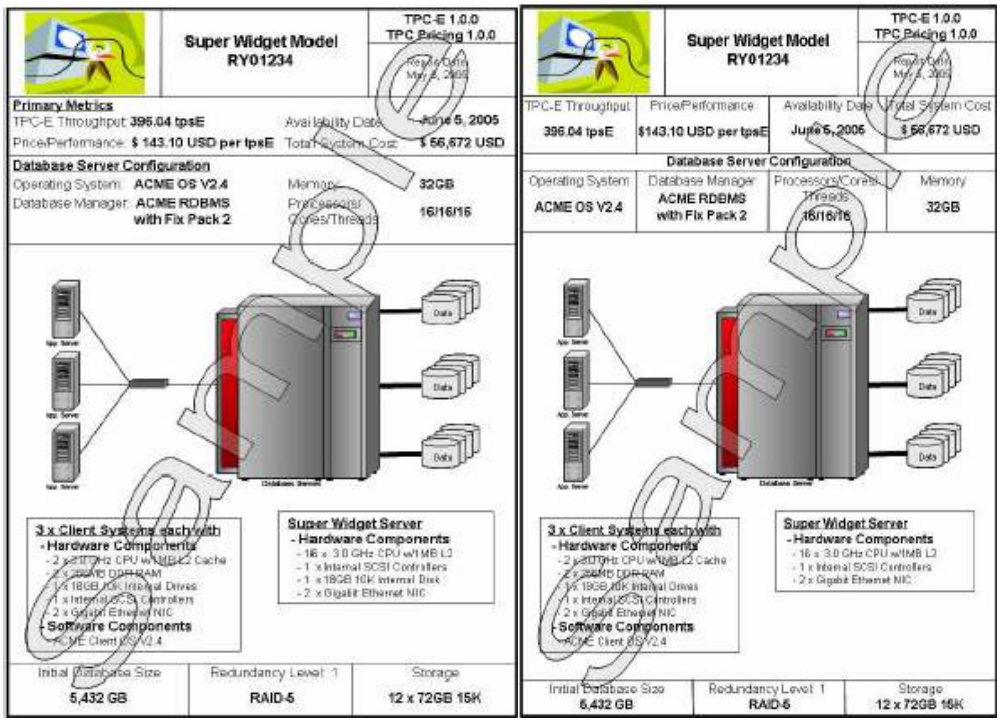
▪ 일반사항

- Test Sponsor's Full Disclosure Report는 TPC-E 표준에서 정한 순서와 항목과 일치
- TPC 실행요약문서는 Full Disclosure Report의 도입부를 포함
- TPC 실행요약문서에 포함될 수치데이터
  * 측정기간, 복구시간
  * 측정기간 내에 완수되는 트랜잭션 수
  * 초당 처리되는 트랜잭션 수
  * 모든 트랜잭션에 대해서 90% 이내의, 최소, 최대, 평균 응답시간
  * 최소, 평균, 최대 pacing delay
  * 개별 트랜잭션 타입별로 트랜잭션 믹스율
- 개별 트랜잭션의 Frame Implementation
- Benchmark Sponsor, Other Participating Companies에 대한 기술
- Customer Tunable 파라미터, 옵션

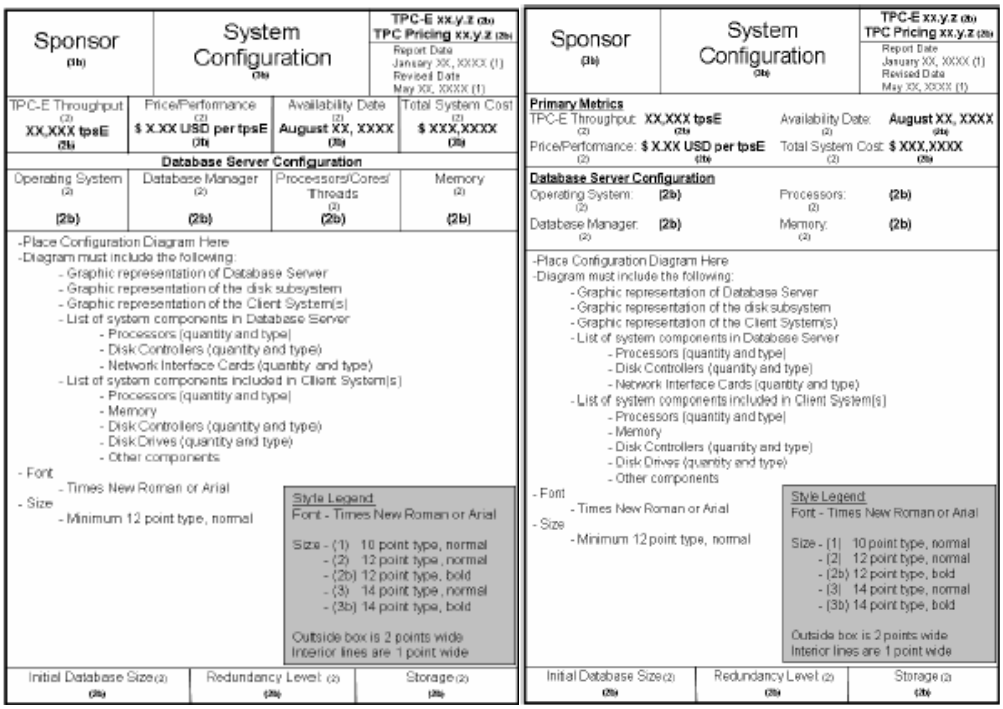
▪ Database Design, Scaling & Population Related Items
▪ Transaction related Items
▪ SUT, Driver, and Network related Items
▪ EGen related Items

- Performance Metrics and Response Time related Items
- Transaction and system properties related Items
- Pricing related Items

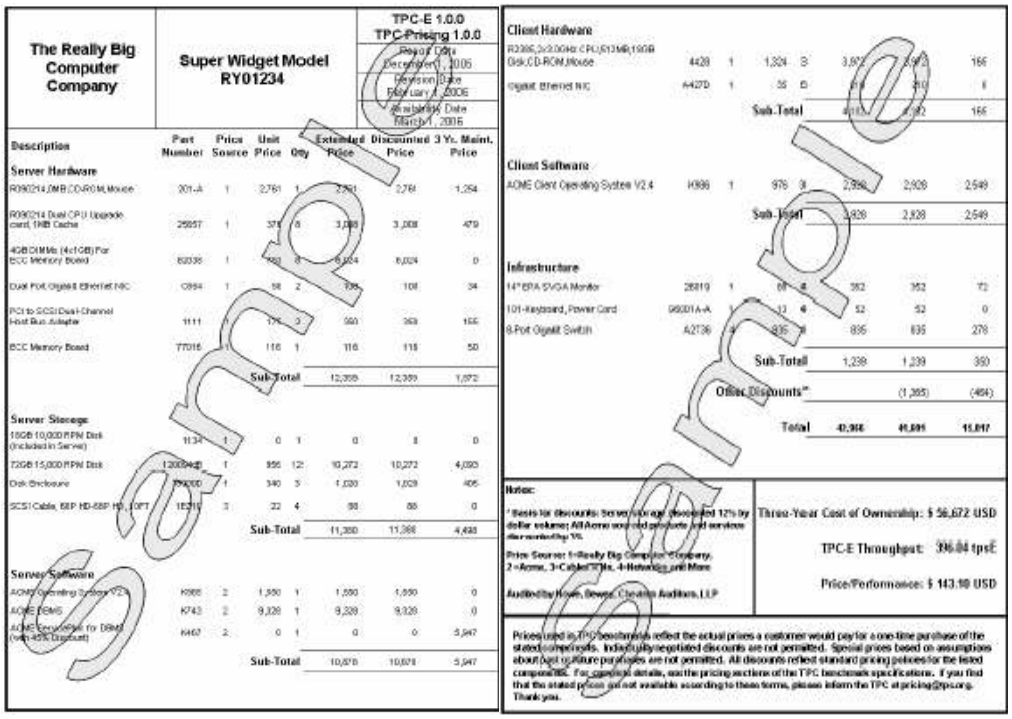
⊙ 벤치마크 실행결과 샘플



<그림20> 벤치마크 샘플(1)



<그림21> 벤치마크 샘플(2)



The Really Big Computer Company	Super Widget Model RY01234	TPC-E 1.0.0 TPC-Pricing 1.0.0 Report Date December 1, 2005 Revision Date February 1, 2006 Availability Date March 1, 2006

Description	Part Number	Price Source	Unit Price	Qty	Extended Price	Discounted 3 Yr. Maint. Price	3 Yr. Maint. Price
Server Hardware							
RX8014,2MB,CD-ROM,Mouse	201-A	1	2,761	1	2,761	2,761	1,354
RX8014 Dual CPU Upgrade card, 1MB Cache	25857	1	375	8	3,000	3,000	479
4GB DIMMs (4x1GB) For ECC Memory Board	80336	1	753	8	6,024	6,024	0
Dual Port Gigabit Ethernet NIC	C894	1	58	2	116	116	34
PCI to SCSI Dual-Channel Host Bus Adapter	1111	1	175	2	350	350	155
ECC Memory Board	77016	1	116	1	116	116	50
				Sub-Total	12,369	12,369	1,072
Server Storage							
18GB 10,000 RPM Disk (Included in Server)	1134	1	0	1	0	0	0
72GB 15,000 RPM Disk	1200HB	1	856	12	10,272	10,272	4,093
Disk Enclosure	RA2000	1	340	3	1,020	1,020	405
SCSI Cable, 68P HD-68P HD, 10FT	1E210	3	22	4	88	88	0
				Sub-Total	11,380	11,380	4,498
Server Software							
ACME Operating System V2.x	K888	2	1,550	1	1,550	1,550	0
ACME DBMS	K743	2	9,328	1	9,328	9,328	0
ACME Server Agent for DBMS (with 40% Discount)	K467	2	0	1	0	0	5,947
				Sub-Total	10,878	10,878	5,947

Client Hardware							
R3366,2x3.0GHz CPU,512MB,160GB Disk,CD-ROM,Mouse	4428	1	1,324	3	3,972	3,972	166
Gigabit Ethernet NIC	A427D	1	35	0	0	0	0
				Sub-Total	4,032	4,032	166
Client Software							
ACME Client Operating System V2.4	K886	1	976	3	2,928	2,928	2,549
				Sub-Total	2,928	2,928	2,549
Infrastructure							
14" EPA SVGA Monitor	28819	1	88	4	352	352	72
101-Keyboard, Power Cord	99301A-A	1	13	4	52	52	0
8 Port Gigabit Switch	A2736	1	835	1	835	835	278
				Sub-Total	1,239	1,239	350
				Other Discounts**		(1,265)	(464)
				Total	40,906	40,895	15,017

Notes:

* Basis for discounts: Server hardware discounted 12% by dollar volume; All Acme sourced products and services discounted 5%

Price Source: 1=Really Big Computer Company, 2 =Acme, 3=Cables "R"Us, 4=Networks and More

Audited by Howe, Bowes, Cheatem Auditors, LLP

Three-Year Cost of Ownership: $ 56,672 USD

TPC-E Throughput: 396.04 tpsE

Price/Performance: $ 143.10 USD

Prices used in TPC benchmarks reflect the actual prices a customer would pay for a one-time purchase of the stated components. Individually negotiated discounts are not permitted. Special prices based on assumptions about past or future purchases are not permitted. All discounts reflect standard pricing policies for the listed components. For complete details, see the pricing sections of the TPC benchmark specifications. If you find that the stated prices are not available according to these terms, please inform the TPC at pricing@tpc.org. Thank you.

Numerical Quantities Summary				
Reported Throughput:				396.04 tpsE
Response Times (in seconds)	**Minimum**	**Average**	**90th %ile**	**Maximum**
Broker-Volume	0.000	0.251	0.441	1.261
Customer-Position	0.012	0.333	0.559	1.449
Market-Feed	0.221	0.351	0.662	0.872
Market-Watch	0.324	0.562	0.731	2.011
Security-Detail	0.220	0.261	0.671	1.563
Trade-Lookup	0.551	0.374	0.704	2.111
Trade-Order	0.011	0.252	0.452	0.934
Trade-Result	0.202	0.401	0.481	1.932
Trade-Status	0.061	0.162	0.300	0.552
Trade-Update	0.334	0.481	0.951	0.931
Data-Maintenance	0.402	0.431	0.662	11.171
Transaction Mix (in percent of total transactions)			**Transaction Count**	**Mix**
Broker-Volume			266,839	4.90
Customer-Position			3,392,162	13.00
Market-Feed			266,154	1.00
Market-Watch			5,417,394	19.00
Security-Detail			4,277,316	14.00
Trade-Lookup			2,281,236	8.00
Trade-Order			2,880,060	10.10
Trade-Result			2,851,544	10.00
Trade-Status			5,703,088	19.00
Trade-Update			570,309	2.00
Data-Maintenance			120	N/A
Test Duration and Timing				
Ramp-up Time (hh:mm:ss)			00:15:40	
Measurement Interval (hh:mm:ss)			2:00:00	
Business Recovery Time (hh:mm:ss)			2:35:40	
Total Number of Transactions Completed in Measurement Interval			28,515,441	

<그림22> 벤치마크 샘플(3)

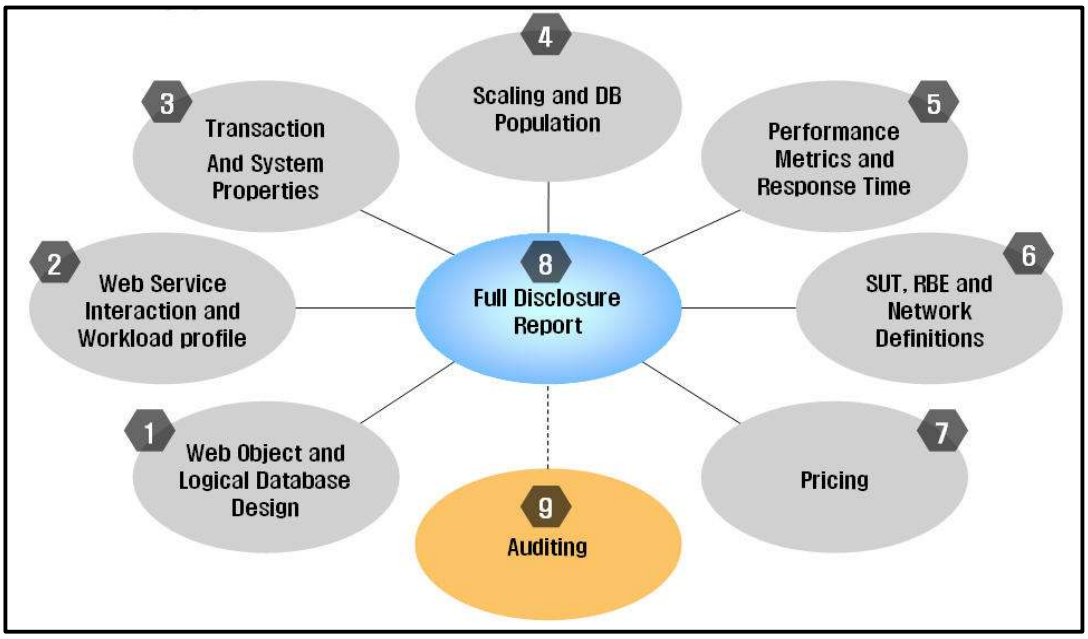
6) TPC-App 벤치마크 개요

TPC(Transaction Performance Processing Council)에서 TPC-W를 대체하기 위하여 새롭게 제안한 TPC-App는 Application Server와 Web Service의 성능 측정을 위한 벤치마크 표준이다.

TPC-App 벤치마크는 데이터베이스 서버, 웹 서버, 여타 애플리케이션 서버들과 데이터를 교환하는 전형적인 전자상거래 작업을 서버가 중간에서 얼마나 잘 수행하는가를 측정하게 되고, 점수는 1초당 처리한 웹 서비스의 수로 처리가 된다.

24시간 365일 운영되는 B2B 서비스의 액티비티들을 시뮬레이션한 것으로 주문을 하거나 온라인 상품 카탈로그를 보는 등의 인터넷 쇼핑몰을 시나리오 기반으로 TPC-App 벤치마크의 워크로드는 구성되어있다.

TPC-APP 벤치마크의 규격은 다음과 같이 구성되어 있다.



<그림23> TPC-App 벤치마크 규격의 구성

TPC-App는 2개의 성능 측정 기준이 있으며, 첫 번째는 어플리케이션 서버 시스템마다의 SIPS(Web Service Interactions per second)이고 두 번째는 전체 테스트 설정(SUT(System Under Test))에서의 통합 SIPS이다.

⊙ TPC-App 벤치마크 규격 내역

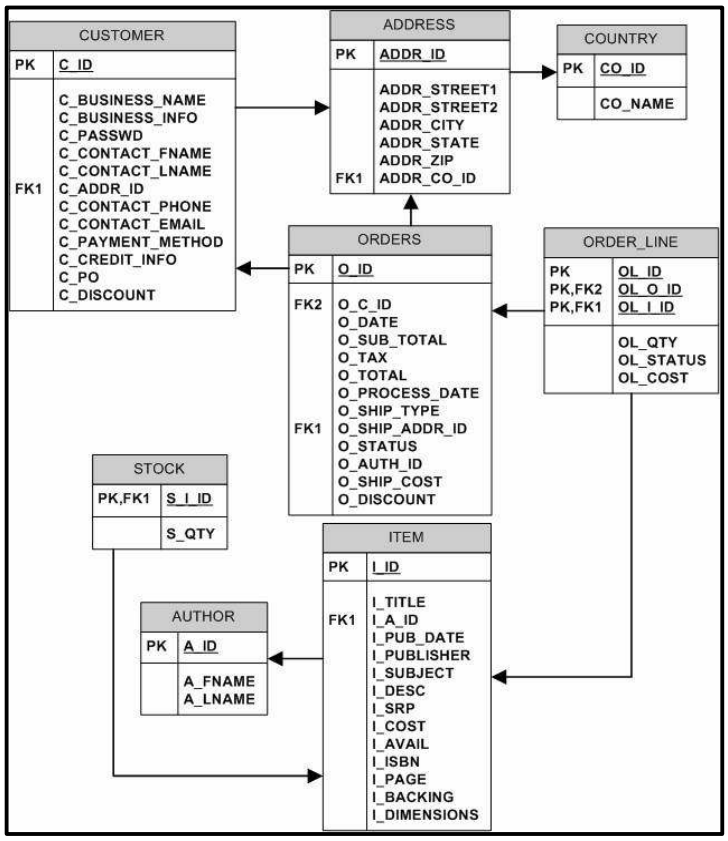
Clause 1 - Web Object and Logical Database Design

1장에서는 기본적인 용어 정의와 DB의 엔티티 간의 관계에 대한 정의가 규정되어 있다.

- 용어

- SUT(System Under Test) : System Under Test; 어플리케이션의 모든 컴포넌트들이 시뮬레이션 되는 네트워크와 연결되어진 어플리케이션 서버와 DB 서버
- RBE(REMOTE BUSINESS EMULATION) : Remote Business Emulator; TPC-App 워크로드(workload)에 의해 동작되는 소프트웨어 컴포넌트
- EB : Emulated Business client; 네트워크로 연결된 SUT(System Under Test)에 대하여 HTTP나 TCP/IP를 통해서 SOAP 메시지를 주고받는 Web Service를 거쳐서 통신하는 클라이언트를 에뮬레이션 하는 객체(e.g. 프로세스나 쓰레드)
- Web Service Interaction : Active EB와 SUT(System Under Test)사이에 주고받는 SOAP 메시지의 순서; Application이 XML로 된 데이터를 Web Service와 주고받는 것

▪ Logical Database



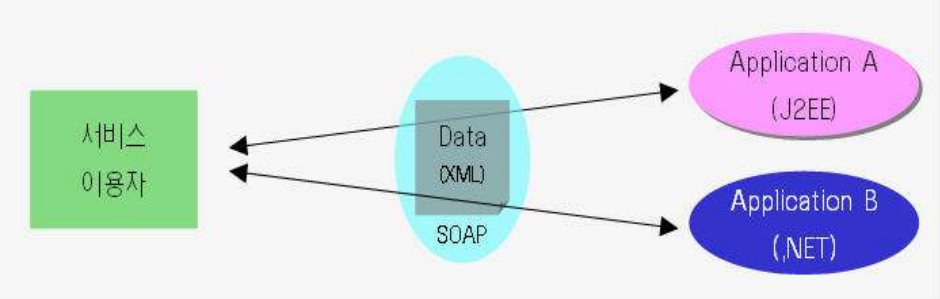
<그림24> Logical Database

Clause 2 – Web Service Interactions and Workload Profile

2장에서는 TPC-App 벤치마크에서 사용되는 Web Service Interaction의 종류와 정의가 규정되어 있으며, Web Service Interaction으로 구성되는 Workload의 프로필이 규정되어 있다.

Web Service Interaction은 SUT(System Under Test)상에서 동작하는 RBE(Remote Business Emulation)는 WS-I BP 1.0 규격을

따르며, Web Service Interaction의 WS-I BP 1.0 규격 준수 여부를 테스트하기 위해서는 WS-I 웹 사이트(www.ws-i.org)를 참고하여 확인한다.



<그림25> Web Service

Web Service는 인터넷상에서 단일 기업 또는 다수 기업간에 존재하는 기존 컴퓨터 어플리케이션 프로그램을 XML(eXtensible Markup Language)을 기반으로 상호 연동시키는 표준화된 소프트웨어 기술로서 TPC-App 벤치마크에서 사용되는 Web Service의 종류와 정의는 다음과 같다.

- New Customer Web Service
  - 새로운 고객과 관련된 어플리케이션으로서 고객의 등록, 새로운 고객 ID 생성 등으로 구성됨
- Change Payment Method Web Service
  - 고객이 결제방식을 변경할 때 사용하는 Web Service Interaction임
- Create Order Web Service
  - DB에 새로운 주문정보를 생성하고 주문을 배송하는 배송시스템에 새로운 주문이 들어왔음을 알려주는 Web Service Interaction임

▪ Shipping Process

– 배송 프로세스는 Create Order Web Service Interaction에서 보내진 메시지와 배송 큐를 통해 재고관리프로세스를 처리한다. 주문된 상품의 재고 파악, 배송 상태에 따른 주문 상태 업데이트, 배송 라벨 확보, 상품 재고가 필요한 경우 재고 관리 큐에 상품 입고에 대한 정보 업데이트 등의 배송을 위한 비즈니스 로직을 수행한다.

▪ Stock Management Process

– 재고 관리 큐를 통해서 배송 프로세스에서 보내진 메시지를 받아 처리한다.

▪ Order Status Web Service

– 고객의 마지막 N번째 주문에 대한 정보를 처리한다.

▪ New Product Web Service

– 최근에 변경된 아이템의 리스트 정보를 리턴하며, 신제품 Web Service Interaction은 Test Run을 통해서 그 결과를 변경함

▪ Product Detail Web Service

– 아이템에 대한 요청에 대하여 아이템의 상세정보를 리턴한다.

▪ Change Item Web Service

– DB의 Item 엔티티의 I_PUB_DATE 필드를 갱신하는 Web Service

TPC-App의 Workload의 Profile은 <표32>와 같다.

<표32> Workload의 Profile

Web Service Interaction Type	Required Mix
New Customer	1 %
Change Payment Method	5 %
Create Order	50 %
Order Status	5 %
New Products	7 %
Product Detail	30 %
Change Item	2 %

Web Service는 WS-I BP 1.0 규격에 호환되도록 구현되어야 하고, WS-I(Web Services Interoperability Group) BP(Basic Profile) 1.0에서 제공하는 분석도구를 이용하여 로그를 남기고 성능측정에 적용된다.

Clause 3 - Transaction and system properties

Transaction은 DB를 변경하는(Insert, Delete, Update 등) 작업의 한 단위로서 관련된 엔티티들이 모두 동의할 때 commit이 이루어지고, 동의하지 않으면 rollback이 이루어지며 다음과 같은 예제가 가능하다.

- N은행 현금인출기에 인출을 할 때 계좌의 비밀번호 확인, 잔금 확인, 인출 확인까지 정상적으로 이루어지면 인출이 완료되고(commit)

▪ 이 중 하나라고 오류가 발생하면 처음부터 다시 시작해야 함(rollback)

DB에 접근하는 프로그램으로써 Transaction은 ACID 속성을 가져야 하고 그 정의는 다음과 같다.

▪ Atomicity (원자성)

– All or nothing

– 한 트랜잭션이 커밋(commit)되면 모든 효과가 유지되고 한 트랜잭션이 중단되면 모든 효과가 취소됨

– 예를 들어, 개체 이름을 바꾸는 경우 새 이름이 만들어지고 이전 이름이 삭제되거나(커밋) 또는 아무 변화도 발생 안함(중단)

– Atomicity Test는 Create Order Web Service를 통해서 진행

▪ Consistency (일관성)

– 수행 전후의 결과가 논리적인 착오가 발생하지 않도록 보장

– 트랜잭션은 시스템 상태의 올바른 변환이 목적이고 트랜잭션은 상태를 일관되게 보존함

▪ Isolation (독립성)

– 순차적으로 수행된 것처럼 DB상태를 보장

– 동시적 트랜잭션은 완료되지 않은 다른 트랜잭션의 업데이트로부터 격리되고 이러한 업데이트는 일관된 상태를 만들지 않게 됨

▪ Durability (지속성)

– 일단 Transaction이 완료되면 그 결과를 지속적으로 보장

– 트랜잭션이 일단 커밋 되면 시스템에 오류가 발생하더라도 그 효과는 지속됨

Clause 4 - Scaling and Database Population

TPC-App 벤치마크의 DB Scaling은 설정된 EB의 숫자에 의해 정해지며, ITEM 테이블의 cardinality는 NUM_ITMES에 의해 정해지고, STOCK 테이블, AUTHOR 테이블의 cardinality는 ITEM 테이블의 function에 의해 정해지게 된다.

TPC-App 벤치마크의 DB cardinality는 <표33>과 같다.

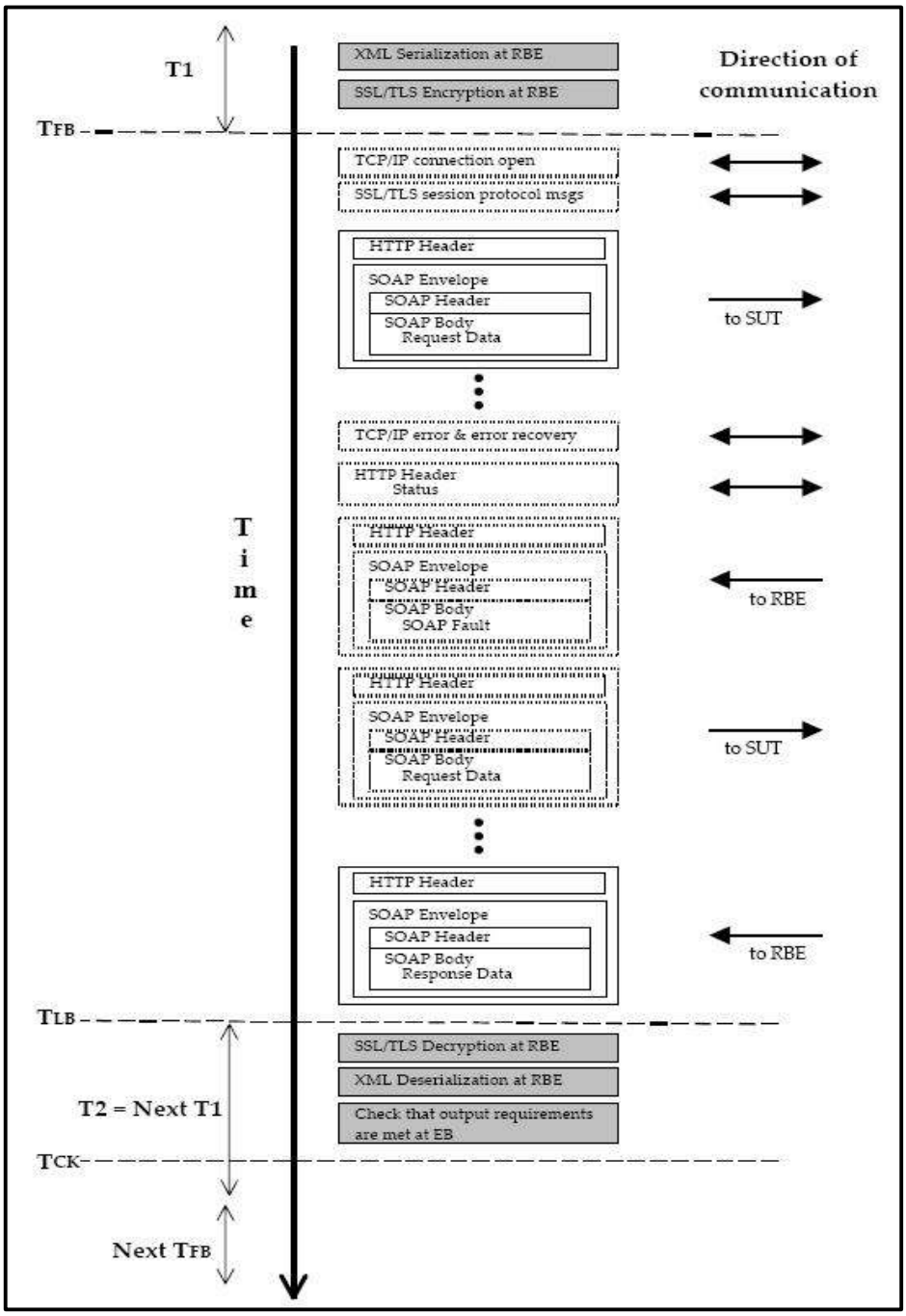
<표33> TPC-APP 벤치마크의 DB cardinality

Table Name	Cardinality (in rows)	Typical Row Length (in bytes)	Typical TableSize (in bytes)
CUSTOMER	192 * (number of Configured EBs)	575	10,752k
COUNTRY	92	54	4.85 k
ADDRESS	1.4 * CUSTOMER	154	4,139 k
ORDERS	10 * CUSTOMER	126	24,192 k
ORDER_LINE	5.5 * ORDERS	56	59,136k
AUTHOR	.25 * NUM_ITEMS	49	1,225k
ITEM	NUM_ITEMS	789	78,900k
STOCK	NUM_ITEMS	18	1,800k
Note 1: 테이블 사이즈는 100개의 EB를 산정한 사이즈 Note 2: row의 길이와 테이블 사이즈는 요구사항이 아니고, 저장공간과 액세스 비용에 대해서는 고려되지 않았음			

Clause 5 - Performance Metrics and Response Time

TPC-App 벤치마크의 지표는 어플리케이션 서버당 SIPS, SIPS 합계, SIPS당 비용, 유효한 날짜 등이다.

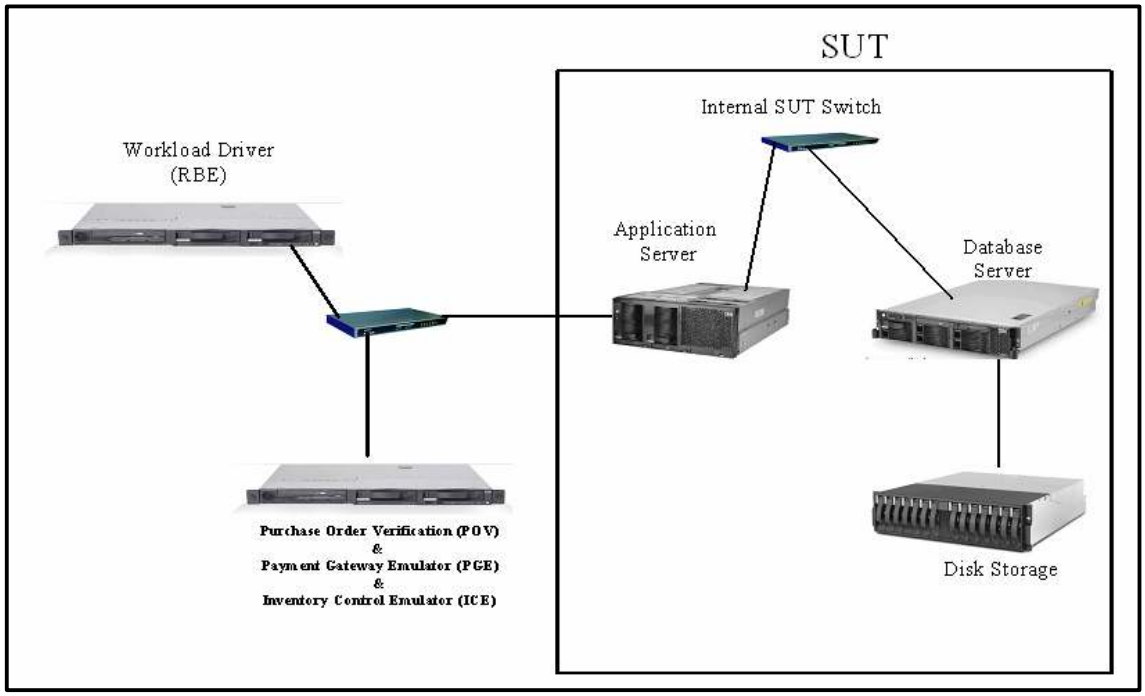
SIPS는 Web Service Interaction Per Second의 약자로서 초당 Web Service Interaction의 수를 의미하며, Web Service Interaction의 수를 확인하기 위하여 SIRT(Web Service Interaction Response Time)을 이용하게 되고, SIRT는 <그림 26>에서 T2-T1과 같이 측정된다.



<그림26> Performance Metrics and Response Time

Clause 6 - SUT, RBE and Network Definitions

TPC-App 벤치마크를 운용하기 위한 시스템 환경은 다음 그름과 같이 구성되어야 한다. 일반적인 인터넷 B2B 쇼핑몰과 유사한 환경을 구축하기 위해 전자결제, 재고 관리 등의 외부 시스템까지 구성된다.



<그림27> SUT, RBE and Network Definitions

- Remote Business Emulator(RBE)

  - RBE(REMOTE BUSINESS EMULATION)는 TPC-App 워크로드(workload)를 움직이는 소프트웨어 컴포넌트

  - RBE(REMOTE BUSINESS EMULATION)는 SUT(System Under Test)에 대한 Web Service 이슈인 비즈니스를 에뮬레이션

함

- Business Session Length(BSL)

– 비즈니스 세션이 연결된 동안 Active EB에 의해 동작하는 Web Service Interaction의 개수

– Customer ID, Business Name, Customer Password

–비즈니스 세션이 연결되면, Customer ID를 선택하고, 이름과 패스워드를 선택하고 Web Service Interaction을 수행한다.

- System Under Test(SUT)

– SUT(System Under Test)는 시뮬레이션 되는 어플리케이션의 모든 컴포넌트를 포함

– SUT(System Under Test)에는 네트워크 연결, 어플리케이션 서버 시스템, DB 서버 시스템을 모두 포함

– SUT(System Under Test)는 다음과 같은 작업을 수행한다.

* 상용화된 인터페이스를 통한 모든 DB 액세스
* TCP/IP 소켓 통신에 의한 컴포넌트간의 커뮤니케이션
* 모든 어플리케이션은 Web Service Interaction으로 구현

- Purchase Order Validation(POV)

– POV는 신규 고객이나 신규 고객이 지불 수단을 변경하는 경우 고객의 신용을 승인하는 외부시스템을 대표함

– POV는 SUT(System Under Test)에 포함되지 않는 외부시스템

- Payment Gateway Emulator(PGE)

– PGE는 Create Order Web Service Interaction의 한 부분으로 신용카드 결제의 승인을 담당하는 외부시스템을 대표함

– PGE는 SUT(System Under Test)에 포함되지 않는 외부시스템

- Inventory Control Emulator(ICE)

– ICE는 부가적인 아이템 재고 요청에 받아 처리하는 외부시스템을 대표함

– ICE는 SUT(System Under Test)에 포함되지 않는 외부시스템

- Shipment Notification Emulator(SNE)

– SNE는 Fedex, UPS 같은 외부 배송업체를 대표함.

– 외부 배송업체는 배송상품의 배송 라벨 및 배송상태 조회번호로 대표되는 이미지가 포함되는 배송주소를 이용함

– SNE는 SUT(System Under Test)에 포함되지 않는 외부시스템

Clause 7 – Pricing

Pricing 관련 내용은 TPC Pricing 규격을 참조하며, 8시간 동안 처리량을 유지해야 하는(Continuous Operation Requirement)저장장치와 다른 동작을 위한 리소스는 SUT(System Under Test)에 포함되어 가격산정에 반영되고,

충분한 저장장치와 잠재적인 다른 리소스도(Archive Operation Requirement) 가격 산정에 반영되어야 한다.

TPC Pricing 규격에 포함되지 않은 추가적인 요구사항은 다음과 같다.

- SUT(System Under Test) 가격
- EB(Emulated Business), 네트워크 비용
- RBE(REMOTE BUSINESS EMULATION), PGE, ICE, POV 비용
- 전원공급장치 비용
- 부가적으로 사용되는 S/W

Clause 8 – Full Disclosure Report

TPC-App 벤치마크 수행이 완료되면 이를 요약하고 정리하는 최종보고서가 산출물로서 나오게 되며, Overview, Pricing,

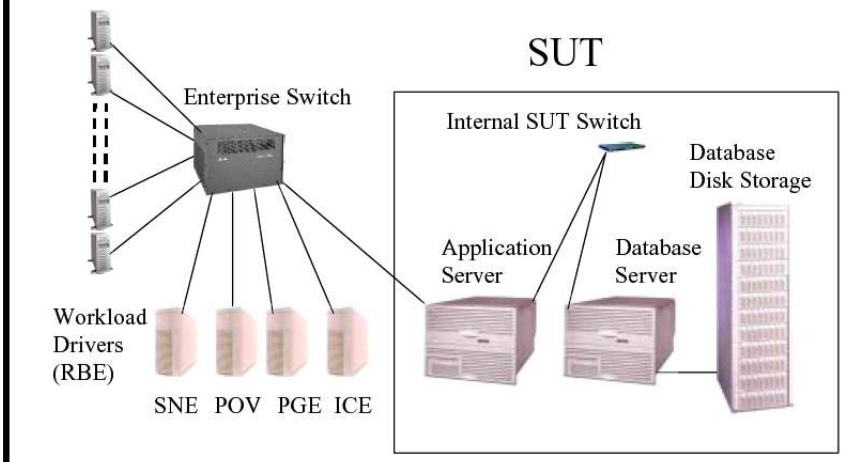
Numerical Quantities로 구성된다.

- Overview page 예제

Some Logo	Mothra 2000	TPC-App Rev. 1.1.1 Report Date 7/4/2003

SIPS per App Server System	Total SIPS
10,439.6 SIPS	10,439.6 SIPS
Price Performance	**Availability Date**
80.42 $USD	July 10, 2003

Catgegory	Application Server Systems	Average Response Time
Clustered	10	0.5
Total Cost	**End Users**	**Application Server**
839,506 $USD	550	Excalibur AppMaster XII
Managed Runtime	**Distributed Transaction Mgr**	Web Server
Merlin 3.0	Pivotman	Ramoth Webmaster II
Database Mgr	**Message Queueing Product**	**Other Software**
Mrs Murphy's DBMS	Message Master II	Bambi's .NET MISL Optimizer



Function	System	Num Systems	Operating System	Processors	Memory
Application Server	Mothra 2000	1	Exec 8	16 Thunderbolt IV 900 Mhz/2MB L2	8 GB
Database Server	Mothra 2000	1	Exec 8	16 Thunderbolt IV 900 Mhz/2MB L2	32 GB

<그림28> Overview page

- Pricing Page 예제

Some Logo	Mothra 2000	TPC-App Rev. 1.1.1 Report Date 7/4/2003

SIPS per App Server System	Total SIPS
10,439.6 SIPS	10,439.6 SIPS
Price Performance	**Availability Date**
80.42 $USD	July 10, 2003

Description	Part Number	Price Source	Unit Price	Qty	Extended Price	Discounted Price	3-yr Maint
Application Server Hardware							
Mothra 2000 Server	MO716-DDN	1	$100,000	1	$100,000	$100,000	$24,000
CPU:4x 700MHz Thunderbolt IV, 2MB L2	MO747-2MB	1	$22,000	4	$88,000	$88,000	
MEM: 16MB L3 Cache	MO714-SPD	1	$15,000	4	$60,000	$60,000	
MEM: 4GB Memory, SDRAM	RAM512-4G	1	$24,000	2	$48,000	$48,000	
ACC: High Availability Package	MO7001-HAP	1	$25,000	1	$25,000	$25,000	
ACC: PCI Module, 3.3v	PCI3-MOD	1	$400	4	$1,600	$1,600	
ETHERNET: 10/100Mbit/sec, PCI 32-bit	E1010052-PCI	1	$135	2	$270	$270	
ETHERNET: 10/100Mbit/sec, Dual Port	E1010053-PCI	1	$300	2	$600	$600	
ETHERNET: 1000Mbit/sec, PCI 64-bit	E100051-P64	1	$850	2	$1,700	$1,700	
CTRL: Fibre Channel HBA, 64-bit PCI	FCH201-P64	1	$1,700	2	$3,400	$3,400	
CTRL: SCSI, 2 Channel LVD, 64-bit PCI	PCI50233-P64	1	$765	2	$1,530	$1,530	
COMM: 56K LAN Modem	MDM56	1	$275	1	$275	$275	
DISK: 18GB Drive, 10Krpm, SCSI LVD	H8110	1	$600	2	$1,200	$1,200	
PWR: AC Entry Module, 3 PH	APA1000	1	$2,000	2	$4,000	$4,000	
Mothra 2000 Value Add Software	MO700011-N	1	$5,000	1	$5,000	$5,000	$324
Mothra 2000 SW Update Subscription	S200016-L	1	$16,200	3	$48,600	$48,600	
3-Yr. Cmprhnsv-Gold Svc Wrrnty Upgrd	WAD7	1	$22,992	1	$22,992		$22,992
MONITOR: 15-inch Color	VGA2100-P	1	$300	2	$600	$600	
			Sub-Total		$412,767	$389,775	$47,316
Database Server Hardware							
Mothra 2000 Server	MO716-DDN	1	$100,000	1	$100,000	$100,000	$24,000
CPU:4x 700MHz Thunderbolt IV, 2MB L2	MO747-2MB	1	$22,000	1	$22,000	$22,000	
MEM: 16MB L3 Cache	MO714-SPD	1	$15,000	4	$60,000	$60,000	
MEM: 4GB Memory, SDRAM	RAM512-4G	1	$24,000	2	$48,000	$48,000	
ACC: High Availability Package	MO7001-HAP	1	$25,000	1	$25,000	$25,000	
ACC: PCI Module, 3.3v	PCI3-MOD	1	$400	4	$1,600	$1,600	
ETHERNET: 10/100Mbit/sec, PCI 32-bit	E1010052-PCI	1	$135	2	$270	$270	
ETHERNET: 10/100Mbit/sec, Dual Port	E1010053-PCI	1	$300	2	$600	$600	

<그림29> Pricing Page

- Numerical Quantities Summary Page

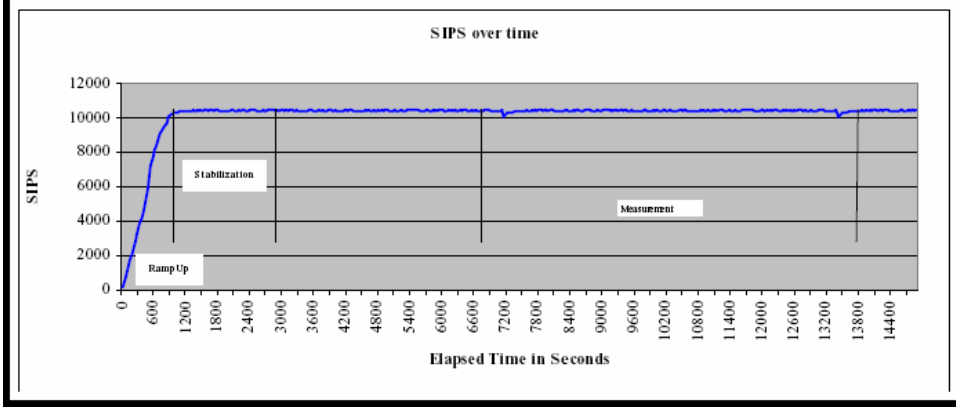
Some Logo	Mothra 2000	TPC-App Rev. 1.1.1 Report Date 7/4/2003

SIPS per App Server System	Total SIPS
10,439.6 SIPS	10,439.6 SIPS
Price Performance	**Availability Date**
80.42 $USD	July 10, 2003

Numerical Quantities Summary

SIPS Interaction Summary

Interaction Name	Mix%	Interactions per sec	Count	Min Resp Time	Avg Resp Time	Max Resp Time	90th Percentile Resp Time
New Customer	1.01	2.02	14544	1.34	1.55	2.65	1.60
Change Payment Method	4.99	9.98	71856	1.50	1.88	2.99	1.93
Create Order	50.15	100.3	722160	2.03	2.50	2.66	2.60
Order Status	4.91	9.02	70704	.033	0.40	0.43	0.50
New Products	7.07	14.14	101808	0.42	0.55	0.66	0.63
Product Detail	29.85	59.7	429840	0.78	0.85	0.97	0.83
Change Item	2.02	4.4	31680	0.56	0.66	0.78	0.75



<그림30> Numerical Quantities Summary Page

Clause 9 - Auditing

TPC-App 벤치마크 결과는 TPC가 인증한 감사 인에 의하여 감사가 이루어져야 하며, TPC-App 벤치마크 규격에는 감사가 이루어지는 분야와 사항에 대하여 정의되어 있다.

7) SPECWeb2005

⊙ SPECWeb2005(웹 서버 벤치마크) 개요

SPECWeb2005는 웹 환경에서 클라이언트들에게 서비스를 제공하는 웹 서버를 벤치마크 하기 위한 SPEC의 벤치마크 도구이다. 이러한 벤치마크 도구는 인터넷의 보급과 발달로 다양한 서비스가 웹상에서 가능하게 됨에 따라, 다양한 성능을 가진 웹서버를 필요로 하게 되고 성능이 좋은 웹 서버를 도입함으로써 빠르고 효율성이 좋은 서비스를 제공 받기를 원하고 있다. 이와 같은 요구는 웹 서버를 서비스 영역과 이용 분야를 구체화하여 벤치마크 함으로써 보다 성능이 좋은 웹 서버를 이용하기 위한 측정 기준을 필요로 하고 있다.

벤치마크란 서로 다른 컴퓨터 기종들의 성능을 객관적으로 평가하는 표준 평가 도구이다. 이러한 벤치마크 도구 중 웹 서버의 성능을 평가하는 대표적인 프로그램으로 SPECWeb이 있다. SPECWeb은 다양한 버전으로 시대에 발맞추어 새로운 버전이 연구중에 있으며, 가장 최근에는 SPECWeb2005를 발표 하였다. 이는 서로 다른 웹 서버를 객관적으로 평가하기 위한 표준 프로그램으로 기존의 SPECWeb99에 이은 최신버전의 프로그램이다.

SPEC은 1988년 11월에 설립된 이래 HP, 썬마이크로시스템즈

등 주요 시스템업체가 컨소시엄 형태로 참여하고 있으며 시스템 성능 테스트에 있어서 권위를 인정받고 있으며, 또한 CPU 성능 테스트에 대해 가장 권위 있는 벤치마크 테스트로 주목받고 있다.

⊙ SPECWeb2005의 발달 과정

SPECWeb2005는 다음과 같은 기존 버전들로부터 발달 되었다.

▪ SPEC95

SPEC95는 SPEC에서 발표한 벤치마크로 SPECint95와 SPECrate_int95, SPECfp95, SPECrate_fp95 등으로 구분되어 있다.

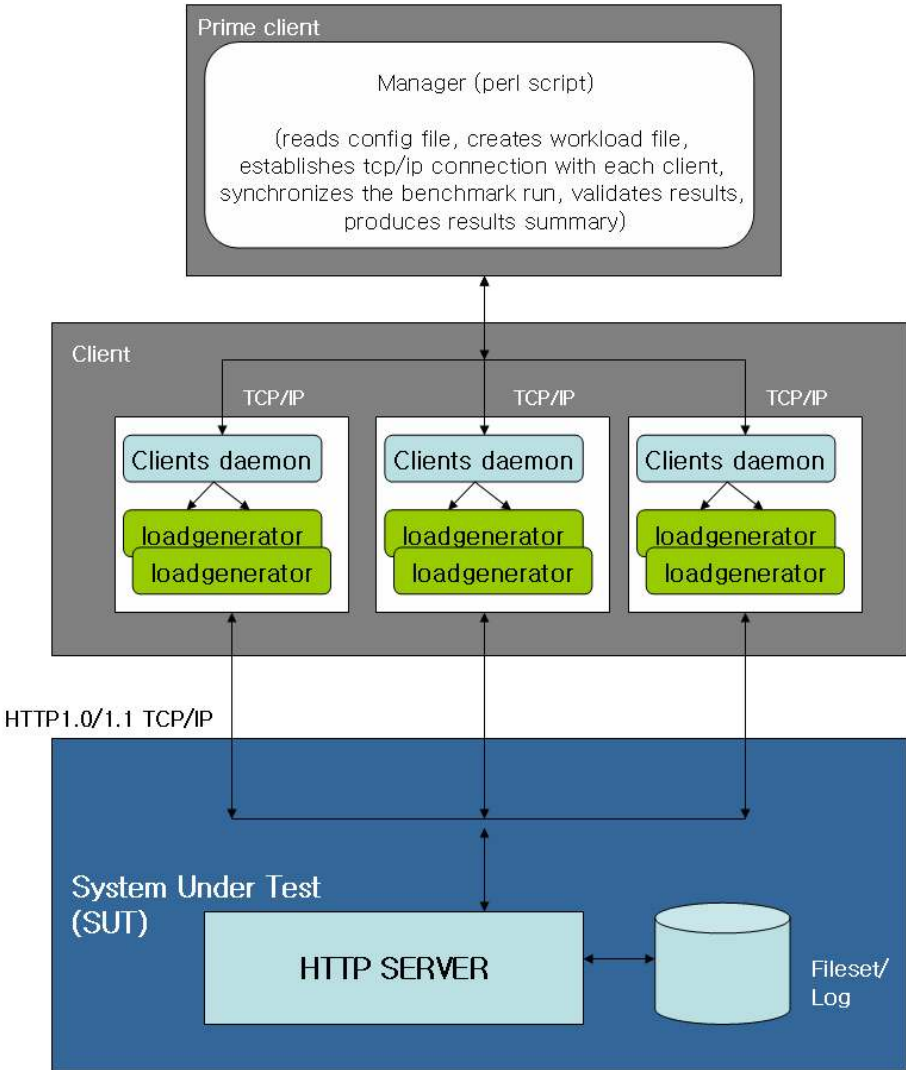
▪ SPECWeb96

SPECWeb96은 SPEC이 개발한 벤치마킹 방법으로서 벤치마킹을 위한 환경은 서버 H/W, 웹 서버 S/W, 클라이언트로 구성된다.

▪ SPECWeb96 평가 방법

평가방법으로는 클라이언트들에서 SPECWeb96 S/W를 사용 서버 측의 부하를 발생하여 웹 서버에 접근하는 Web Browser를 시뮬레이션 하는데 작업부하는0-1KB를 35%, 1KB-10KB를 50%, 10KB-100KB 14%, 그리고 100KB-1MB를 1%로 발생시키며, NCSA, CommerceNet, Netscape등의 사이트에 접근하는 Client들에 대한 로그를 기반으로 요구하는 파일 사이즈, 빈도 등을 고려하고 있다. 클라이언트의 수는 서버를 포화상태로 이르게 할 수 있는 구성이면 되고, 자유로이 선택하며 서버 처리능력이 포화상태에 이르고 응답시간이 급격히 느려질 때까지 부하를 점차 증가시킨다. 서버의 처리능력이 포화 상태에 이르렀을 때가 웹 서버가 서비스

할 수 있는 최대 HTTP Operation수를 측정한다.



<그림31> SPECWeb96의 벤치마킹 구성도

SPECWeb96은 시스템이 얼마나 효과적으로 HTTP(Hypertext Transfer Protocol) GET 요구를 처리할 수 있는지를 측정하도록

설계되었다. 따라서 SPECWeb96 testbed는 시험 대상인 웹 서버 소프트웨어를 수행하는 서버 머신(UNIX 또는 Windows NT 머신)과 여러 개의 클라이언트 머신으로 구성된다. 클라이언트에서는 SPECWeb96 소프트웨어를 이용하여 서버 소프트웨어에 스트레스를 주는 작업부하가 생성된다. 이러한 작업부하는 서버 소프트웨어가 접속 건수에 의해 포화되고 응답 시간이 급격히 저하될 때까지 서서히 증가된다. 이렇게 서버가 포화되는 시점을 그 웹 서버 소프트웨어가 지원하는 초당 최대 HTTP 동작의 수가 되며, 이 값이 SPECWeb96 성능 지수가 된다.

▪ SPECWeb96의 작업부하 생성 방법

SPECWeb96의 작업부하는 여러 서버로부터의 로그(Log)를 분석하여 결정된다. 먼저 NCSA(National Center for Supercomputing Applications), HP, HAL사의 로그를 분석하고, 이를 Netscape과 CommerceNet사로부터 받은 요약 자료와 비교하였으며, 이러한 로그들로부터 파일의 크기와 액세스 빈도와의 관계가 매우 유사하다는 것을 발견하게 되었다. 즉, 꽤 많은 웹 액세스가 다소 작은 파일(적은 그래픽적인 요소, 짧은 텍스트 파일 등)을 요구하고, 대부분의 액세스는 수 KB의 크기를 갖는 파일(대부분 HTML 파일 및 그것의 주요 그래픽 등)을 요구한다. 그러나 그 이상 파일 크기가 증가하면 액세스 빈도는 점차 감소한다. 예를 들어, 다소 많은 액세스가 수십 KB의 크기를 갖는 HTML 파일이나 다른 문서 및 사진을 요구하지만, 100KB 이상의 큰 문서나 멀티미디어 파일은 매우 드물게 요구된다. 요약하면, 가장 일반적인 액세스 형태는 홈페이지 및 인덱스를 자주 브라우징 한 후, 가끔씩 매우 적은 수의 대용량 파일을 선택하여 다운로드 한다.

이러한 로그 데이터 분석을 기반으로, 네 가지 클래스(Class)의 파일들로부터 작업부하를 결정하였다. 즉, <표34>에 나타난 것처럼 1KB 이하의 파일은 전체 액세스의 35%, 1KB에서 10KB 사이의 파일 액세스는 50%, 10KB에서 100KB 사이의 파일 액세스는 14%, 그리고 100KB에서 1MB 사이의 파일 액세스는 1%를 차지한다고 가정한다. 또한 각 클래스 당 9개, 총 36개의 크기를 갖는 파일을 가정한다<표35>. 그러나 한 클래스 내에서의 액세스는 균일하지 않고 중간점을 중심으로 Poisson 분포를 갖는다고 가정한다. 이는 어떤 파일(예를 들어, index.html)이 다른 파일보다 자주 액세스 되고 또 다른 파일(예를 들어, mydog.gif)은 아주 드물게 액세스 되는 동작을 묘사하고 있다.

<표34> Class 당 파일 크기 및 액세스 빈도

class0	0~1KB	35%
class1	1~10KB	50%
class2	10~100KB	14%
class3	100KB~1MB	1%

<표35> 각 Class 에서의 파일 크기

class0	class1	class2	class3
102	1,024	10,240	102,400
204	2,048	20,480	204,800
306	3,072	30,720	307,200
408	4,096	40,960	409,600
510	5,120	51,200	512,000
612	6,144	61,440	614,400
714	7,168	71,680	716,800
816	8,192	81,920	819,200
918	9,216	92,160	921,600

마지막으로, 서버의 throughput 기대치가 커짐에 따라 전체 파일 집합의 크기가 증가되어야 한다. 이는 웹 사이트의 인기가 올라갈수록 웹 사이트의 크기가 커진다는 것을 의미하는 것이 아니라, 고성능 서버의 필요성이 증대된다는 것을 의미한다. 특히, 두개의 작은 시스템이 두 배의 성능을 갖는 하나의 큰 시스템으로 대체될 수 있다고 가정하는 것이 불합리하지는 않지만, 두 시스템에서의 파일들이 다소 중복되고 파일 공간이 선형적으로는 증가하지 않는다는 점을 반영해야 한다. 따라서 SPEC에서는 전체 파일 집합의 크기가 서서히, 즉 throughput이 4배로 커질 때 전체 파일 집합의 크기는 2배로 증가되도록 정한다. 또한 디렉터리의 수도 전체 파일 집합의 크기에 적용되었던 확장 함수 sqrt(throughput/5)*10을 따른다. Throughput 증가에 따른 디렉터리 수 및 필요한 디스크 공간 크기를 요약하면 <표36>과 같다.

<표36> Throughput 당 디렉토리 수 및 필요한 디스크 공간 크기

Throughput	디렉토리 수	디스크 공간 크기
1	4	22MB
2	6	31MB
5	10	49MB
10	14	69MB
20	20	98MB
50	31	154MB
100	44	218MB
200	63	309MB
500	100	488MB
1000	141	690MB

요약하면, 이러한 작업부하는 여러 명의 멤버에 대한 홈 페이지를 지원하는 시스템의 동작을 의미한다고 생각할 수 있다. 즉, 멤버 당 하나의 디렉토리를 갖는 디렉토리의 집합이 있으며, 각각은

4개의 클래스에 9개의 파일이 존재하는 총36개의 파일로 구성된다. 이러한 파일에 대한 액세스는 모든 디렉토리에 걸쳐 균등히 분산되지만, 각 디렉토리 또는 각 멤버의 홈 내에서는 파일 액세스가 언급한 함수에 따라 분산되므로 여러 개의 인기 있는 파일, 몇 개의 보통 파일, 다수의 비인기 파일을 만들어낸다.

▪ SPECWeb96의 내부 구현

SPECWeb96은 하나 이상의 클라이언트(또는 드라이버) 시스템이 SUT(System Under Test)에 대해서 HTTP GET 요구의 작업부하를 생성해 내도록 한다. SPECWeb96은 이러한 드라이버를 수행하는 코드를 제공하며, SUT에서 수행되는 HTTP 서버는 시험되는 대상으로 시험 자가 준비한다.

SPECWeb96은 ANSI C와 Perl로 구현되었는데, 벤치마크 드라이버와 작업부하 생성기는 C로, 툴 환경, 보고서 작성기 등은 Perl5로 구현되었다. 특히, 이러한 언어들은 여러 UNIX나 NT 시스템에 쉽게 이식될 수 있다는 특징을 갖는다.

SPECWeb96의 C 코드는 SPEC의 또 다른 네트워크 벤치마크(SPEC SFS로도 알려진) LADDIS로부터 많은 영향을 받았다. LADDIS는 네트워크상에서 부하 생성 프로세스간 조정을 수행하는 프레임워크를 이미 제공하므로, SPECWeb96에서는 LADDIS의 NFS 관련 코드만 HTTP 프로토콜 요구를 생성하는 코드로 대체하였다.

▪ SPECWeb96의 한계

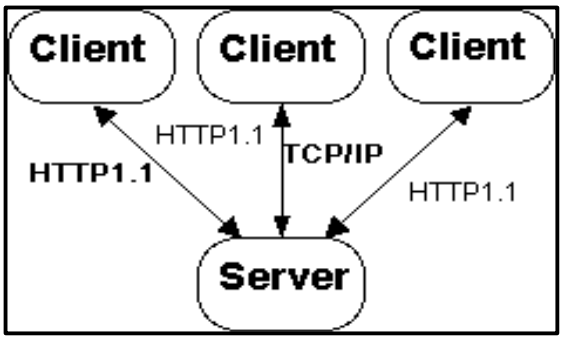
한편, SPECWeb96의 한계로는 HTTP GET만을 테스트함으로

써 POST, CGI Call, 보안기술등에 대해 고려하지 않고 있으며, HTTP 1.0 환경만을 지원함으로써 Session 유지기능 등을 가지는 HTTP 1.1은 지원하지 않고 있으며, 실제 웹서비스 환경인 WAN 환경에서의 테스트가 아니라는 점이다. 따라서 보다 완벽한 성능 평가를 위하여 SPECWeb96은 2000년 2월 SPECWeb99로 대체되었다.

▪ SPECWeb99

SPECWeb99는 SPECWeb96에서 발전한 것으로 작업부하는 여러 회사의 홈페이지를 서비스하는 웹 서비스 제공자에 접근하는 경우를 시뮬레이션 하는 것으로 각각의 홈페이지는 작은 아이콘으로부터 용량이 큰 문서나 이미지 등으로 구성된다. 테스트에 사용되는 웹 서버의 개수에 제한이 없으며, DBMS를 고려한 테스트가 아니며, 서버와 클라이언트로 구성되어 있는 단순한 웹 환경에서의 테스트를 위한 것이다. 또한, LAN환경에서의 테스트이고, WAN까지를 고려한 테스트가 아니며, 서버가 처리할 수 있는 최대 동시 연결(Connection) 수가 결과가 된다. 한편 SPECWeb99는 시스템 규모산정 툴로 디자인 된 것이 아니므로, 서버용량 산정을 위한 목적으로는 사용하지 말 것을 권장하고 있다.

- SPECWeb은 인터넷/인트라넷 Web Server 성능을 측정한다.
- 초기 릴리즈에서는 HTTP request를 서비스하는 서버의 능력을 측정하면서, 정적 웹 페이지에 대한 서버 성능에 중점을 둔다.
- 2005년 11월 공식적으로 후속 기준인 SPECWeb2005에 의해서 대체되었다.



<그림32> SPECWeb99의 논리적 구성요소

▪ SPECWeb99_SSL

SPECWeb99_SSL은 SPECWeb99에서 Secure Socket Layer(SSL) 암호화/암호해독이 추가된 벤치마크를 뜻한다.

- SPECWeb2005의 정의

SPECWeb2005는 Standard Performance Evaluation Corporation (SPEC), 비영리적인 그룹, 시스템 통합 사업자, 대학, 연구 단체, 컴퓨터 개발업체에 의해 개발된 소프트웨어 기준 제품이다. SPECWeb2005는 정적이고 동적인 웹 페이지 요구에 대하여 서비스를 제공하는 서버 시스템의 능력을 평가하기 위하여 디자인 되었다.

- SPECWeb2005의 특성

SPECWeb2005는 SSL(Secuure Socket Layer)과 non SSL의 측정을 지원하는 SPECWeb99와 SPECWeb99_SSL의 후속 프로그램으로 웹 사용자에게 객관적이고 대표적인 웹서버 플랫폼 벤치마크 측정도구이다. SPECWeb2005는 한 가지 방식의 작업부하

를 통해 웹 서버를 측정하는 것 보다는 오늘날의 다양한 웹 환경을 적절히 고려하여 작업부하 환경을 다음과 같이 3가지(banking, ecommerce, and support)로 디자인 되었으며, 기존의 동시 접속자수에서 session-based로 작업부하 측정 기준을 변경함으로써 작업부하에 대한 벤치마크와 서버를 이용하는 사용자 수에 대한 보다 명확한 상호관계를 분석할 수 있다.

SPECWeb2005는 HTTP 1.1과 SSL(HTTPS)을 지원하는 모든 웹 서버 소프트웨어에서 사용이 가능하다.

SPECWeb2005 벤치마크 테스트를 하기 위해서는 반드시 서버와 클라이언트 사이에 하나 이상의 네트워크를 통해 연결하여야 한다. 벤치마크 코드는 각각의 측정자에게 분산되어 배포되고 fileset은 서버에 의해 생성된다. 이때 다음 테스트 통제 파일은 특정의 테스트 조건을 위해 형성되고 벤치마크는 그 통제 파일로 사용된다.

SPECWeb2005는 일반화된 테스트 방식이지만 웹 서버에 있어서 가장 기본적인 기능을 강조하는데 많은 노력을 하였다. 결과적으로 웹상에서 기본적인 기능을 표준화 하여 다른 시뮬레이션 평가보다 뛰어난 성능을 보이고 있다. SPECWeb2005 의 평가목적은 웹상에서 미리 정의된 작업을 요청함으로써 동시 연결들의 최대 다수를 측량 하는 것이다. SPECWeb2005는 뚜렷이 구분되어지는 작업에 따라 3가지로 분리 되어 구성되는 특징이 있다.

즉,SPECWeb2005_Banking,SPECWeb2005_Ecommerce와 SPECWeb2005_Support로 구성 된다.

SPECWeb2005는 다음과 같은 특징을 가진다.

① 1가지의 작업부하 측정에서 3가지의 작업부하 측정으로 세분화된 구분(Banking, Ecommerce, Support)을 하였다.

② Client/prime client는 보다 이식성을 강화하기 위해서 자바코드로 완전하게 다시 작성하였다.

③ 접속기반 QOS(quality-of-service)에서 웹 페이지 기반 QOS로 변화 하였다.

④ 페이지 응답 요청 사이에 User think time 을 적용하였다.

⑤ 클라이언트가 GETs 요청을 할 때 유사한 페이지 값이 남아 있을 경우를 대비하여 캐싱을 적용한 응답시간을 고려하였다.

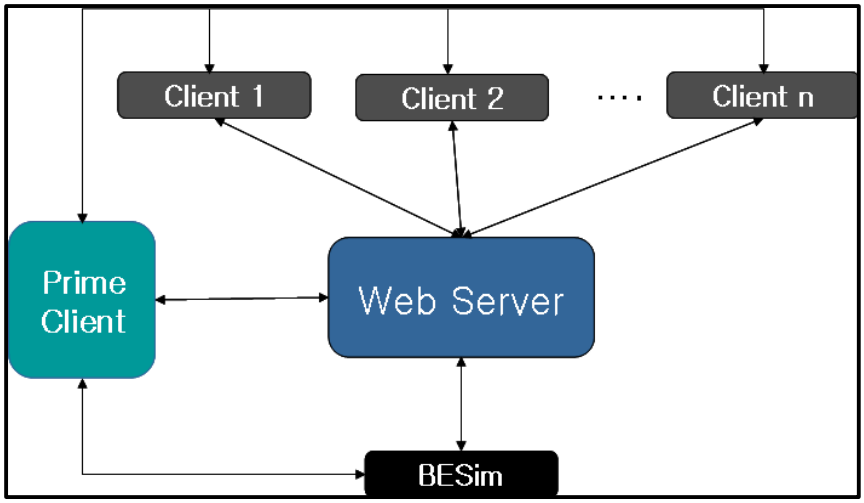
⑥ Client의 유연성을 크게 증가 시키고, 작업부하의 수정은 구성 파일을 통하여 편리하게 할 수 있게 구성 하였다.

▪ SPECWeb2005의 구성

SPECWeb2005에는 4개의 중요한 논리적인 구조로 구성 된다: 클라이언트, Prime clienet, 웹 서버 및 BeSim. 이 논리적인 구조는, 아래에 그림을 통하여 설명하고 있다.

클라이언트는 서버에 HTTP 요청을 보내고 서버에서 HTTP 응답을 받은 응용 프로그램이 실행한다. 이식성을 강화하기 위해, 응용 프로그램 및 Prime client 프로그램은 자바로 구성되었다.

Prime clienet는 다른 클라이언트들을 초기화하고 행동을 통제하며, 웹 서버 및 BeSim에 대하여 초기화 루틴을 가동시키고, 벤치마크 시험의 결과를 모아 저장하는 역할을 한다.



<그림33> SPECWeb2005의 논리적 구성요소

웹 서버는 하드웨어와 클라이언트가 발행한 요구를 클라이언트들로부터 정보를 수집하는 것이다.

BeSim은 웹 서버가 HTTP 응답(예를 들면 고객 자료)을 완료하기 위해 필요로 하는 특정의 정보를 회수하기 위해 의사를 소통해야만 하는 back-end 과정에 필요한 적용 서버를 에뮬레이트 하는데 필요하다. 즉, BeSim 타입의 웹 서버와 back-end 과정에 필요한 서버 사이의 통신을 에뮬레이트 하기 위해 존재한다.

Primary Metric(SPECWeb2005)은 공식(아래)에서 나타낸 바와 같이 100을 곱함으로써 결과값과 더불어 각각의 작업부하(banking, ecommerce, support)에 대한 접수의 값을 확인 할 수 있으며, SPECWeb2005 Primary Metric을 통해 각각의 작업 부하 점수에 관련된 종합 시스템 점수를 제공한다.

$$SPECweb2005=\sqrt[3]{\left(\frac{bank\_measured}{bank\_reference}\right)\times\left(\frac{ecommerce\_measured}{ecommerce\_reference}\right)\times\left(\frac{support\_measured}{support\_reference}\right)}\times 100$$

<그림34> SPECWeb2005 Primary Metric Calculation

작업부하 submetrics 점수는 동시접속 세션 단위로 나타난다. 이것은 SUT가 벤치마크의 QOS(Quality-of-service) 필요 물에 대처하는 동안 접속할 수 있었던 동시의 사용자 세션의 수를 나타낸다. Primary Metric은 상대적이고 종합적인 점수를 보여주고 submetric은 인터넷을 위한 실용적인 웹 사이트의 특성을 workload-by-workload view를 통해 제공한다.

SPECWeb2005 벤치마크는 http 서버의 측정에 사용되어왔다. HTTP 서버의 작업 부하는 한개 이상의 클라이언트 시스템에 의해 일어나고 Prime clienet에 의해 제어되며, 각 클라이언트는 HTTP 요청을 서버에 보내고 응답을 확인한다.

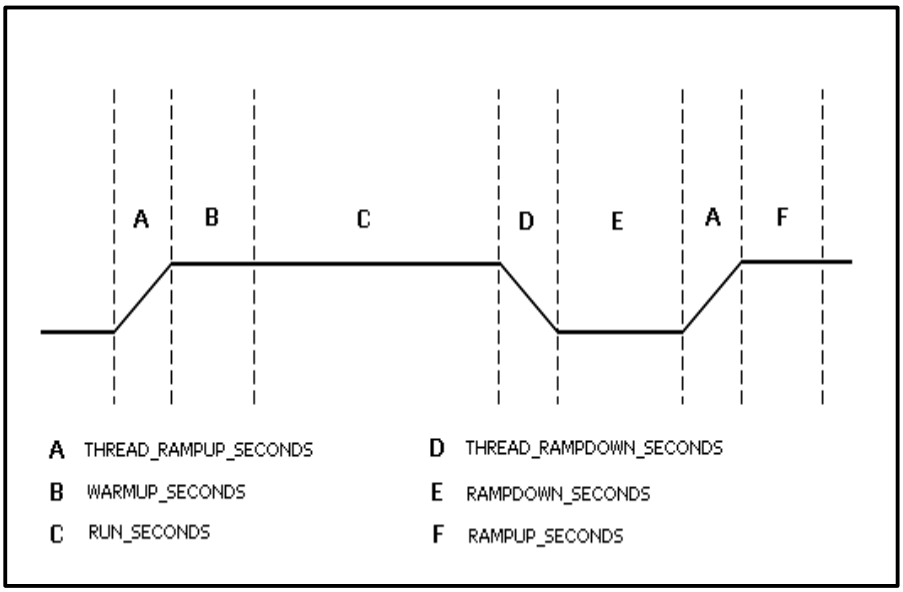
이때 모든 HTTP 요청이 보내어 지고, 응답 을 받음으로써 구성한다. 전형적으로, 동적응답과 모든 임베디드 이미지 파일, 수신된 바이트의 수, 응답 시간, QOS 기준 웹페이지 업무가 클라이언트에 기록된다.

벤치마크에 시작에 앞서, 하나 이상의 클라이언트 프로세서들은 각 클라이언트 시스템에 의해 시작된다. 이 프로세스들은 기본 포트(1099)를 따르거나, Test.config 에있는 유저에 의해 분류된 포트들을 따른다. 일단 모든 클라이언트 프로세스 들이 시작되면, 그 클라이언트 시스템은 Prime clienet에 의한 run-specific 초기화와 과부하에 준비한다. Prime clienet는 구성 파일로부터 나온 중요한 Value 들과 Test.config, Testbed.config , Workload-specific 구성

파일(예: SPECWeb_Banking.config)를 읽을 것이고, BeSim과 웹 서버를 위한 초기화를 수행할 것이다. 이때 연결이 성공적이면 Prime clienet는 각 클라이언트 프로세스를 초기화 한다. 모든 초기화가 성공적으로 마쳐지면 Prime clienet들은 벤치마킹을 시작할 것이다. 벤치마크가 종료될 때 Prime clienet는 모든 클라이언트들에게서 받은 데이터 결과를 수집하고, 이 자료를 모으고, 결과 파일에 이 정보들을 기록한다. 세 번의 모든 반복이 끝나면, ASCII text report file 과 HTML report file 형식으로 최종 보고 파일이 생성된다.

동시 세션들의 수는 벤치마크가 진행되는 동안 연속적으로 HTTP 서버에 요청들을 보내는 load-generating 프로세스/스레드들의 수와 일치한다. 각 스레드들은 workload-dependent 상태의 연속을 방해할 "유저 세션" 을 시작할 것이다. 시작되었던 유저 세션이 끝나면, 그 스레드는 새로운 유저 세션을 시작할 것이고 이 과정은 반복된다. 이 과정은 유저들이 사이트를 들어오고, 서버의 HTTP요청을 연속적으로 만든 후에 그 사이트와 연결을 종료 하는 것을 보여준다. 새로운 유저 세션은 그 이전 유저세션이 끝나자마자 시작되고, 이 과정은 벤치마크가 끝날 때까지 계속된다.

Prime clienet는 벤치마크 실행의 단계를 제어한다. 이 단계들은 아래 다이어그램을 통해 확인 할 수 있다.



<그림35> SPECWeb2005 Benchmark Phases

thread ramp-up period(A) 는 load generating 스레드들이 시작되는 기간을 말한다. 이 단계는 요청에 즉각적이고 full-load spike와 함께 벤치마크 실행을 시작하기 보다는 사용자의 activity를 올려주는 단계이다. warm-up period(B) 는 시스템이 실 측정 간격의 이전 캐쉬를 최적화 할 수 있는 기간이다. warm-up period의 끝에, 모든 결과들은 load generator로부터 클리어 되고 다시 레코링을 시작한다. 따라서 '실행 기간' 의 시작 전에 기록된 어떠한 에러들도 벤치마크 런의 결과에 영향을 미치지 않는다. run period(C) 는 벤치마크 결과가 기록되는 동안의 시간이다. 모든 HTTP 요청들의 결과는 마지막 벤치마크 결과에 기록된다. thread ramp-down period(D) 는 간단하게 A 의 반대라고 생각하면 된다. 이것은 모든 load-generating thread들이 정지되는 시간이다. 비록 이 기간 동안 load-generating thread들이 여전히 서버로 요청을 보내고 있지만, 모든 결과의 기록은 '실행' 기간의 끝에서 정지 될

것이다. ramp-down period(E)는 서버와 클라이언트에게 그들의 "unloaded" 상태를 바꾸기 위해 주어진 시간이다. 이것은 주로 다음 test iteration의 시작 전 TCP 커넥션 connection clean-up을 위한 충분한 시간을 확보하기 위한 기간이다. ramp-up period(F)는 두 번째와 세 번째 벤치마크 실행 반복을 위해 warm-up period(B)를 대체한다. 그것은 서버의 캐쉬가 이미 프라임(우위) 된 시점이라고 가정한다. 그래서 그것은 thread ramp-up period 와 run period 사이에서 steady-state 상태에 도달하기 위한 그 다음의 반복을 위해 더 짧은 시간을 요구한다. load-generating thread는 HTTP서버와 한 번의 커넥션에서 동적 요청을 만들 것이다. 그리고 그 커넥션뿐만 아니라 그 서버에 평행의 이미지 request를 만들기 위한 추가 커넥션을 재사용한다. 이것은 그 서버의 이미지 파일을 요청하기 위한 멀티플 커넥션들을 사용하는 일반 브라우저 행동을 모방하려고 한다. load-generating thread의 기록은 웹 페이지에서 돌아온 이미지를 추출하지 않는다. 그 대신, 각각의 동적 페이지를 위해 요청될 그 페이지의 이미지 파일은 workload-specific configuration file로부터 회복된다. load-generating thread에 의해 요청된 각 페이지를 위해, load generator는 HTTP서버에 요청을 보내기 바로 직전에 타이머를 시작하고, 그 페이지가 받아진 마지막 바이트 응답에서 타이머를 멈춘다. 또한 모든 서포팅 이미지 파일의 응답에 시간에 맞출 것이고, 모든 서포팅 이미지 파일을 포함한 페이지를 위해 전체 응답 시간에 도달하기 위해 동적 페이지 응답 시간을 더한다.

유효한 응답들은 집합된 페이지 응답시간에 체크된 작업부하분량을 위한 각각의 QOS밸류들, 응답하는 QOS필드(TIME_GOOD, TIME_TOLERABLE, or TIME_FAIL)는 증가 될 것이다. '실행'의 끝에서 Prime clienet들은 모든 클라이언트들로부터 수집된 실

행 데이터를 더하고 그 런이 벤치마크 QOS 기준에 맞는지 결정한다.

▪ SPECWeb2005의 Workload Design

SPECWeb2005는 Workload Design으로 크게 3가지로 구분된다.

▪ SPECWeb_Banking

SPECWeb_Banking은 온라인 은행 업무에 근거하며, 이 작업 부하에 있는 모든 요구는 SSL에 근거한 내용을 포함한다.

일반적인 Banking업무에서 요구되는 패턴을 분석하여 4가지의 카테고리로 분류하여 벤치마크를 수행한다. 4가지 과정은 다음과 같다.

① Account information access or Account Maintenance related

② Bill Pay related

③ Money Transfer

④ Loan related

SPECWeb2005에서는 기존 SPECWeb99에서의 Banking업무에 대한 처리내용을 분석하여 기존에 분석하기 어려웠던 부분을 보강하고자 하였다.

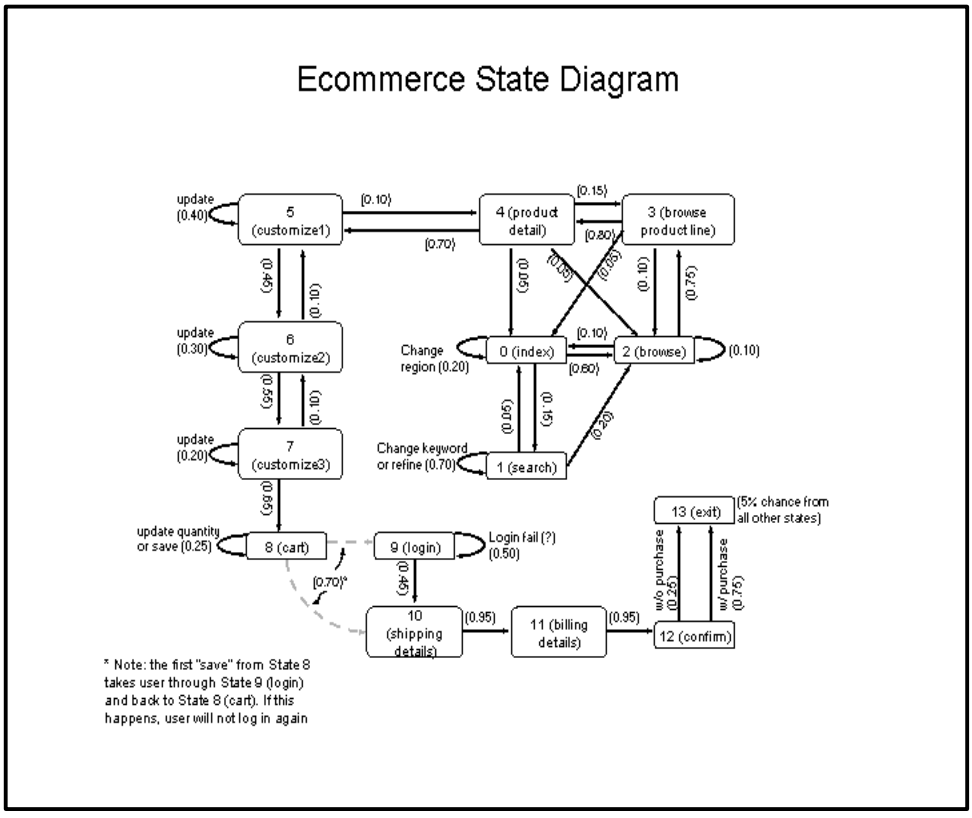
▪ SPECWeb_Ecommerce

SPECWeb_Ecommerce는 최종 사용자가 제품을 검색하고, 판매자가 주문을 받고, 구매를 허용하는 내용을 포함한다. 작업 노드

는 대중적인 웹을 포함하는 평균 페이지 크기를 분석해서 크기 및 접속 빈도와 같은 통계를 모으기 위하여 웹 판매점을 찾아보기뿐만 아니라 실제적인 전자상거래 사이트의 기록 파일을 개발하고 제품을 구매할 때의 정보를 분석한다.

다음과 같은 단계로 구성된다.

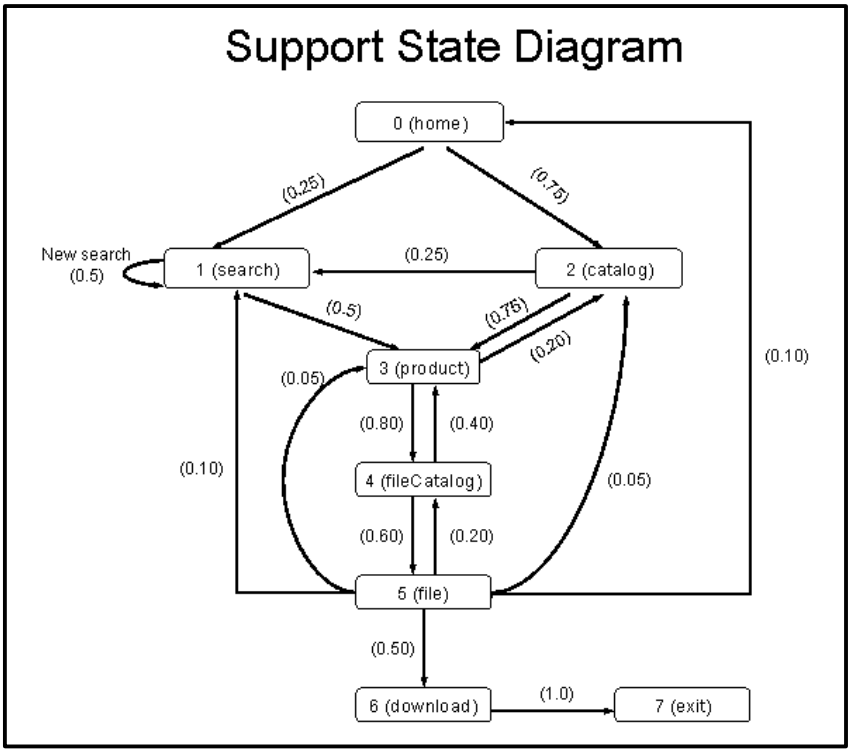
- Browse : 검색단계(색인, 조사, browse, browse_productline, productdetail)
- Customization : 주문단계
- Purchase : 구입단계



<그림36> Ecommerce State Diagram

▪ SPECWeb_Support

SPECWeb_Support는 vendor의 support web site를 벤치마크하기 위해 디자인 되었다. 사용자는 제품을 검사하고, 유효한 제품의 리스트를 검색하고, 특정 표준에 근거한 유효한 다운로드의 리스트들을 검사하고, 그 후에 파일을 다운로드하는 과정을 벤치마크한다.



<그림37> Support State Diagram

SPECWeb2005는 웹 서버 퍼포먼스를 측정하는 표준화된 벤치마크이다. 이전 버전의 성공을 위해 탄생한 SPECWeb2005는 사용자들에게 넓은 범위의 시스템으로부터 나온 결과들을 공정하게 비교할 수 있도록 하는 objective 측정을 제공한다. SPECWeb2005는

일반적이지만 세 가지의 전혀 다른 타입의 고객 행동을 나타내는 workload들을 포함한다. 그것은, 쇼핑, 뱅킹, 파일 다운로딩 이다. 동적 서버 응답을 생성하기 위한 PHP와 JSP 스크립트를 제공함에 따라, SPECWeb2005는 HTTP서버의 동적 요청 수행이 오늘날 동적 콘텐츠를 생성하는 기술을 위한 가장 일반적인 두 가지 기술에 기초했음을 확인했다.

8) SPECjAPPServer2004

⊙ Java/Client 벤치마크 개요

SPECjAPPServer2004은 SPEC의 Java/Client를 위한 벤치마크 기준으로 SPECjAppServer2001과 그 뒤를 이어서 발표된 SPECjAppServer2002의 후속 모델이다. SPECjAPPServer2004이외에도 현존하는 Java/Client 벤치마크 기준으로는 SPECjbb2005이 있다. SPECjAppServer2004는 J2EE1.3의 사양을 근간으로 J2EE 서버들과 컨테이너들의 성능(performance)과 확장성(scalability)을 측정하기 위한 것이며, SPECjbb2005는 서버측면의 자바 성능을 평가하기 위한 벤치마크 기준이다. 이들 SPECjAppServer2004와 SPECjbb2005의 특성을 비교하면 <표37>와 같다.

<표37> SPECjAppServer2004와 SPECjbb2005의 특성

구분	SPECjAPPServer2004	SPECjbb2005
목적 및 대상	J2EE1.3의 사양을 근간으로 J2EE 서버들과 컨테이너들의 성능(performance)과 확장성(scalability)을 측정하기 위한 벤치마크	서버측면의 자바성능을 평가하기 위한 벤치마크
발표시기	2004. 4	2005. 6
측정단위	JOPS	bops, bops/JVM
이전모델	SPECjAppServer2002	SPECjbb2000

SPECjbb2005에 대한 세부적인 내용은 3.2.5에서 언급하고 있어 여기에서는 상세하게 언급하지 않는다. 한편, SPECjAppServer2004는 SPECjApp Server2002의 후속버전으로 이에 대한 내용을 개략적으로 살펴보고자 한다. SPECjAppServer2002(Java Application Server)는 완전한 엔드-투-엔드 웹 어플리케이션에서 J2EE API들을 부분적으로 사용하는 자바 엔터프라이스 어플리케이션 서버의 성능을 측정하기 위한 클라이언트/서버 벤치마크이다. 2002년 9월에 발표된 SPECjAppServer2001과는 엔터프라이스 자바빈스가 EJB1.1 스펙 대신에 EJB2.0 스펙을 사용하여 정의된다는 사실을 제외하고는 동일하다.

<표38> SPECjAPPServer2004, SPECjAPPServer2002, SPECjAPPServer2001의 비교

구분	SPECjAPPServer2004	SPECjAPPServer2002	SPECjAPPServer2001
목적 및 대상	J2EE1.3의 사양을 근간으로 J2EE 서버들과 컨테이너들의 성능(performance)과 확장성(scalability)을 측정하기 위한 벤치마크	완전한 엔드-투-엔드 웹 어플리케이션에서 J2EE API들을 부분적으로 사용하는 자바 엔터프라이스 어플리케이션 서버의 성능을 측정하기 위한 클라이언트/서버 벤치마크	
		EJB2.0 스펙 사용	EJB1.1 스펙사용
발표시기	2004. 4	2002.12	2002. 9
측정단위	JOPS	TOPS, Price/TOPS	BOPS, Price/BOPS

이전모델	SPECjAppServer2002	SPECjAPPServer2001	-

SPECjAppServer2002는 국부적인 인터페이스처럼 몇가지 EJB2.0 특성들의 장점을 갖는다. EJB-QL는 언어와 엔티티 빈즈들 사이의 CRM(Container Managed Relationships)을 쿼리 한다. 클라이언트 측의 SPECjvm98와 서버 측의 SPECjbb2000의 결합은 SPECjAppServer2002와 SPECjAppServer2001이 자바 어플리케이션에서 동작하는 시스템의 능력을 측정하기 위해 대부분의 객관적이고 대표적인 벤치마크이어서 주어진 자바 사용자들의 SPEC 관습을 지속한다. SPEC은 서버 SUT(Systems Under Test) 뿐만 아니라 자바 엔터프라이스 에플리케이션 서버, JVM(Java Virtual Machine)를 실험하기 위해서 SPECjAppServer2002 벤치마크를 설계하였다.

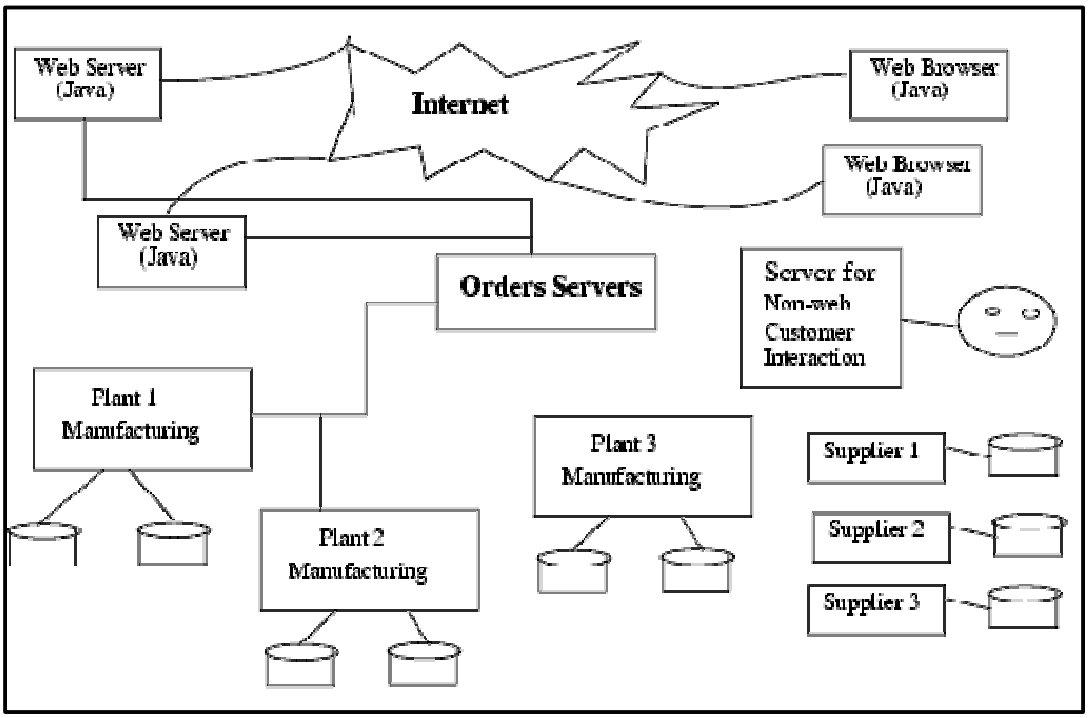
⊙ SPECjAppServer2004 모델

SPECjAppServer2004는 J2EE1.3의 사양을 근간으로 J2EE 서버들과 컨테이너들의 성능(performance)과 확장성(scalability)을 측정하기 위한 J2EE 벤치마크 이다. J2EE는 기본적으로 분산된 어플리케이션들 즉 웹 컴포넌트들과 컴포넌트 기반 트랜잭션 미들웨어(OTM : Object Transaction Middleware)의 확장성을 수립하기 위한 기반구조이다. 요구가 증가하기 때문에 J2EE 어플리케이션들은 자동적으로 조정이 필요하다. 일반적인 DBMS 확장성(예를 들어, DB 입출력, 현금, 자금 관리 등)에 대한 관점은 이러한 작업부하에 초점을 맞추고 있지 않으며, TPC-C, TPC-D, TPC-W 등의 다른 표준 작업부하에 의해서 적절하게 통제(stressed) 되었다.

SPECjAppServer2004는 동적인 웹 페이지의 생성, 메모리관리, 연결 폴링(connection pooling), 비활성화/활성화 객체의 유지, 캐싱, 메시지 큐잉 등 다양한 서비스를 다루기 위해서 J2EE 어플리케이션 서버들의 능력을 통제한다.

여기에서는 제조, 공급망관리(SCM: Supply chain management) 그리고 주문/물품명세 등과 같은 업무에 중점을 두고 있으며 업무 범위와 업무에 친숙한 시나리오로서 제조, 공급망관리(SCM: upply chain management) 그리고 주문/물품 명세를 사용한다. 이것은 내용이 충실한 산업-강도 분산 문제이다. 또한 매우 과중하고 임무 지향적(Mission critical)이면서 글로벌(Global)한 무중단서비스(7일X24시간)를 지원해야 하기에 강력하고 확장성을 지닌 기반구조를 필요로 한다. 이는 포춘의 500대 기업들이 갖는 관심사 중 하나다. 이는 매우 중요한 사항들이기에 다음과 같은 J2EE 서비스들의 사용을 필요로 한다.

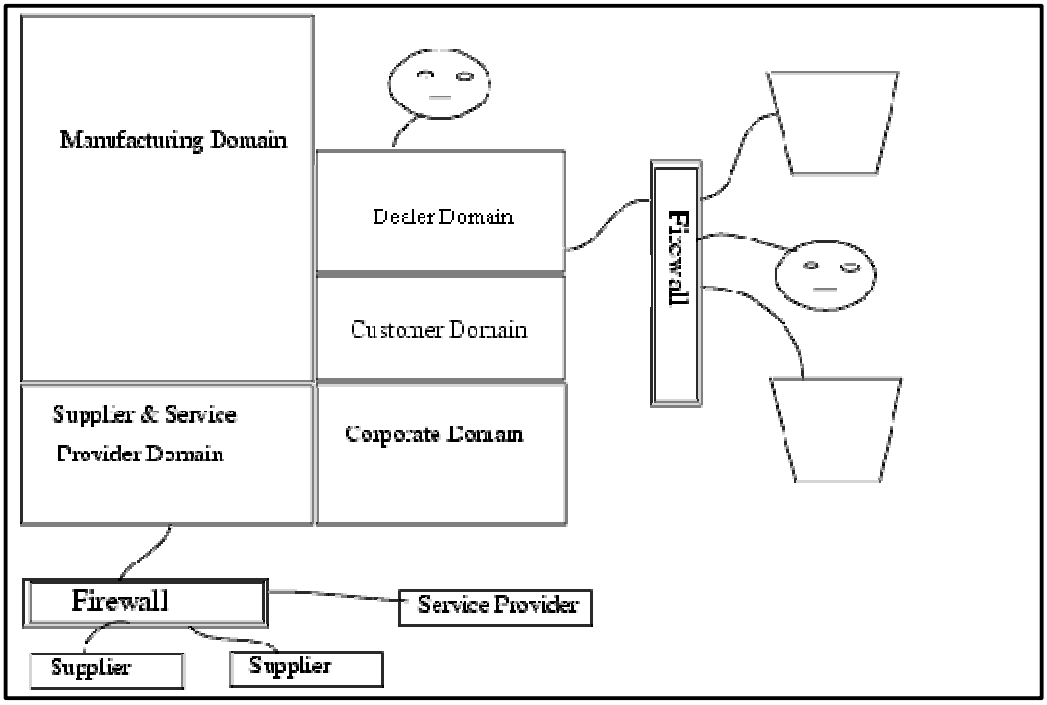
- 동적 웹 페이지 생성
- 트랜잭션 컴포넌트들
- 분산된 트랜잭션들
- 메시지 및 비동기 태스크 관리
- 다중-사이트 서버들을 갖는 다중 회사 서비스제공자
- 기존시스템과의 인터페이스 및 사용
- 안전한 트랜잭션
- 규칙기반 인증
- 객체 지속성



<그림38> SPECjAppServer2004 전 세계 분산 비즈니스

비즈니스에서는 제품 구성에 대한 사양, 주문 및 상태확인 등 고객들의 직접적인 의사교환을 위해서 웹을 전략적으로 사용한다. 이러한 비즈니스들에서는 자동적인 제조, 재고, 공급망 관리, 요금 부과 등을 완전하게 지원하기 위해 노력한다. <그림38>는 이러한 비즈니스에서 필수적인 것이 무엇인가를 묘사한다. 즉시(Just in Time) 제조의 개념을 사용하여, 그들은 최대 효율과 최소 재고를 추구한다. 그들의 고객과 비즈니스 프로세스들이 동시에 동작하기에 트랜잭션 지향적이어야 하고 보안을 추구해야 하며, 국제화가 요구된다. 또한 24시간 365일의 중단 없는 서비스와 완전한 분산이 보장되어야 한다. 이와 같이 회사에서는 그들의 어플리케이션 시스템 개발자들이 비즈니스 규칙에 초점을 두고 확장 가능한 트

랜잭션들이 복잡하거나 너무 섬세하지 않게 메세징, 디렉터리/명명, 보안, 인증 혹은 서비스가 H/W 사양으로 매핑(Mapping) 하여야 한다. 이에 비즈니스들은 확장가능성, 도달가능성, 분산된 컴포넌트 등을 기초로 RAD/IDE 기반구조 하에서 개발되는 것이 필요하다.



<그림39> SPECjAppServer2004의 다중 도메인 국제적인 비즈니스

위의 <그림39>은 <그림38>와 같은 동일한 비즈니스를 묘사한 것으로 국부적인 도메인들을 보였다. 도메인에 대한 용어는 논리적인 엔터티를 나타내는 것으로 이는 운영을 위한 분리된 비즈니스 영역을 말한다. 이러한 분리된 비즈니스 영역을 나타내는 5가지 도메인들은 다음 표에서 제시한 바와 같이 판매자(Dealer), 제조(Manufacturing), 공급자(Supplier), 고객(Customer)과 협력

(Corporate) 도메인이 있다.

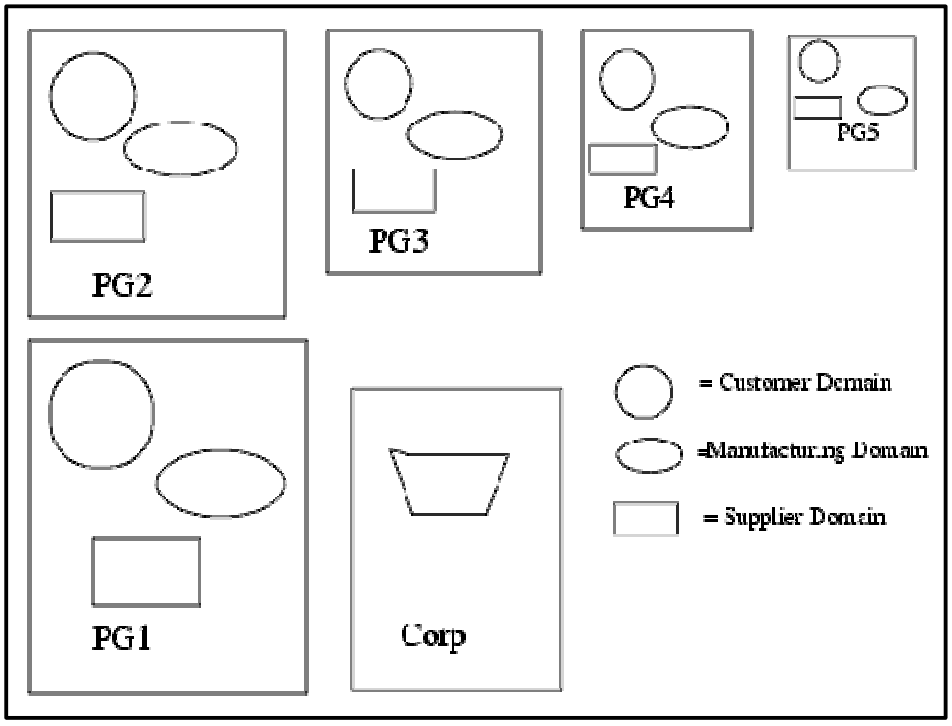
판매자 도메인은 비즈니스 도메인들에서 고려될 수 있는 제조, 공급자, 고객과 협력업체 등 다른 도메인에 의해서 제공되는 서비스에 접근하기 위해서 사용되는 웹-기반 어플리케이션의 사용자인터페이스 요소들을 포함한다. 모델화 되는 크기의 협력에 있어서 각각의 비즈니스 도메인은 DB와 어플리케이션들은 분리된다. 대부분의 경우에서 그들은 분리된 컴퓨팅 하드웨어 상에서 구현된다. 거기에는 회사내부와 외부의 공급자 및 고객들 사이에 제조자-고객 관계가 존재한다. 역사적인 이유 때문에, 각각의 비즈니스 도메인은 분리된 엔터티 ID들을 가질 수 있다. 이러한 이유로 전역적인 고객과 공급자 DB는 어플리케이션들을 수반하는 협력 도메인 내에 존재한다. 분산 DB와 어플리케이션들은 SPECjAppServer2004 작업부하를 위한 기반을 제공한다. 예를 들어 자동차 판매상과 같은 고객들은 웹을 통한 직접적인 접근 등, 어떠한 몇 가지 방법들을 통해서 비즈니스에 접근한다. 전화 혹은 판매원을 통하는 것과 같은 개인-대-개인의 접촉은 고객 서비스 제공자 혹은 판매자 자신들이 웹 인터페이스를 사용함으로써 이러한 웹 접근전략으로 전환된다. 모든 다국적 사무실과 공장들은 다른 사무실이나 공장에서 가지고 있는 자료를 자주 접근하여야 하며, 그들의 글로벌(Global)한 자료를 동시에 비교, 수집, 검증하여야 한다. 그들 회사는 또한 완전히 분리된 공급 회사와 상호작용을 수행해야 한다. 각각의 공급자들은 그들의 컴퓨팅 자원들을 독립적으로 보유하고 있다.

<표39> 도메인 및 특성

도메인	특성
판매자 (Dealer)	어플리케이션은 자동차 판매상들이 그들의 회계자료를 유지하고 판매 물품목록을 추정하며, 자동차들을 판매하고, 쇼핑카트를 유지하며, 자동차들을 구입을 모든 사용자들에게 허용하기 위해서 웹기반 사용자인터페이스를 구현한다. 웹 구성요소들은 판매자 도메인의 요구되는 서비스들을 제공하는 적정 세션 빈즈들을 통해서 비즈니스 도메인들과 통신한다. 판매자 도메인들의 구성요소들은 DB나 엔티티 빈스 들과 직접적으로 통신하지 않는다.
고객 (Customer)	고객 도메인의 작업은 OLTP적인 특성을 갖는다. 고객 도메인의 기능성은 새로운 주문의 추가, 특정한 주문의 상태 검색 혹은 특정한 고객의 모든 주문들과 주어진 주문의 취소를 포함한다. 또한 고객들로부터 주문을 받았을 때, 협력 도메인에 대한 요구를 보냄으로써 신용 체크가 수반되어야 한다. 따라서 고객 또는 판매자는 특정 주문의 상태와 주문에 관련된 모든 주요사항의 진행 상태를 파악할 수 있어야 한다.
제조 (Manu facturing)	제조 도메인은 제조공장내의 생산라인의 활동을 모델화 한 것이다. 그런 생산라인은 두 가지 형태의 라인이 존재하는데 Plannedlines과 LargeOrder lines이다. Plannedlines은 스케줄상에서 동작하고 사전에 정의된 수의 위제트를 생산한다. LargeOrder lines은 단지 판매자와 같은 고객으로부터 대규모 주문이 발생했을 때 동작한다. WorkOrder가 시스템 내로 진입될 때, 생산이 시작된다. 계획된 라인 line *WorkOrders*는 전형적으로 예측 어플리케이션의 결과로부터 생성된다. *LargeOrder* line *WorkOrders*는 고객의 주문으로부터 생성된다. *WorkOrder*is 가 생성 될 때, 위지트 유형에 부합하는 BOM은 검색되어지며, 요구되는 부품들에 대한 재고 부족이 발생한다. 위지트는 조립라인을 통해서 이동하기 때문에 진행상황을 *WorkOrder* 상태에 반영해야 한다. *WorkOrder*is가 완료되었을 때, 완료 또는 재고 갱신으로 표시한다. 부품들의 재고가 고갈되기 때문에 공급자는 위치되어지는 것이 필요하고 구입 주문은 보내져야 한다. 이것은 공급자 도메인을 접속함에 의해서 수행된다.
공급자 (Supplier)	공급자 도메인은 공급자들과의 상호작용을 위한 책임을 진다. 공급자 도메인은 주문되는데 필요한 부품들에 기초하여 선택하는 공급자를 결정한다. 회사는 선택된 공급자에게 주문을 보낸다. 구입 주문은 구입하는 다양한 부품의 수량과 인도장소, 인도일자 등을 포함한다. 부품들을 공급자로부터 받게 되면 공급자 도메인은 재고 갱신을 위해서 제조 도메인에 메시지를 전송한다.
협력 (Corporate)	협력 도메인은 고객들과 그들의 재고들에 대한 전역적인 목록을 관리한다. 신용제한을 포함한 모든 고객에 관한 신용정보는 협력도메인 내 독립적으로 하나의 DB내에서 유지된다. 이는 보안과 프라이버시를 제공한다.

SPECjAppServer2004 전사적 비즈니스는 성공적인 비즈니스의 하나이다. 그것은 많은 다른 의미들을 통해서 성장한다. 이것은 단순한 성장 뿐 아니라 분사와 회사합병 등을 포함한다. 첫 번째 방

법, 성장은 간단하게 회사는 보다 많은 고객을 확보하고 보다 많은 물건을 판매하는 것을 의미한다. 회사의 공장들은 점점 더 커지고 많아진다. 그러한 비즈니스는 다양한 공급자들의 요구를 갖는다. 그들 역시 성장한다.



<그림40> 분산된 세계적인 SPECjAppServer2004 비즈니스 조정

SPECjAppServer2004 비즈니스는 또한 분사와 합병을 통해서 성장할 수 있다. 이들 두 가지 경우에 있어서 복합기업인 회사는 전적으로 새로운 형태의 비즈니스를 가질 수 있다. 그것은 분리된 고객과 공급자 집단을 위한 새로운 부서들을 조직하기도 한다. 신규 조직의 고객들, 제조 공장들, 그리고 공급자들의 통합은 제조그

룹(Production Group : PG)이라고 불린다. <그림40>은 몇 개의 성장 단계를 거친 회사 상태를 보이고 있다. 몇 가지 PG들은 증가하기 때문에, 존재하는 PG의 크기들은 크게 성장하며, 새로운 것 중 하나가 가장 작게 된다. 하나의 협력업체 도메인은 다양한 PG들 사이에서 조정자로서 동작한다.

이것은 실세계상에서 발견되는 비선형 크기 모델을 표현한다. SPECjAppServer2004 비즈니스는 글로벌(Global)한 운영상황을 가정한다. 그것의 제조와 조립, 분배 공장들은 여러 도시에 흩어져 있다. 각각 위치에는 어플리케이션 서버, DB 서버 클라이언트 연결 멀티플렉서 등등의 다중의 컴퓨터들이 존재한다. 최종사용자는 데스크 탑, 웹 탑 등을 갖는다. 거기에서는 다수의 컴퓨터가 비즈니스를 위해서 서로 다른 형태의 업무를 수행한다. 몇몇 컴퓨터들은 다른 점이 많은 여러 서비스와의 결합하는 않고 바로 단일서비스만을 제공한다. 노드의 고장에 관계없이 서비스가 제공되어야 하기에 몇몇 컴퓨터들은 고가용성을 보장하도록 구성되어야 한다. 또한 비용요소들이 기술되어야 하며, 다른 한편으로 모든 비즈니스 컴퓨터에 고장이 발생하지 않도록 유념해야 한다. 대신 비즈니스는 과잉 서비스가 독립적인 컴퓨터들에 사상되도록 구성되어야 한다. SPECjAppServer2004 비즈니스는 고객, 공장(제조, 조립, 분배 등) 그리고 공급자들로 구성된다. 비즈니스는 복잡하고 분산되어 있으며, 많은 컴퓨터들이 다양한 위상(位相)에서 독립적으로 사용하도록 구성되어 있다. 이것은 어플리케이션 서버들과 미들웨어의 과부하를 위한 많은 흥미 있는 방법들을 제안한다.

SPECjAppServer2004 작업부하는 그들의 주요 고객이 자동차 판매상인 자동차제조자를 대상으로 특별하게 모델화되었다. 고객들은 제품 카탈로그의 탐색, 자동차의 판매 혹은 구입, 판매제품의 인도과정을 추적하기 위해서 웹 기반의 사용자 인터페이스를 사용

한다.

9) SPECjbb2005

⊙ SPECjbb2005(자바 서버 벤치마크) 개요

SPECjbb2005 (자바 서버 벤치마크)는 서버 측면의 자바 성능을 평가하기 위한 SPEC의 벤치마크이다. 이전(以前) 벤치마크인 SPECjbb2000과 같이 SPECjbb2005는 중간층(middle tier)을 중요시하는 3-tier 클라이언트/서버 시스템을 에뮬레이션 하여 서버 측면의 자바 성능을 평가한다. 이 벤치마크는 JVM (Java Virtual Machine: 자바가상머신), JIT(Just-In-Time: 즉시실행) 컴파일러, 폐영역 회수(garbage collection), 스레드(threads)의 실행 및 운영체제의 일부분을 시험한다. 이것은 또한 CPU, 캐쉬, 기억 계층(memory hierarchy)의 성능 및 공유메모리 프로세서(shared memory processors: SMPs)의 범위성(scalability)을 측정한다. SPECjbb2005는 실세계의 애플리케이션을 설계하는 데 좀 더 객체지향적인 방법으로 구현된 새로운 고도화 작업부하(enhanced workload)를 제공하고, 현재의 애플리케이션을 좀 더 실질적으로 반영하는 벤치마크에 XML 프로세싱과 BigDecimal 연산처리와 같은 새로운 특징들이 있다.

⊙ SPECjbb2005 벤치마크의 특징

- 현재 서버 측면의 자바 애플리케이션의 가장 보편적인 형태인 3-tier 시스템을 에뮬레이션 한다.
- 업무로직 및 객체처리와 같은 중간층을 중요(predominate)시 한다.
- 드라이버 스레드에 의한 클라이언트 대체 및 객체의 바이너리 트리에 의한 데이터베이스 저장을 대체한다.

- 적용되는 작업부하 량의 증가 및 범위성에 관한 도표 그래프 뷰를 제공한다.

⊙ 개발 배경

SPECjbb2005는 중간층을 강조하는 3-tire 시스템을 에뮬레이션 하는 자바프로그램이다. 임의입력선택은 첫째 층(사용자 층)을 표현한다. SPECjbb2005는 중간층인 비즈니스 로직을 거의 완전하게 구현한다. 셋째 층은 개별 데이터베이스 보다는 자바 컬렉션으로 구현된 객체들의 테이블로 표현된다.

SPECjbb2005의 이면의 동기(動機)는 데이터베이스와 사용자 간의 미들웨어로서 자바 사용을 보편화 하고 지속하자는 것이다. SPECjbb2005는 벤치마크를 수행하기 위해 시스템을 고립시켜 단순화(simplification)하는 중간층 시스템의 전형적인 예(例)다. 이런 전략은 벤치마커들이 다양한 JVM 시스템을 측정하는데 빠른 데이터베이스 시스템에 투자하는 비용을 절감 시켜 준다.

SPECjbb 2000의 후속(follow-on)으로 발표(release)된 SPECjbb2005는 TPC-C 벤치마크에 의해 시작되어(inspired) TPC-C의 스키마, 입력 생성(generation) 및 트랜잭션 프로파일의 명세(specification)를 대체적으로 따르고 있다. SPECjbb2005는 각 스레드가 특정 로직 처리의 호출 이전에 독립적으로 임의 입력을 생성하는 독립형(single) JVM에서 실행된다. SPECjbb2005는 네트워크와 디스크 IO에서 적용되지 않는다.

완전히 발전한(full-fledged) 다층(multi-tier) 자바 벤치마크를 위해서, SPEC은 포괄적인 애플리케이션 서버 벤치마크인 “SPECjAppServer2004”를 개발하였다.

⊙ 일반 개념

SPECjbb2005에서 웨어하우스는 저장 데이터의 일부분이다. 웨어하우스는 25MB를 점유하며 수개의 컬렉션(HashMaps, TreeMaps)내에 다수의 객체 형태로 저장된다. 스레드는 웨어하우스에서 처리를 요구하는 활성화 사용자의 포스팅 트랜잭션을 표현한다. 웨어하우스와 스레드 간에 또, SPECjbb2005 메인의 스레드와 다양한 JVM 함수 사이에는 일대일 매핑이 존재한다. 벤치마크가 완전하게 실행되는 과정에서 웨어하우스의 수가 증가하는 것과 같이, 스레드의 숫자 또한 증가하게 된다.

"포인트(point)"는 부여된 웨어하우스의 수에서 측정 간격상의 처리율(throughput)이다. 완전한 벤치마크 실행은 웨어하우스의 갯수를 증가 (따라서 쓰레드의 갯수도 증가)하는 포인트 측정의 일련의 과정이다.

사용자는 실행을 위한 애플리케이션의 인스턴스의 개수를 조정할 수 있다. 한개 이상의 인스턴스를 선택하면, 개별 인스턴스의 합으로 표현되는 최종 측정치를 가지는 다수의 인스턴스가 동시에 실행될 것이다. 다수의 애플리케이션 인스턴스는 지역 소켓 통신과 제어기(controller)를 사용하여 동기화(synced) 한다.

⊙ SPECjbb2005 벤치마크 관련 문서

SPECjbb2005 벤치마크 지원 및 기술 문서는 4개로 구성되어 있다.

- SPECjbb2005 FAQ (최종 수정: 2006년 3월 15일 V1.04)
- SPECjbb2005 사용자 지침 (최종 수정: 2006년 3월 15일 V1.06)
- SPECjbb2005 실행 및 보고 규정 (최종 수정: 2006년 3월 15일 V1.05)
- SPECjbb2005 설계 문서 (최종 수정: 2006년 3월 15일 V1.05)

⊙ SPECjbb2005 벤치마크 결과물은 4개로 구성되어 있다.

- 결과물 파일 (Results files)
- html 파일 (Html file)
- 원시 파일 (Raw file)
- 텍스트 보고 페이지(Text report page)

위의 모든 벤치마크 결과물 파일은 동일한 정보를 서로 다른 방식으로 표현한 것이다. 여기에는 개별 웨어하우스의 개수에서의 종합 점수, 검정과 에러 메시지, 구성에 관한 정보, 입력 매개변수에 대한 정보, 그리고 각 포인트에 대한 상세내용 등의 SPECjbb2005 측정치를 포함하고 있다.

⊙ SPECjbb2005 벤치마크 테스트 환경(SUT)

벤치마크를 실행하기 위해 요구되는 하드웨어의 조건은 독립형 공유-주소(shared-address) 시스템이 필요하다. 벤치마크를 수행하기 위해 필요한 최소 구성은 1.7 기가 펜티엄 M 프로세서, 512메가바이트 힙을 사용하는 1기가 메모리를 가진 랩탑 컴퓨터에서 8웨어하우스까지 운용되어야 한다. 벤치마크 실행을 위해 요구되는 소프트웨어인 SPECjbb2005는 오직 하나의 운영체제와 J2SE 5.0의 특징을 지원하는 자바가상머신(JVM)이 필요 하다. 테스트에는 서로 다른 HW, OS, and JVM 구성을 모두 포함해야 한다.

SPECjbb2005 벤치마크는 단독형 자바 애플리케이션의 하나 또는 그 이상의 인스턴스로서 하나의 머신(단독 OS 이미지 내에서)에서 실행된다. 클러스터 또는 머신의 집합 사용은 허락되지 않는다. 네트워크, 데이터베이스 또는 자바 컴퍼넌트는 필요치 않으며, 오직 자바 환경이 필요하다.

⊙ SPECjbb2005 벤치마크 개발 및 사용

SPECjbb2005는 자바 소위원회 핵심설계팀인 BEA 에서 개발하였는데, BEA는 이 제품의 위상을 설계, 구현, 시험에 참여한 다쓰머트 기술대학 (Darmstadt University of Technology), HP, IBM, Intel, 그리고 Sun 으로 구성되었다.

SPECjbb2005 벤치마크는 SPEC의 웹사이트에서 온라인 주문할 수 있으며, 현재 SPEC의 모든 벤치마크 가격이 온라인 주문서에 있다. SPEC 회원은 추가 비용 없이 벤치마크를 제공받을 수 있다.

SPECjbb2005 라이선스를 구매함으로써 결과를 발표할 수 있다. SPECjbb2005 결과를 발표하기 위한 현재의 비용을 알고 싶으면 웹사이트에 연락하면 되고, SPEC 회원은 무료로 제출할 수 있다.

⊙ SPECjbb2005 벤치마크 운영 검정

발표 가능한 결과가 되기 위해서나 기존의 발표 결과와 직접적인 비교를 위해 벤치마크 실행은 반드시 다수의 검정 체크를 통과하여야 한다.

- jbb.jar의 검사합(checksum)은 SPEC가 제공한(shipped)한 동일한 바이트코드를 나타내야 한다.
- 각 측정 간격의 생략시간(elapsed time)은 0.5%보다는 낮거나, 특정 측정 간격 길이 (240 초)보다는 10%가 높은 범위에 있어야 한다.
- 실행은 8 웨어하우스의 최소치와 기대된 최고 보다 2배의 웨어하우스 사이에 포함되어야 한다.
- 자바 환경은 어느 지점에서의 운용되는 것보다 우선하는 벤치마크를 통해 부분적인 conformance testing을 통과하여야 한다.

⊙ 설계 문서

SPECjbb2005는 SPEC(Standard Performance Evaluation Corporation) 에서 개발한 소프트웨어 벤치마크 생산품이다. SPEC는 컴퓨터 판매업체, 시스템 통합업체, 대학, 연구기관, 발표자(publishers), 컨설턴트 등으로 구성된 비영리 기관이다. 이것은 자바서버 애플리케이션의 운영에 시스템 성능을 측정하기 위해 설계되었다.

SPECjbb2005는 SPECjbb2000를 기반으로 설계되었다. SPECjbb2000는 처음에 TPC-C 벤치마크로 시작하였고, TPC-C의 스키마, 입력 생성 및 운영 프로파일의 명세를 따라가게 되었다. SPECjbb2005는 자바클래스를 포함한 데이터베이스 테이블로 대치되었고 자바 객체를 가지 데이터 레코드로 교체되었다. 자바 객체는 메모리에서 자바 컬렉션 인스턴스나 다른 자바 데이터 객체에 의해 유지되어 진다.

SPECjbb2005는 중간층을 중시하는 3-tier 시스템을 에뮬레이트하는 자바 5.0 애플리케이션으로 구현되었다. 세 층은 모든 동일한 JVM 내에서 구현되었다. (벤치마크가 다수의 JVM 인스턴스를 산출한다고 할지라도) 모델이 된 시스템은 여전히 다수의 구역(districts)을 감당(serve)하는 웨어하우스를 가진 도매상 회사이다. 고객(혹은 사용자로 알려진) 이 새로운 주문이나 기존 주문에 관한 상황 요구 등을 시작하면서 운영이 초기화 된다. 회사 내 부가적인 운영은 배송 요구 절차 고객 지불 입력, 재고 수준 체크 및 주어진 고객에 의한 최근 행동의 보고 등으로 창출된다. 웨어하우스 당 단지 1명의 고객만이 존재한다. 웨어하우스는 저장된 데이터의 일부분이다. 이는 대략 자바 컬렉션 객체내의 저장 데이터의 25MB를 점유한다. 사용자들은 직접적으로 자바 스레드와 연계

(map) 된다.

각 스레드는 확률 배분을 사용하여 혼합된 작업에서 선택된 작업을 순차적으로 작동을 실행한다. 벤치마크가 모두 실행되는 동안 웨어하우스의 개수가 증가하고, 따라서 스레드의 수도 증가한다.

SPECjbb2005는 거의가 자기 포함적(self contained)이고 자기 주도적(self-driving) (스스로 데이터, 멀티-스레드 작업을 산출하고, JRE를 벗어나는 어떤 패키지에도 의존하지 않는) 이다.

SPECjbb2005는 메모리 레지던트(resident) 이고 디스크에 I/O는 수행하지 않으며, 단지 지역 네트워크 I/O만 가지고, think times를 가지지 않는다.

SPECjbb2005결과물은 SPECjbb2000 이나 TPC-C의 결과물과 비교될 수 없다. Comparisons to either이들 중 어떤 것과의 비교는 SPEC의 실행과 TPC 의 공정 사용 정책 규정의 보고에 대한 심각한 위배이다. "위배 시 페널티를 받게 된다."

⊙ SPECjbb2005 벤치마크 성능 측정치

SPECjbb2005는 2개의 성능 측정치를 가지고 있다. 하나는 SPECjbb2005 bops (business operations per second)로서 초당 수행되는 비즈니스 운영의 전체 비율이고, 또 하나는 SPECjbb2005 bops/JVM으로 벤치마크 동안 실행되는 JVM의 개수로 SPECjbb2005 bops를 나눈 것이다.

SPECjbb2005에서 2개의 측정치가 사용되는 이유는, SPECjbb2005 bops 측정치는 하나의 벤치마크 실행에서 전체의 JVM으로 달성되는 총 결과 치로 측정되기 때문이다. SPECjbb2005 bops/JVM 은 전체 측정치에 대한 단일 JVM의 공

헌도이며, 그러므로 이것은 단일 JVM의 성능 및 범위(scaling)의 성능 측정치이다.

측정치 계산이 실제 최고치 보다 예상 최고치에 의존하는 이유는 예상 기대치가 좀 더 크게 개발되고, 측정 기간에 특별히 System.gc() 호출이 없게 개발된 시스템으로, 주어진 시스템에서 발생하는 SPECjbb2005의 최고 정점에서의 웨어하우스 개수는 좀 더 적정성을 가질 것이다. 좀 더 나은 예측력을 허용하기 위해 SPECjbb2005 bops metric은 예상 최고치를 사용하여 계산되었다.

독립-JVM 측정치은 아래와 같이 계산된다.

"포인트"는 주어진 웨어하우스 숫자에서 수행된 업무의 측정 기간을 나타낸다. 전체 벤치마크 실행은 증감되는 웨어하우스 숫자를 가진 측정 포인트의 순서로서 구성된다. (그리고 스레드의 증감 숫자) SPECjbb2005의 컴파일 운영 측정치는 다음과 같이 계산된다.

·모든 포인트(웨어하우스 개수)는 1부터 MaxWh(이 숫자는 SPECjbb.props내의 웨어하우스의 input.ending_number 에 의해 지정된다) 웨어하우스 까지 실행된다. 최소 값에, 1부터 8까지 모든 포인트는 실행되어 진다.

·1부터 웨어하우스의 예상 최고 개수(실험되어지기 위해 예상되어진 예상최고치는 SPECjbb.props내의 input.expected_peak_warehouse에 0의 값이 형태(nonzero)o로 명시되거나 또는 0으로 설정되는 경우는 Runtime.getRuntime.availableProcessors()에 기술된다)까지의 웨어하우스의 개수의 경우는, 측정 기간은 30초이다. 웨어하우스의 개수가 좀 더 클 경우는 측정 시간은 4분이다. 모든 JVM 인스턴스는 이 측정기간과 동일하다.

·M웨어하우스 개수부터 2M까지의 웨어하우스 내의 모든 포인트의 경우의 모든 인스턴스의 처리율은 평균으로 산출한다. 이 평균이 SPECjbb2005의 측정치인 'SPECjbb2005 bops'이다. 2.3 섹션의 실행 및 보고 규정에서 설명한 것처럼, 2M 웨어하우스까지의 모든 포인트를 실행하지 않은 시스템으로 부터의 산출된 결과물도 여전히 유효한 것으로 간주된다. M+1부터 2M의 범위 내 모든 측정 포인트들은 0 SPECjbb2005 bops에서 계산되었다.

·SPECjbb2005의 두 번째 측정치인, SPECjbb2005 bops/JVM은 실행에 사용되어진 JVM 인스턴스 개수로 SPECjbb2005 bops 수치를 나누어 획득한다. SPECjbb2005에 내재된 보고 도구는 수평축에 웨어하우스의 포인트를 계산하고, 수직축에 처리율 합으로 구성된 그래프를 산출한다. 1부터 8 혹은 2N의 최소치 내의 모든 포인트 들은 실행되어지고 보고되어지기를 요구되어진다. M+1부터 2M의 범위 내에서 간과된 포인트들은 0 SPECjbb2005 bops의 처리율로 보고 될 것이다. 측정치의 평균 포인트는 리포트에 표시될 것이다.

⊙ SPECjbb2005의 의의

SPECjbb2005가 OLTP 벤치마크의 완성품이 아닌 반면에, 이것은 거대 비즈니스 애플리케이션의 표준적이다. 이것은 또한 자바 플랫폼의 기능 및 성능 테스트에 훌륭하다. 이것은 자바가상머신(JVM), 직시실행 컴파일러(JIT), 폐영역 회수, 스레드, 자바 라이브러리의 부분, 그리고 운영체제의 일부분을 효과적으로 실행된다. 이 벤치마크는 CPU, 캐쉬, 메모리 계층의 성능 및 공유 메모리 처리 시스템의 범위성을 측정한다.

10) 분석결과

⊙ TPC-E

▪ 주요 현황 및 특성

TPC-E 성능평가모델은 2006월 8월까지 Draft Version으로 소개되고 있기 때문에, 아직 완성된 평가모델이라 할 수 없다. 그러나 현재에 발표된 Draft Version0.32d를 기준으로 분석한 결과에 의하면, TPC-E 평가모델은 기존의 TPC-C 평가모델과 비교하여 보다 현실 지향적인 OLTP 애플리케이션 환경을 기반으로 하고 있으며, 성능평가를 위해서 필요한 요소들을 더욱 폭넓게 반영하고 있다. TPC-E 평가모델은 Release상태가 아니기 때문에, 실제 적용 현황에 대해서는 구체적으로 언급할 수는 없으나, 현재의 Draft Version에 대한 특징을 정리하면 다음과 같다.

▪ 현실 지향적 OLTP 애플리케이션 환경

TPC-C의 경우 9개의 테이블과, 5개의 트랜잭션을 대상으로 하고 있는 반면, TPC-E의 경우 33개의 테이블과 12개의 트랜잭션을 대상으로 하고 있다. 또한 트랜잭션 처리과정에서 발생할 수 있는 시스템중단 혹은 오류와 같은 문제를 해결하는데 소요되는 시간까지를 포함하고 있기 때문에, 보다 현실에 가까운 환경을 대상으로 하고 있는 것이 특징이다.

▪ 성능평가를 위한 부하 측정단위 변경

TPC-C와 비교하여 분당 처리되는 트랜잭션 수를 평가기준으로 하는 것이 아니라, 초당 처리되는 트랜잭션 수를 기준으로 하고 있다. 이는 성능평가의 기준이 되는 시간단위 변경 이상의 의미를 가지며, 다른 평가요소에 어떠한 영향을 미칠 것인지를 고찰할 필요가 있을 것이다.

▪ 성능평가 기준변경

TPC-E 평가모델에서의 가장 큰 특징은 성능평가 기준의 변환이다. 먼저 Performance Metric의 경우 분당 처리되는 트랜잭션수에서 초당 처리되는 트랜잭션수로 변경되었으며(tpmC->tpsE), Performance Metric 성능평가 기준이 변경에 따라 이에 기반을 둔 Price/Performance Metric 단위가 변경된 점이 가장 큰 특징 중의 하나이다.

▪ 데이터베이스 부하측정을 위한 시스템운영 시간 변경

데이터베이스 크기 및 처리되는 트랜잭션 수 등에 직접적으로 영향을 미치는 시스템운영시간을 달리 규정하고 있다.

▪ 데이터베이스 규모 산정식 변화

데이터베이스에 걸리는 부하의 정도를 측정할 수 있는 일일 증가하는 데이터베이스 크기를 측정함에 있어서 TPC-E의 경우 TPC-C에서와는 다른 산정방식과 저장 공간에 대한 다른 개념을 도입하여 산정하고 있다.

▪ 기존 관련 모델과의 비교

- TPC-C 개요

앞 절에서 기술한 바와 같이 TPC-E의 목적은 OLTP 애플리케이션 환경에서 트랜잭션이 수행되는 동안 시스템에 영향을 미치는 다양한 요인들을 고려하여 정보시스템 성능수준을 평가하려는 것이며, 이를 위해서 실제와 유사한 비즈니스 환경을 구축하여 가상의 트랜잭션을 수행시켜 봄으로써 트랜잭션이 성능에 영향을 미치는 요인과 정도를 측정한다. 그러나 트랜잭션이 미치는 영향을 측정하기 위해서 구축된 비즈니스 모델이 모든 OLTP환경을

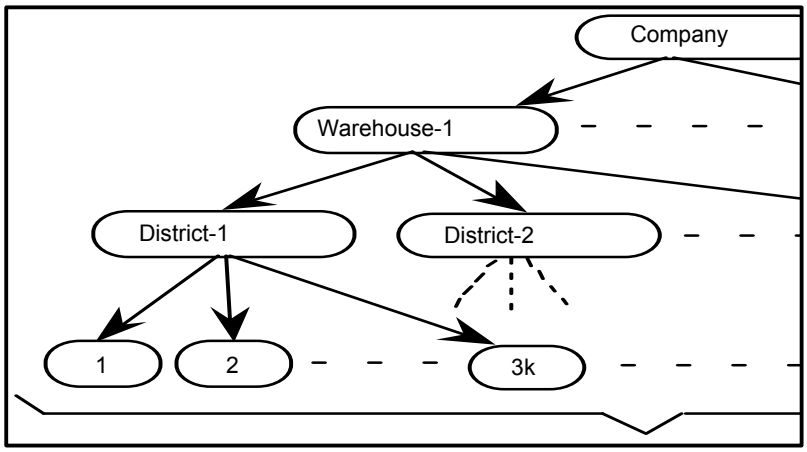
포함하지 못하기 때문에, 구축된 비즈니스 모델에 따라서 측정 결과가 달리 산출되는 것은 당연하다 할 것이다.

1992년 Draft 작업을 마무리하고 2006년 4월 TPC가 발표한 TPC-C Revision5.7은 TPC-E와는 다른 비즈니스 환경을 가정하여 벤치마크를 실행하고 있는데, TPC-C를 정리하면 다음과 같다.

⊙ TPC-C 비즈니스 환경과 데이터베이스

TPC-C 비즈니스 및 애플리케이션 환경은 대규모 유통업체의 주문처리 프로세스 처리 환경을 내용으로 하고 있다.

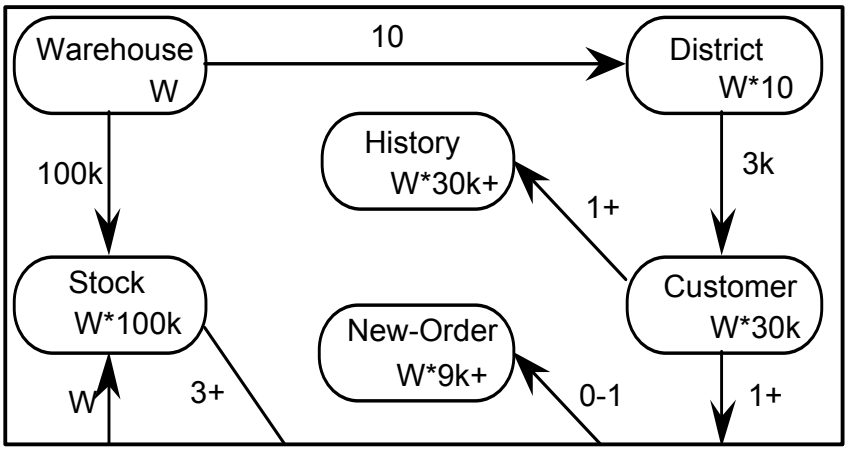
대규모 유통업체는 지리적으로 분산된 판매처와 이와 관련된 물류센터를 가지고 있는데, 비즈니스 확장에 따라서 새로운 판매처와 물류센터를 신설할 수가 있으며, 개별 물류센터는 10개의 판매처를 커버할 수 있고 개별 판매처에서는 3,000명의 고객에게 서비스할 수 있는 환경이다. 또한 모든 물류센터에는 100,000개의 아이템에 대한 재고를 확보하고 있다.



<그림41> TPC-C 비즈니스 환경

TPC-C 비즈니스 환경에서 주된 업무는 고객의 신규 주문 혹은 이미 주문한 내용을 문의하고 확인하는 것이며, 모든 주문은 평균 10라인 주문아이템으로 되어 있고, 그 중에 1% 정도는 현지 물류센터에 재고가 충분하지 않아서 다른 지역의 물류센터로부터 주문제품을 운송하여 고객에게 인도해야 하는 환경으로 가정되어 있다. 이러한 환경에서 TPC-C가 구축한 주된 애플리케이션은 고객 배송을 위한 주문처리 시스템, 고객이 대금을 지불하기 위한 지불내역 입력시스템, 그리고 배송가능날짜를 확인하기 위해서 재고수준을 파악하기 위한 시스템으로 이루어져 있다.

이러한 애플리케이션 환경의 시스템 구현을 위해서 TPC-C 데이터베이스는 다음과 같이 9개의 테이블로 구성되어 있으며, 개별 테이블 간의 관계는 다음과 같다.



<그림42> TPC-C 데이터베이스 테이블 사이의 관계

⊙ TPC-C 데이터베이스 트랜잭션

TPC-C 데이터베이스 트랜잭션은 데이터베이스에서의

SELECT, RETRIEVE와 같은 작업 단위를 의미하며, TPC-C에서는 다음과 같은 트랜잭션으로 구성되어 있다.

- 신규 주문처리 트랜잭션
- 대금 지불 트랜잭션
- 주문처리(내역) 확인 트랜잭션
- 배송처리 트랜잭션
- 재고수준확인 트랜잭션

⊙ TPC-C 데이터베이스 SCALING AND POPULATION

TPC-C 초기 데이터베이스 크기는 다음과 같으며, 이를 기준으로 다음과 규칙에 의해서 데이터베이스 크기가 증가하게 된다.

<표40> 초기 데이터베이스 크기

테이블 명	크기(행)	테이블길이(바이트)	테이블크기(1,000 바이트단위)
WAREHOUSE	1	89	0.089
DISTRICT	10	95	0.950
CUSTOMER	30K	655	19,650
HISTORY	30K	46	1,380
ORDER	30K	24	720
NEW-ORDER	9K	8	72
ORDER-LINE	300K	54	16,200
STOCK	100K	306	30,600
ITEM	100K	82	8,200

위와 같은 초기 데이터베이스를 기준으로 60일간 처리되는 트랜잭션 결과를 저장할 수 있는 공간이 요구되며, 일일 증가되는 데이터베이스 크기는 다음과 같이 산정한다.

Daily Growth = (dynamic-space / (W*62.5)) * tpmC

* dynamic-space: HISTORY, ORDER, ORDER-LINE 테이블과 같이 트랜잭션이 처리결과가 저장되는데 소요되는 저장 공간

* 62.5: normalized factor

⊙ TPC-C Performance Metrics

TPC-C 성능평가 기준은 SUT와 DRIVER 사이에서 데이터 입•출력이 성공적으로 수행됨으로써 완전하게 처리된 트랜잭션(Completed Transaction)을 기준으로 하며, 측정기간은 트랜잭션이 지속적으로 처리되고 있는 상태인 steady state를 대상으로 한다. 또한 TPC-C Performance Metrics와 관련하여 5개의 트랜잭션별로 요구되는 트랜잭션 믹스 율, 대기시간, 응답시간 등의 제한요구사항 등이 적용되는데, 다음은 이러한 성능평가에 적용되는 제한사항들을 정리한 것이다.

<표41> 성능평가 관련 제한 요구사항

Transaction type	Min. % of mix	Min. Keying Time	90% Response Time	Min. Mean of Think time Distribution
New-Order	n/a	18sec.	5sec.	12sec.
Payment	43.0	3sec.	5sec.	12sec.
Order-Status	4.0	2sec.	5sec.	10sec.
Delivery	4.0	2sec.	5sec.	5sec.
Stock-Level	4.0	2sec.	20sec.	5sec.

TPC-C 표준과 TPC's Use Policies, Guideline 등 모든 TPC-C 관련하여 사용되는 성능평가 기준은 다음과 같다.

- Performance Metric: TPC-C Max. Qualified Throughput rating (=>tpmC)
- Price/Performance Metric: 전체 3년 동안의 가격을 Max. Qualified

Throughput rating으로 나눈 값 (=> price/tpmC)

⊙ TPC-C와 TPC-E의 비교분석

TPC-C와 TPC-E의 비즈니스 및 애플리케이션 환경이 다르기 때문에, 두 성능평가 모델에서 사용되고 적용되는 수치를 절대적으로 비교하는 것은 큰 의미를 갖지 않는다. 그러나 두 성능평가 모델의 궁극적인 목적이 가상으로 구현된 OLTP환경에서 데이터베이스에 걸리는 작업부하의 정도를 측정하려는 것이기 때문에, 데이터베이스에 걸리는 부하의 정도와, 이를 측정하는 기준과 방법을 비교함은 구체적인 성능평가를 위해서 큰 의미를 가질 것이다. 우선 두 성능평가 모델의 차이를 비교하면 다음과 같다.

- Primary Metric 측정단위: TPC-C의 경우 분당 처리되는 완전한 트랜잭션 수(=> max. completed(qualified) Transaction-Per-Minute-C: tpmC)를 기준으로 하는 반면, TPC-E의경우에는초당처리되는유효한 트랜잭션 수(=> valid Transaction-Per-Second-E: tpsE)를 기준으로 하고 있다. 이는 측정단위를 1분으로 하느냐? 1초로 하느냐의 단순한 차이를 넘어 측정단위를 이용하여 산정되는 모든 항목(소요 공간, 처리속도 등)에 영향을 미치기 때문에, 단순한 측정단위의 변경 이상의 의미를 갖는다. 또한 TPC-C의 경우 측정대상인 트랜잭션의 상태를 트랜잭션에 따라 달리 정의하고 있는 반면, TPC-E에서는 트랜잭션 상태를 DRIVER와 SUT와의 관계를 통해서 유효한 트랜잭션으로 정의하고 있음도 측정대상이 달라질 수 있기 때문에, Primary Metric을 측정하는데 큰 영향을 미친다.
- Price/Performance Metric : Price/Performance Metric은 TPC-C와 TPC-E에서 동일하게 3년 동안의 가격을 Performance Metric으로 나눈 값으로 정의하고 있기는 하지만, TPC-E의 경우 Performance Metric의 초당 유효하게 처리되는 트랜잭션을 기준으로 하기 때문에, Price/Performance Metric 결과도 다르게 측정된다.
- Keying/Think Times 항목 삭제 : TPC-E의 경우 TPC-C 결과보고서

에 추가된 항목 중에 Keying/Think Times(in Second) 항목이 삭제되어 트랜잭션을 처리하는 과정의 여유 시간적 요인들을 성능평가 요인으로 간주하지 않고 있다. 이는 측정 환경항목 요인의 단순 추가 혹은 삭제의 문제를 넘어서 이로 인해서 트랜잭션 처리에 소요되는 시간 등 다른 측정항목에 영향을 미칠 것이기 때문에, 전체적인 성능평가에서 중요하게 다루어져야할 요소 중의 하나이다.

- Transaction Mix : TPC-C 데이터베이스 트랜잭션과 TPC-E 데이터베이스에서 처리할 트랜잭션이 다르기 때문에(비즈니스 환경과 애플리케이션 환경이 다름으로 인해서), 트랜잭션의 성격에 따라서 믹스 율이 다름은 당연한 결과이지만, 단순 믹스 율뿐만 아니라, 유효하게 처리되는 트랜잭션 수를 병기함으로서 결과적으로 테이블 규모가 증가하지는 않지만 부하가 걸리는 트랜잭션까지도 고려할 수 있는 여지를 포함하고 있다는 차이가 있다.
- Test Duration and Timing : TPC-C의 경우, 테스트 시작에서 트랜잭션이 처리되기 시작한 ramp-up time과, 측정시간 등 모든 시간을 분 단위로 계산하는 반면, TPC-E에서는 기본 측정단위가 초 단위이기 때문에, 모든 시간을 초단위로 계산하는 차이가 있으며, 또한 TPC-E에서는 트랜잭션이 처리되는 과정에서 발생할 수 있는 시스템중단 혹은 오류와 같은 문제를 해결하는데 소요되는 시간까지도 포함하고 있어서 보다 구체적인 평가를 할 수 있다는 차이가 있다.

⊙ TPC-E의 시사점과 적용방안

TPC-C와 TPC-E의 비교분석 결과 가장 두드러진 차이점은 성능평가를 위한 측정단위가 분당 처리되는 트랜잭션 수에서 초당 처리되는 트랜잭션 수로 변했다는 것이다(tpmC -> tpsE). 또한 측정대상이 최대 완전하게 처리되는 트랜잭션 수에서 유효하게 처리되는 트랜잭션 수로 변했다는 것이다(max completed(qualified) Transaction -> valid Transaction). 이 같은 측정대상과 측정시간

단위의 변화는 우선 양적인 수치의 변화를 의미한다. 기본적인 비즈니스 및 애플리케이션 환경이 변하지 않는다고 하더라도 측정대상과 측정시간단위의 변화는 데이터베이스 크기, 트랜잭션처리에 소요되는 시간(response time, pacing delay time)에 영향을 미치기 때문에, TPC-C에서와는 다른 산정방식이 요구된다.

또한 TPC-E에서 새로이 추가된 Business Recovery Timing 항목들은 시스템 여유 율이나 데이터베이스 소요 공간, 기타 보정값에 영향을 미치게 되며, 따라서 최종 산정식에 영향을 미칠 것으로 예상된다. 이 이외에도 기존의 TPC-C에서는 Keying and Think Time을 트랜잭션 처리에 포함시켜 고려하였지만, TPC-E에서는 이를 삭제하였기 때문에, 트랜잭션 수가 근본적으로 달라질 것이다. 이를 고려한다면 기본 보정 값, Peak Time에서의 보정값은 물론 OLTP환경의 기타 산정항목에도 영향을 미치기 때문에, 전체 산정식 결과는 달라질 것이다.

TPC-C와 TPC-E를 비교분석한 결과를 토대로 종합적으로 판단할 때, TPC-E에서 새로이 추가/변경된 내용을 고려하여 기존의 산정식을 수정하기에는 추가 및 변경해야할 항목이 많고, 특히 새로운 정보기술의 발전과 OLTP 애플리케이션 환경변화를 고려할 때, 기존의 TPC-C에 의한 산정방식을 재검토할 필요가 있으며, 경우에 따라서는 TPC-E를 이용한 새로운 산정방식을 도출하는 것이 보다 합리적일 것이라 판단된다.

⊙ TPC - App

▪ 주요 현황 및 특성

TPC-App 벤치마크는 TPC(Transaction Performance Processing Council)에서 TPC-W를 대체하기 위하여 새롭게 제안

한 벤치마크 표준으로서 데이터베이스 서버, 웹 서버, 여타 어플리케이션 서버들과 데이터를 교환하는 전형적인 전자상거래 작업을 서버가 중간에서 얼마나 잘 수행하는가를 측정하게 되고, 점수는 1초당 처리한 웹 서비스의 수로 처리가 된다.

TPC-App는 2개의 성능 측정 기준이 있으며, 첫 번째는 어플리케이션 서버 시스템마다의 SIPS(Web Service Interactions per second)이고 두 번째는 전체 테스트 설정(SUT(System Under Test))에서의 통합 SIPS이다.

SIPS를 측정하여 어플리케이션 서버의 성능을 평가할 수 있도록 제안된 벤치마크이다. 그리고 어플리케이션 서버에서 MS의 닷넷과 선의 자바 등 경쟁 기술들을 비교할 수 있게 해준다.

테스트 비용이 비싼데다 가동할 때 서버를 25대까지 필요로 하고 또한 어플리케이션 서버 성능만이 아니라 다른 서버의 성능까지 테스트하는 등 문제점이 많았던 TPC-W를 대체하게 되며, TPC-App의 개발에는 IBM, MS, HP, BEA시스템스, 오라클, 델, 유니시스, AMD, 인텔 등이 참여했다.

- 기존 관련 모델과의 비교

– TPC-W 벤치마크

TPC(Transaction Performance Processing Council)에서 TPC-W 벤치마크를 대체하기 위하여 새롭게 제안한 TPC-App 벤치마크는 Application Server와 Web Service의 성능 측정을 위한 벤치마크 표준이다.

TPC-W 벤치마크는 인터넷 e-Commerce 비즈니스 환경에서 서버의 성능이나 성능 대비 비용에 대한 정확한 측정을 위해 2000

년 2월에 제안되었다.

기존의 SPECWeb이나 TPC-C 벤치마크가 H/W 벤더 사에서 구성한 e-Commerce 비즈니스 환경에서의 측정 결과이고 복잡한 e-Commerce 비즈니스 환경을 대표하기에는 한계가 있었기에 새로운 벤치마크 표준이 제안되었다. TPC-W 벤치마크에서 구성하는 e-Commerce 벤치마크 Workload는 웹 서버, 웹 캐쉬 서버, 이미지 서버, DB 서버 등 다양한 종류의 서버로 구성되어 있으며, 소매상점 웹 사이트의 행위를 시뮬레이션 하게 된다.

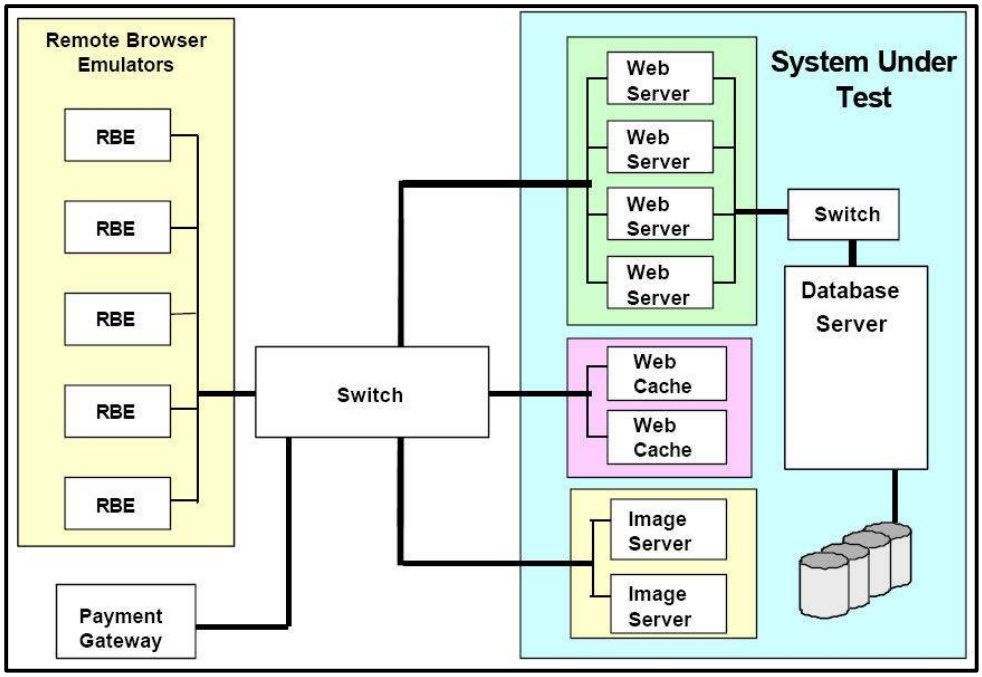
성능을 측정하기 위한 주요 지표는 초당 Web Interaction의 숫자인 WIPS(the number of Web Interation Per Second) 비율, WIPS 당 시스템 비용 등이다.

- TPC-App와 TPC-W의 비교

TPC-App 벤치마크는 데이터베이스 서버, 웹 서버, 여타 애플리케이션 서버들이 서로 통신하는 전형적인 작업을 얼마나 잘 수행하는지 측정하게 된다. TPC-App 벤치마크의 주요 지표는 초당 웹 서비스 페이지 당 응답 시간으로 측정되며, 애플리케이션 서버에서 사용되는 두 가지 경쟁 기술인 MS의 닷넷과 선 마이크로시스템즈를 비롯한 연합 세력들의 자바를 서로 비교하는 기준으로 이용될 수 있도록 제안되어 있다. TPC-App 벤치마크는 기존의 TPC-W 벤치마크를 대신하게 된다. TPC-W 벤치마크는 몇 가지 결점이 있었다. 일단 벤치마크를 수행하기 위해 보통 25대의 서버가 필요할 정도로 비용이 많이 들었으며, 애플리케이션 서버 성능뿐 아니라 정보를 저장하는 서버의 성능은 물론 다른 보조 머신은 성능측정에 포함되지 않아서 TPC-W 벤치마크과정은 상당히 포괄적이어서 측정결과를 적용하고 반영하는데 어려움이 있었다.

TPC-W 벤치마크는 다양한 종류의 서버를 구성해야 한다는 큰 제약사항이 있다. 실제 국내의 대규모 트래픽이 발생하는 포털사이트 등에서나 볼 수 있는 캐시 서버나 이미지서버는 국내의 일반적인 웹 서비스 환경에서는 쉽게 구성하기 어려운 환경이라고 할 수 있다.

TPC-W 벤치마크의 SUT 환경은 다음 <그림43>과 같다.



<그림43> TPC-W 벤치마크 SUT 구성

이러한 TPC-W 벤치마크의 환경 구성에 많은 서버가 필요하고 이에 따른 비용 부담 등의 이유로 TPC에서는 TPC-App 벤치마크 표준 규격을 제안하였고, TPC-App의 SUT 환경은 Web Server, Application Server, DB Server로 구성되며, 실제 국내 대

부분의 웹 서비스 환경과 구성이 유사하다고 할 수 있다.

⊙ 적용방안

TPC-W 벤치마크나 TPC-App 벤치마크 모두 H/W의 성능과 가격대비 성능 등을 비교 할 수 있도록 도와준다. 또한 TPC-App 벤치마크에서 제안하는 SUT 환경은 기존의 타 벤치마크에 비해 매우 현실적으로 제안되고 있다. TPC-App 벤치마크의 가장 중요한 지표인 SIPS(Web Service Interaction per Second)는 H/W 제품군에서 동일한 사양의 서버에서의 성능비교를 위해 사용할 수 있을 것으로 전망된다. 또한, 한정된 예산 내에서 적절한 H/W를 도입해야 하므로, 성능대비비용을 비교하는데 사용하여 예산이나 비용의 효과적인 집행에 도움을 줄 수 있을 것으로 예상된다.

그러나 실제로 현업에서는 H/W간의 성능평가도 중요하지만, 무엇보다 업무나 서비스에 근거하여 필요한 기능 요구사항들이 구체화되고 이를 근거로 하여 이러한 기능 요구사항들이 원활하게 서비스 될 수 있도록 필요한 H/W의 용량산정이 필수적이지만, 현실적으로 필요한 용량산정이 정량적으로 접근하기 보다는 업무 담당자들의 경험에 근거하여 용량을 산정하게 된다.

기존에 몇몇 시스템 통합업체 등에서 제안되던 TpmC를 추정하는 방식 역시 사용자의 추정치에 근거를 하거나 Web 서버나 WAS 서버의 용량산정의 유일한 지표는 전체사용자 및 동시사용자를 근거로 하여 용량을 산정하기에 구축 완료 이후 터무니없이 큰 용량의 서버로 예산이 낭비되거나, 용량 부족으로 서비스에 문제가 발생하는 경우가 자주 발생하였다. 또한, TPC-App 벤치마크는 Web Service를 기반으로 하고 있다. 현재 국내 공공부문 및 민간 기업에서도 Web Service를 도입하고 있으며, 이미 많은 서

비스와 시스템이 Web Service로 구축되고 있지만, 아직까지 Web Service가 보편화되어 있다고 보기 어렵다.

향후 TPC-App 벤치마크 방법론을 참고하여 일반적인 서비스나 정보시스템의 성능평가 모델 개발이 가능할 것으로 전망되며, 제품 구매나 용역 사업 발주 시에 활용방안이 함께 제공된다면, 조만간 Web Service가 보편화되고 확산될 때 효과적으로 사용될 것으로 전망된다.

11) SPECWeb2005

본 절에서는 다양한 버전으로 발달되어 온 SPECWeb의 기존 버전들과의 비교 분석을 통해 서버 성능평가에 적절한 적용 방안을 제시하고자 한다.

⊙ 주요 현황 및 특성

SPECWeb2005는 세 가지로 분류된 평가 기준을 도입함으로써 다양하게 제공되는 웹 서비스를 평가하기 위한 기준을 제시하고 있다. 이전 버전의 웹 서버 벤치마크에서 일괄적으로 평가하는 방법에서 보다 상세한 평가 기준으로 보다 구체적이고 명확한 웹 서버 성능을 평가할 수 있는 방법을 제시하고 있다. 즉, Banking, Ecommerce, Support로 각각 workload design으로 설계함으로써 서버의 성능을 실 상황에 적합하게 평가 한다는 것이다.

▪ SPECWeb2005의 기술적 특성

SPECWeb2005의 핵심은 Banking workload design, Ecommerce workload design, Support workload design 으로 분류

하여 벤치마크를 실행한다는 것이다. 이와 같은 분류를 통해 다양한 웹 서비스 환경에서 서버를 평가할 수 있도록 지원한다. 분류의 특징은 보안과 다양한 알고리즘을 적용한 복잡한 서비스 환경을 평가하는 Banking에서부터 단순한 데이터 제공을 목적으로 하는 Support 서버에 이르기 까지 차별화된 설계로 구성되며, 상세한 내용은 다음과 같다.

우선 SPECWeb2005 Banking Workload Design은 SPECWeb2005 Workload Design중 가장 복잡한 구조로 설계된 것으로 은행 업무에 대한 상황을 근거로 웹서버 로그에 대한 연구를 기반으로 한다. SPEC은 Banking에 관련된 철저한 분석을 위해 텍사스 지역에 근거한 은행에서 웹서버 로그에 대한 실질적인 데이터를 이용하여 분석하고, 은행업무의 특성상 기밀성이 요구되는 데이터들을 분석하기 위해 임시의 거래를 가상으로 실시하여 은행업무 웹 사이트에 처리를 요구함으로써 자료 정보를 분석하였다.

- SPECWeb2005 Banking Workload Design의 Data Summary

SPECWeb2005 Banking Workload Design은 Data Summary를 분석하여 기존 SPEC99의 분석 데이터의 문제점을 파악하여 실질적인 은행 업무에 적합한 응답 코드를 수집하도록 기능을 향상 시켰다. HTTP 응답 코드를 보면 SPEC 99와 실제 은행 데이터 로그 사이에 많은 차이가 발생하는 것을 다음 <표42>에서 확인 할 수 있다.

<표42> HPPT Response Codes

Response Code	Percentage in Banking Data logs	Percentage in SPECWeb99
200 OK	68.1	100
206 Partial Content	0.12	0
302 Found	4.5	0
304 Not Modified	27.05	0
403 Forbidden	0.03	

다음 <표43>은 SPECWeb99에서 대표되는 은행업무 서버 로그에서 보여주는 파일 사이즈를 비교한 것이다. 이 목록은 정적인 파일들로만 구성되며, 동적인 크기의 파일을 요청하지 않는다. SPECWeb99는 0-100byte 사이의 크기를 가진 파일을 요청하지 않고 은행 업무상에서 나타나지 않는 대용량의 파일을 요청하지 않는다. 하지만 Banking Data의 분석 값을 보면 SPECWeb99의 응답 값과 많은 차이가 나타나는 것을 확인 할 수 있다.

<표43> Workload File size distribution for requests

Size Range	Class	Banking Data	SPECWeb99
0-100 B	-1	59.20%	0.00%
100-1KB	0	34.00%	35.00%
1KB-10KB	1	6.30%	50.00%
10KB-100 KB	2	0.50%	14.00%
100KB-1MB	3	0.00%	1.00%

보다 정교하고 실질적인 파일전송 내용을 측정하기 위해 실질적인 은행 업무에서 사용되는 파일과 SPECWeb99에서 보인 파일을 비교해 SPECWeb2005 Banking Workload Design 에서는 세부적인 내용을 분류하고 실제 Banking Data를 모두 측정할 수 있게 지원

하도록 설계 되었다.

▪ SPECWeb2005 Banking Workload Design의 Dynamic Scripts

SPECWeb2005 Banking Workload Design은 17개의 Dynamic Script로 구분한다. 그 구분 내용은 다음 <표44>와 같다.

<표44> Dynamic Scripts

Dynamic Script Number	Dynamic Script	Size of Returned Page
0	Login Welcome	4KB
1	Account Summary	17 KB
2	Check Detail Output	11 KB
3	Bill Pay	15 KB
4	Add Payee	18 KB
5	Post Payee	34 KB
6	Quick Pay	22 K
7	Bill Pay Status	24 K
8	Profile	32 K
9	Change Profile	29 K
10	Order Check	21 KB
11	Place Check Order	25 KB
12	Transfer	13 KB
13	Post Transfer	16 KB
14	Logout	46 KB
15	Check Image Request (front)	NA
16	Check Image Request (back)	

이와 같은 구분은 은행 업무를 사용할 때 사용자가 어떤 서비스를 이용할 것인가를 예측하는 자료로 Markov chain<확률적 의사결정 다이어그램> 공식을 적용하여 <표45>와 같은 확률 값을 추출 하고, 은행 업무와 관련된 벤치마크를 실행할 때 확률을 적용함으로써 보다 정확한 서버의 성능을 평가하게 된다.

<표45> Relative frequencies (normalized to 100%) of each of the dynamic scripts

Dynamic Script Index	Name of the script	Relative Probability % (computed from the proposed Markov Chain)
0	Login Welcome	21.53
1	Account Summary	15.11
2	Check Details	8.45
3	Bill Pay	13.89
4	Add Payee	1.12
5	Post Payee	0.80
6	Quick Payment	6.67
7	Bill Pay Status	2.23
8	Change Profile	1.22
9	Post Changed Profile	0.88
10	Order Checks Request Form	1.22
11	Place Check Order	0.88
12	Transfer Money form	1.71
13	Post Transfer Money	1.22
14	Logoff	6.16

▪ SPECWeb2005 Ecommerce Workload Design

SPECWeb2005 Ecommerce Workload Design은 컴퓨터시스템 판매(컴퓨터 상품 전자상거래)의 웹서버를 가정하여 설계되었으며, 사용자가 상품을 찾고, 둘러보고, 주문(요구)하고 상품을 구매하는 과정을 포함한다. Support workload는 유명한 Web store(ex. 컴퓨터 부품 및 완제품을 판매하는 인터넷 쇼핑몰) 등의 실제 Ecommerce 사이트의 평균적인 페이지 사이즈, 이미지사이즈, 접속빈도 (브라우저의 캐싱 때문에 재접속 되는 것 또한 감안하여) 등을 로그파일의 수집을 통해 분석하고, 사용자가 물건을 구매할 때 일반적으로 작성하게 되는 폼 데이터 등을 수집함으로 인해서 더욱 발전하였다.

▪ SPECWeb2005 Ecommerce Workload Design의 구성

일반적인 Ecommerce Web stores에서 요구되는 점을 조사해보면 Ecommerce Transaction에 있어서 명확히 세 가지 파트로 구분됨을 알 수 있다.

① 브라우즈

브라우즈(웹브라우저를 통하여 사이트에 처음 접속해서 시작하는 부분) 부분은 사용자가 홈페이지를 방문하고, 국가, 지역 등을 선택하고 (대부분의 대형국제 인터넷몰에는 국가 및 지역선택을 할 수 있게 되어있어 지역별 페이지로 바로 이동되기도 한다.) 일반구매자인지 혹은 학생, 국가, 기업 등의 다양한 구매자 타입을 선택하고 구매자의 타입에 맞게 적절한 섹션으로 이동시킨다. 구매자는 자신이 관심 있는 상품타입을 고를 것이고 특정한 모델을 선택할 것이다. 특정상품을 찾기 위해 검색기능을 사용할 가능성도 있다.

② 상품의 조합

상품의 조합은 유저가 제안 또는 요청하는 부분이다. 우선 유저에 의하여 선택된 내부사항(CPU스피드, 메모리 용량, 하드용량 등), 다음으론 AS보증기간, 기타 Support 서비스, 그리고 기타추가 옵션(케이블, 소프트웨어 등)이다.

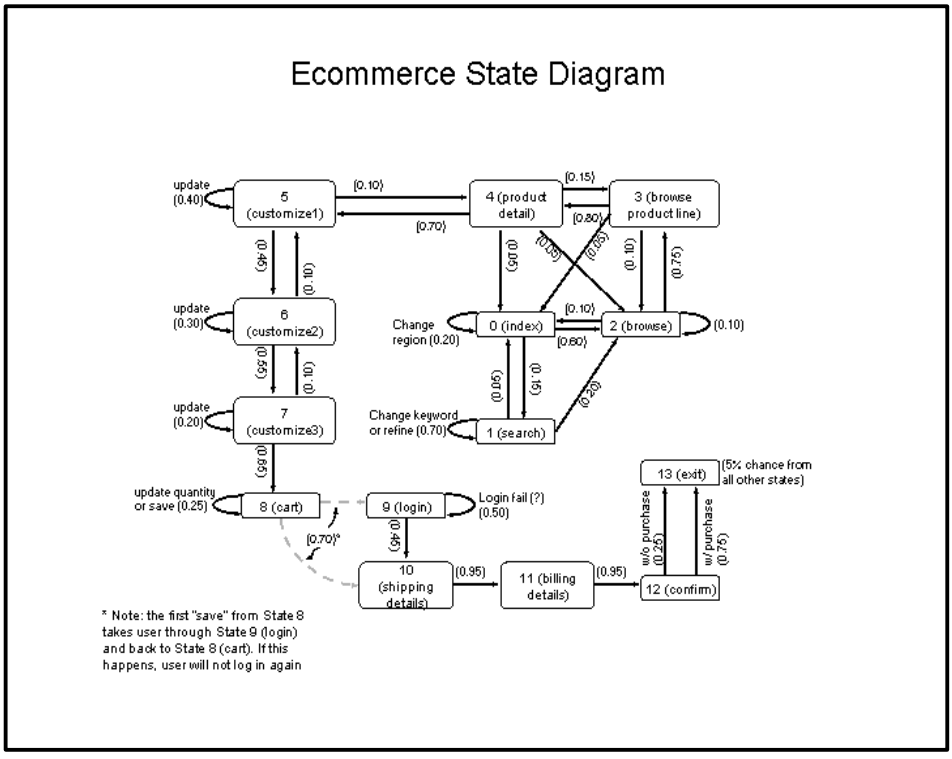
③ Ecommerce

Ecommerce는 실질적인 부분으로 분류된다. 우선 구매자가 그들이 원하는 조합을 선택하여 쇼핑카트에 올린다. 카트페이지는 유저가 선택제품을 추가, 교체 및 삭제하고, 카트내용을 세이브 하는 것이 가능해야한다. 유저가 “Checkout"을 클릭했을 때 세션은 보

안이 이루어져야한다. (이것은 HTTPS를 통해 이루어지는 것으로 초점을 맞추고 있다.) SSL 단계에선 다양한 페이지가 존재한다. (로그인/등록 , 지불 및 배송정보입력, 상세지불정보, 주문요구 및 수정, 또 이러한 것들에 대한 확인 등이 해당된다.)

- SPECWeb2005 Ecommerce Workload Design의 Markov Chain

SPECWeb2005 Ecommerce Workload Design역시 Markov Chain을 적용하여 사용자가 물품을 주문함에 있어서 거치게 되는 과정을 확률로서 벤치마크에 도입하여 평가한다. <그림44>는 Markov Chain을 Ecommerce 상태에 도입되어 적용된 그림이다.



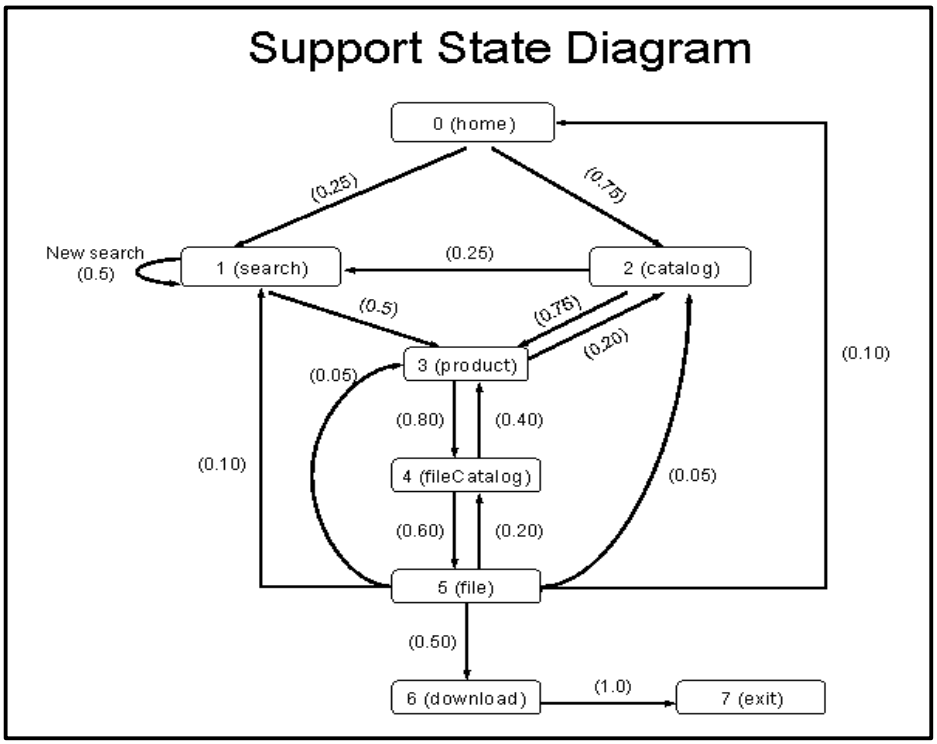
<그림44> Ecommerce State Diagram

▪ SPECWeb2005 Support Workload Design

SPECWeb2005 Support Workload Design은 Vendor's Support web site를 가정하여 설계되었다. 이용자는 특정제품(혹은 파일)등을 검색하고, 리스트를 보고, 각 항목에 따른 필터링이 가능하고, 또한 파일들을 다운로드 받을 수 있다. Support workload는 대형 공급사의 실제 회사들의 평균적인 페이지 사이즈, 이미지사이즈, 접속빈도 (브라우저의 캐싱 때문에 재접속 되는 것 또한 감안하여) 등을 로그파일의 수집을 통해 분석함으로써 더 향상되었다. Support workload 에서는 파일 다운로드를 위한 실제의 로그파일들에서 보여 지는 액세스 패턴들이 모델이 되었다. Support workload 클라이언트로부터 다이내믹 페이지가 요구되는 반면, 다른 두개의 Workload(뱅킹과 전자상거래)에 비하여 페이지가 더욱 심플해졌다. 예를 들어, across page의 request(세션쿠키 안에 저장되어야 될)가 계속 유지되어야하는 사용자 데이터가 없다. 그 대신, 이 Support workload에서는 대용량의 파일 다운로드가 강조된다. Support workload에는 보안컴포넌트(HTTPS)가 없기 때문에 주 서브시스템인 네트워크와 디스크에 주위를 기울여야 할 것으로 예상된다.

▪ SPECWeb2005 Support Workload Design의 Markov Chain

SPECWeb2005 Support workload Design역시 Markov Chain을 적용함으로써 하나의 상태에서 다른 상태로의 확률적 이동 가능성에 대하여 나타낸다. 다음은 Markov Chain을 적용한 Support State Diagram을 보여준다.



<그림45> Support State Diagram

Markov Chain을 적용하여 상황을 확률적으로 분석함으로써 사용자가 페이지를 이동하면서 발생하게 되는 페이지 요청(ex. 이미지) 및 파일 다운로드에 대해 보다 명확한 응답시간을 측정할 수 있는 지표로 사용된다.

⊙ 기존 관련 모델과의 비교

SPECWeb2005는 기존 99와 99_SSL과 완전히 다른 방식을 설계되었으며 평가 방법 또한 기존 버전들과 완전히 다르다. 즉, 새로운 개념으로 설계 및 개발 되었다.

- Workload Design

SPECWeb2005는 세 가지 workload 측정으로 세분화(Banking, Ecommerce, Support) 하였다. 본, 내용은 앞 절에서 상세히 설명하였다.

- 자바 코드 디자인

SPECWeb2005는 Client와 Prime Client의 이식성을 강화하기 위해서 자바코드를 이용하여 완전히 새롭게 작성하였다. 이는 운영체제에 제한받지 않고 설치함으로써 기존 시스템에 영향을 받지 않고 사용할 수 있다.

- 웹 페이지 기반 QOS(Quality of Service)

SPECWeb99처럼, SPECWeb2005 는 커넥션 스피드를 시뮬레이터 하기 위한 지정된 receive mechanism을 이용한다. 시뮬레이터된 연결 속도는 최대 100000 byte/sec로 증가하며, 이 비율은 광대역 전송의 대폭적인 채택으로 오늘날 일반화된 좀 더 빠른 연결 속도를 반영하기 위해 채택되었다. 기존 버전과의 가장 큰 변화는 SPECWeb2005 커넥션 기반 QOS로부터 웹 페이지 기반 QOS로 바뀐 것이다. 이는 기존 버전에서 측정하던 접속기반 QOS에서 웹 페이지 기반 QOS로 변화함으로써 사용자가 서비스를 요청할 때 서버가 얼마나 효과적으로 서비스를 제공하는지 평가 할 수 있게 되었다. 이는 기존 버전에서 같은 서비스를 얼마나 많은 사용자에게 제공할 수 있는가에 대한 평가에서 이벤트 기반 평가를 변화함으로써 보다 현실적인 환경에서의 서버 성능 평가를 측정할 수 있는 환경을 제공한다.

- User Think Time

SPECWeb2005 벤치마크의 다른 새로운 특징은 유저 세션 요청 사이에 일어나는 (예: workload 사이의"states") “think time"을 포함 시킨다. 즉, 서비스를 요청함에 있어 응답을 주기 전에 time을

둠으로써 실질적인 서비스 시 발생하는 처리 시간을 적용한다. 일반적으로 Banking에서의 User Think Time 은 10 sec을 적용한다.

▪ 브라우저 캐싱 분석을 통한 응답 값에 요청

SPECWeb2005는 서비스를 요청한 클라이언트의 브라우저 캐싱을 시뮬레이터 함으로써 기존에 제공된 이미지, 텍스트 등을 반복 요청할 때 캐시에 남아 있음을 서버에 알려줌으로써 중복된 데이터의 요청을 피한다. 이는 보다 명확한 서비스 응답 측정을 도와준다.

⊙ 적용 방안

SPECWeb2005는 웹 서비스를 제공하는 서버 측정에 있어 서비스 분류를 통한 벤치마크 활용을 고려해 볼 수 있을 것이다.

앞서 언급한 바와 같이 SPECWeb2005 벤치마크는 뛰어난 이식성, 웹 페이지 기반의 QOS를 지원하는 등 향상된 기술들은 당장의 서비스 평가 도구로써 활용 가능성이 매우 높다.

▪ 웹 서버 벤치마크 적용 방안

웹상에서의 서비스가 다양해짐에 따라 벤치마크 도구도 전문적인 분야를 벤치마크 하기 위한 도구가 필요하다. 즉, 웹상에서의 다양한 서비스를 제공하는 서버가 서비스 용도에 적합한지 평가할 수 있는 도구가 필요하다. 국내에서 서비스 되는 웹 서버는 크게 웹 서치, 데이터 제공, 뱅킹, 전자상거래, 개인화된 홈페이지 그리고 이러한 서비스를 하나로 포함하고 있는 포털 사이트들이 있다. 대규모의 포털 사이트 경우 각 서비스의 특징별로 서버를 분류하여 제공하고 있다. 이러한 국내의 서버에 대한 적합성 분석 및 표준제안에 SPECWeb2005를 국내실정에 맞도록 설정, 적용한다면

뛰어난 결과를 얻을 수 있을 것으로 예상된다.

▪ Banking Workload Design 로그 분석을 통한 적용

SPECWeb2005의 Banking Workload Design은 미국 텍사스지역에 있는 은행의 로그를 분석함으로써 가성 데이터를 획득하고 이 데이터를 벤치마크 평가에 도입시켰다. 또한, 분석 할 수 없는 로그 값을 가상으로 두어 실질적은 은행업무와 가장 유사한 환경을 구성하려 하였다. 하지만 이와 같은 로그 분석 데이터는 국내의 네트워크와 구성 프로그램의 차이를 고려할 때 분석된 로그 값이 국내 실정에 적합하다고 할 수 없다. 따라서 국내의 은행에서 발생하는 로그 값을 분석하여 벤치마크 평가 기준으로 도입할 필요성이 있다.

▪ 복합적 Workload Design 분석을 통한 적용

포털 사이트의 경우 가장 많은 사용자들이 이용하는 서비스를 모아서 사용할 수 있게 제공한다. 이러한 환경은 Banking, Ecommerce, Support 환경을 복합적으로 벤치마크 하여 성능을 판단할 수 있을 것이다. 하지만 보다 구체적인 평가를 위해서 분류의 기준을 보다 상세화 할 필요성이 있을 것이다. e-mail, blog, mini homepage등 개인화된 웹에 대한 평가기준을 분류하여 각 포털 서버의 성능을 평가할 필요성이 있다. 즉, 주요 서비스를 세부화 하여 분류 평가함으로써 분야별 서버평가와 종합적 서버 평가가 가능하게 할 것이며, 성능 개선을 위한 서버 문제점을 부분적/종합적 해결할 수 있는 방안으로 활용될 수 있을 것이다.

▪ 기술 발전에 따른 효과적인 적용 방안

국내 웹 서비스 환경은 이미 세계수준이라 할 수 있다. 이는 곧 최신 기술이 발 빠르게 도입되고 있음을 말한다. 이러한 최신 웹 기술의 도입은 규모가 큰 포털 사이트를 중심으로 서비스 되고 있

다. 구글과 같은 정보를 제공하는 검색 서비스를 바탕으로 콘텐츠 제공, 커뮤니티, 쇼핑을 한 번에 서비스하는 포털 사이트들이 가장 많은 이용자들을 확보하고 있다. 이러한 사이트들은 개인화된 웹 서비스를 사용자에게 제공하여 “개인화 포털”이란 개념을 도입함으로써 다양한 서비스를 제공하고 있다. 이러한 서비스는 RSS, AJAX, 태그 등의 기술을 기반으로 고급화된 개인화 포털을 제공하고 있다. 이 기술들의 특성은 리치 클라이언트를 고려한 설계이기 때문에 웹 서버에 집중적인 부하를 주지 않고 클라이언트가 부분적인 업무를 처리함으로써 보다 빠른 처리 속도를 보여주게 된다. 결국, 보다 향상된 현재의 웹 서비스 응답시간을 보여주게 되는 것이다. 이러한 기술의 변화는 SPECWeb2005의 벤치마크 방식으로 웹 서버를 벤치마크 하기에 어려움을 야기할 것이다. 서버중심의 평가에서 서버와 리치 클라이언트간의 workload를 비교 분석하여 서버만의 성능을 명확하게 평가하기 어려울 것으로 생각된다. 즉, 리치 클라이언트에서 처리하는 workload를 명확히 평가하기 위한 방안에 대한 연구가 필요하다. 현재까지 SPECWeb2005는 기존 SPECWeb99에서 벤치마크 하기 힘들었던 부분을 명확히 평가함으로써 웹 서버의 새로운 평가 도구로서 가능성을 보여주고 있으며, 형식적인 평가가 아닌 실질적인 평가를 하기 위한 다양한 평가 기술을 적용하였다. 이는 국내의 은행, 전자 상거래에서 사용하고 있는 현장 서버를 평가하고 새로운 서버 도입 시 현장에서 가장 뛰어난 성능을 적용할 수 있는 서버를 판단하는 평가 도구로써 그 적용가능성이 용이 할 것으로 판단된다.

12) SPECjAPPServer2004

본 절에서는 SPECjAPPServer2004의 사이징 기준으로의 적용방안을 제시하고자 한다. 이를 위해 우선, SPECjAPPServer2004 벤

치마크의 진행 현황을 살펴보고 이를 통해 SPECjAPPServer2004가 갖는 일반적인 문제점을 제시하고 기존 WAS 산정기준과 SPECjAPPServer2004의 벤치마크 기준과의 차이점을 분석하며, 마지막으로 WAS 기준으로의 반영 방안을 제시하고자 한다.

⊙ 벤치마크 현황 및 일반적인 문제점

SPECjAPPServer2004벤치마크 현황 및 문제점으로는 크게 표준작업부하 등과 같은 벤치마크의 구조상의 문제와 다양한 H/W 벤더 플랫폼 지원 불가, 낮은 범용성 및 적용상의 한계, 벤치마크 활용상의 문제 등 4가지를 들 수 있으며, 세부적인 내용은 다음과 같다.

■ 표준작업부하의 문제 : SPECjAPPServer2004벤치마크는 글로벌 판매 및 생산기지를 갖은 자동차업체에 대한 비즈니스모델을 기반으로 하고 있음. 따라서 공공부문 작업부하와는 많은 차이가 있어 이를 직접적으로 적용 하는데 한계가 있음. 이러한 문제점을 해결하기 위해서 작업부하의 차이를 반영하기 위한 방안이 제시될 필요가 있음.

■ 다양한 H/W 벤더 플랫폼 지원 불가 : SPECjAPPServer2004에 대한 성능측정은 2004년 4월 28일 발표된 이후 2006년 상반기까지 주로 SUN(10여건). IBM(5건), HP(4건), BEA(2건)을 중심으로 이루어져 왔으며, 이중 대부분이 SUN, HP, BEA의 경우 BEA Weblogic 서버와 SUN Java Application Server Platform 그리고 IBM의 경우 자사의 WebSphere 6.0 Application Server를 플랫폼으로 하고 있음. 벤치마크를 수행하지 않은 여타의 H/W 벤더사의 제품이나 플랫폼을 적용하는 데 한계가 있음.

■ 낮은 범용성 : 벤치마크 대상 시스템이나 플랫폼이 SPEC내의 클라이언트/서버 벤치마크 기준인 SPECjbb2005나 TPC의 PC-C 벤

치마크 기준 등 상대적으로 유사하거나 비교되는 벤치마크 기준에 비해 상대적으로 매우 낮은 수준이며, 이를 감안할 경우 벤치마크의 범용성이 낮음.

<표46> SPECjAPPServer2004의 년도 별 벤치마크 실적

구분	1사분기	2사분기	3사분기	4사분기	합계
2004	-	1건	1건	1건	3건
2005	1건	2건	4건	5건	12건
2006	4건	1건	-	-	5건
합계	-	-	-	-	20건

■ 적용상의 한계 : SPECjAPPServer2004 벤치마크의 경우 앞서 언급한 바와 같이 J2EE1.3의 사양을 근간으로 J2EE 서버들과 컨테이너들의 성능(performance)과 확장성(scalability)을 측정하기 위한 벤치마크로서 자바 환경으로 구축되는 경우는 적정할 수 있으나 자바 플랫폼이 아닌 WAS의 경우 이를 적용하는데 한계가 있음.

⊙ WAS 규모 산정식과 SPECjAPPServer2004 벤치마크와의 비교

기존의 WAS에 대한 규모 산정식은 범용적인 WAS에 대한 규모산정을 위해 만들어 졌다. 그러나 기존 SPEC이나 TPC의 벤치마크 기준은 작업부하의 특성을 세분화하여 작성되어 있으며, 작업부하 형태에 따른 서버의 성능평가가 가능하다고 볼 수 있다. 한편, 여기에서는 SPECjAPPServer2004 벤치마크와 기존의 WAS 산정식의 비교를 통해 차이점을 확인하고 이를 바탕으로 적용방안을 마련하는데 기초를 제시하고자한다. WAS 산정식은 측정단위

는 OPS로 Java OPS를 사용하고 있는 SPECjAPPServer2004 벤치마크 차이를 보이고 있음. 이는 앞서 언급한 바와 같이 WAS 산정식이 범용적인 Web Application Sever(WAS)의 규모산정을 위한 기준으로 제시됨으로써 기인한 것으로 WAS의 작업부하 형태를 적절히 반영하고 있지 못함. 규정된 응답요구 시간은 시스템의 성능에 큰 영향을 미칠 수 있음.

현재 SPECjAPPServer2004 벤치마크에서는 도메인별로 비즈니스트랜잭션의 형태에 따라 응답시간에 요구사항을 정의하고 있으나 WAS서버 산정식에서는 이를 반영할 수 있는 구체적인 보정치가 존재하지 않음.

<표47> WAS산정식과 SPECjAPPServer2004의 비교

구분	WAS 산정식	SPECjAPPServer2004	비고
측정단위	OPS(Operation Per second)	JOPS(Java Operation Per second)	
응답시간	특별한 규정 없으며, 반영 방법이 없음	모든 비즈니스에 대해서 90% 이상의 응답시간이 2초 이내 규정	
트랜잭션 비율	특별한 규정 없으며, 반영 방법이 없음	트랜잭션 유형별로 트랜잭션의 비율을 규정 세부내용은 <표3-19> 참조	

<표48> 판매자 도메인 비즈니스 트랜잭션별 응답요구시간

비즈니스 트랜잭션 형태	90% 이상 요구 응답시간
구매(Purchase)	2초
관리(Manager)	2초
검색(Browser)	2초

▪ SPECjAPPServer2004 벤치마크에서는 비즈니스 트랜잭션은 구매, 관리, 검색의 3가지 형태로 구분하고 있으나, WAS사이징 기준에서는 이를 반영하고 있지 않음.

⊙ SPECjAPPServer2004 적용 방안

SPECjAPPServer2004에 대한 산정기준으로의 적용은 벤치마크의 활용도 측면에서 고려해 볼 수 있다. 앞서 언급한 바와 같이 SPECjAPP Server2004 벤치마크는 표준작업부하 등과 같은 벤치마크의 구조, 다양한 H/W 벤더 플랫폼 지원 불가, 낮은 범용성 및 적용상의 한계, 벤치마크 활용 부족 등의 제약을 갖고 있다. 따라서 단기적(1~2년 內)으로 적용하는 것은 현실적으로 실효성이 낮을 것으로 보이며, 향후 시장에서의 활용도 등을 감안하여 적용여부를 결정하는 것이 바람직할 것이다. 특히 전략적인 측면에서 장기적으로 규모산정지침의 민간부문 정보화사업의 산정지침으로의 확산을 위해서는 SPECjAPPServer2004와 같이 민간부문의 비즈니스 모델에 적합한 벤치마크를 적극적으로 고려할 필요가 있다.

<표49> WEB/WAS 산정기준

구분	항목	내용	입력값 범위	일반값
①	동시사용자 수	소프트웨어나 시스템을 네트워크 상에서 동시에 사용하는 사용자	WEB : 5~10% WAS : 10~20%	WEB : 5% WAS : 10%
②	사용자당 오퍼레이션 수	사용자 한 사람이 초당 발생시키는 오퍼레이션 수	3~6	5
③	인터페이스 부하 보정	서버가 타 서버와 통신하게 될 때 인터페이스에서 발생하는 부하를 고려한 보정	2%~10%	5%
④	피크타임 부하 보정	갑자기 많은 접속으로 인해 부하가 발생하는 것을 해결하기 위한 보정	15%~50%	30%
⑤	시스템 여유율	시스템의 안정된 운영을 위한 보정	-	30%
산정식		CPU(OPS 단위) = 동시사용자 수 * 사용자당 오퍼레이션 수 * 인터페이스 부하 보정 * 피크타임 부하 보정 * 시스템 여유율		

한편, 적용 방안을 모색하기 위해서 현행 WEB/WAS 산정기준을 토대로 판단해 볼 때, 동시사용자, 사용자당 오퍼레이션 수, 인터페이스부하보정, 피크타임부하보정, 시스템 여유율 등 5개의 산정항목 중 사용자당 오퍼레이션수와 인터페이스 부하보정, 피크타임 부하보정 등의 산정항목에 영향을 미칠 수 있어 이에 대한 검토가 필요하다. 특히, 산정식의 핵심항목인 사용자당 오퍼레이션 수는 SPECjAPPServer2004의 오퍼레이션의 정의를 수용하여 재정립하여야 하고 사용자 당 오퍼레이션 수 역시 해당 벤치마크를 반영하도록 하여야 할 것으로 보인다. 앞서 언급한 바와 같이

SPECjAPPServer 2004에서 제시하고 있는 비즈니스별 트랜잭션 응답시간을 보장하기 위해서는 피크타임 부하보정이나 혹은 추가적인 보정항목의 설정을 통해 반영하여야 한다. 산정기준 수립 시 현행의 WEB/WAS 산정기준과 같이 서로 다른 업무특성을 대상으로 동일한 기준을 적용하기 보다는 벤치마크나 혹은 업무특성에 따라 별도의 산정기준을 수립하여 제시하는 것이 바람직할 것이다.

⊙ SPECjbb2005의 새로운 특징

데이터베이스는 벤치마크 자체적으로 코드화되어 구현되는 바이너리트리(BTree) 구조가 아닌 소팅을 요구한 테이블에 운영되는 해쉬맵(HashMaps) 과 트리맵(TreeMaps) 구조로 설계되었다. 벤치마크의 의도는 라이브러리를 사용하는 자바 개발자들의 실행을 반영하기 위한 것이다. 이는 라이브러리가 자바개발자들이 그들 자신이 코드화한 구현보다는 적당한 성능(functionality)을 제공하기 때문이다. SPECjbb2005는 벤치마크의 메인 부분에 System.GC 호출을 가지고 있지 않는다. 왜냐하면 주기적인 System.GC 호출에 의한 장기간 애플리케이션 운영에의 방해를 받지 않기 위해서이다. 현재 애플리케이션의 특징을 증대하기 위해, 재무 자료 및 연산의 처리는 부동소수점에부터 BigDecimal의 범위에서 변화가 가능하다. 이것은 현재 산업 형태에 부합되고 10진수와 반올림이 필요한 재무 자료의 수합 (amounts) 및 연산에 용이(ensure)하다. 이 코드는 진보된 객체지향적인 프로그래밍 방식을 반영하여 재구현 (Refactored)되었다. 현재 자바의 수준은 자바 5.0이다 그리고 벤치마크는 현 언어 수준의 여러 특성들을 반영하였다. 다수의 컬렉션 데이터 구조는 일반적으로 구현되었으며, 다른 소스 코드도 자동 박싱(auto-boxing) 과 조사방법(enumeration)을 사용하여 구현되었다. 트랜잭션 로깅은 이제 자바 5.0의 JAXP XML기능을 활

용한 DOM 객체로 구축(building) 하고 작성(writing)되었다.

또한 벤치마크는 JRE의 여러 인스턴스에 배치되었고, 각각은 자체 데이터 테이블에 로드된 트랜잭션을 개별적으로 조작한다.

⊙ SPECjbb2000과 SPECjbb2005의 차이점

SPECjbb2005를 만들기 위해 SPECjbb2000에 몇 가지 변화가 있었다. 이 코드는 객체지향적인 형태의 프로그램을 반영한 수많은 장소에서 재구성되었다. 현재 자바 수준은 자바 5.0이고 벤치마크 또한 언어 수준으로부터 다수의 특징을 포함하였다. 다수의 컬렉션 데이터 구조는 일반적이고, 소스 코드는 자동차폐를 사용하였다. 트랜잭션 로깅 코드 내의 에뮬레이션 형태는 자바 5.0 가능의 JAXP XML을 사용한 DOM 객체의 빌딩 및 라이팅으로 구현되었다. 벤치마크는 또한 자바실행환경(java Runtime Environment: JRE)의 다수 객체를 배치하여 데이터 테이블에 로드되는 트랜잭션을 개별적으로 처리토록 하였다. SPECjbb2005는 SPECjbb2000를 완전히 대체하였다. SPEC은 SPECjbb2005를 배포한 날짜로부터 6개월의 전환(transition) 기간을 제공했다. 이 기간 동안 SPEC은 두 벤치마크의 결과를 수용 → 리뷰 → 발표 할 것이다. 이 기간 후에는 SPECjbb2000으로 도출된 결과는 SPEC의 발표에 더 이상 유효하지 않다.

⊙ SPECjvm98과 SPECjAppServer2004와의 관계

SPECjbb2005와 SPECjvm98, SPECjAppServer2004의 관계는 다음과 같다. SPECjvm98은 사용자에게 JVM의 클라이언트 플랫폼의 하드웨어와 소프트웨어를 결합한 측면에서의 성능 평가를 제공한다. 소프트웨어 측면에서 이것은 JVM의 효율, just-in-time

(JIT) 컴파일러, 그리고 운영체제 실행을 평가한다. 하드웨어 측면에서, 이것은 CPU (정수 및 부동소수점), 캐쉬, 메모리, 그리고 다른 플랫폼 특징적인 성능을 평가한다. SPECjAppServer2004는 J2EE 1.3 애플리케이션 서브를 측정하기 위해 고안되었다. SPECjbb2000과 같이 SPECjbb2005는 전형적인 자바 비즈니스 애플리케이션이 운용되는 서버의 성능을 평가하기 위한 벤치마크이다. SPECjbb2005는 도매업자들의 주문처리 애플리케이션을 대표한다. 이 벤치마크는 JVM 서버의 하드웨어 및 소프트웨어 측면의 성능 평가에 사용된다.

⊙ SPECjbb2005 결과의 벤치마크 비교

SPECjbb2005의 결과와 SPECjbb2000의 결과를 비교할 수 없다. 두 벤치마크를 비교하기에는 너무나 많은 차이점을 가지고 있다. 또한, SPECjbb2005 결과와 SPECjAppServer2004 혹은 SPECjvm98 결과도 비교할 수 없다. SPECjbb2005 결과와 TPC-C 결과와도 비교할 수 없다. SPECjbb2005는 서로 다른 데이터 사이즈와 작업부하를 혼합하여 사용하고 있고, 운용 및 보고 규정의 상이한 셑, 상이한 처리율의 측정, 상이한 측정치를 가지고 있다. 따라서 SPECjbb2005 결과는 SPEC의 다른 벤치마크로 변환할 논리적 방법이 없어 비교할 수 없다.

⊙ SPECjbb2005 활용 범위

SPECjbb2005는 하드웨어, 운영체제, 자바실행환경 등 애플리케이션의 모든 측면을 고려한 대표적인 자바 서버 애플리케이션을 시험하기 위해 고안된 것이다. 여기서는 네트웤과 디스크 I/O는 측정되지 않는다.

SPECjbb2005를 조직이 자바 서버의 사이즈를 결정하는데 이용할 수 없다. 왜냐하면 이것은 특정 작업부하에 기반하고 있기 때문이다. 작업부하가 사용자 애플리케이션에 적용되는 적용되지 않든 작업부하를 결정하는 데는 다양한 가정이 존재한다. 또한, 데이터베이스를 실행하는 모든 운영들은 SPECjbb2005 내에서 메모리 운영이다. SPECjbb2005는 서버 환경에서 JVM 생산품을 비교하는 분야에 영향을 끼치는 수준을 제공하는 도구이다.

벤치마크의 애플리케이션 코드는 고도의 범위 평행 애플리케이션으로 기술되어야 한다. 특정 구성의 벤치마크 범위를 잘 기술하는 것은 하드웨어와 소프트웨어의 성능에 달려 있다.

⊙ 분석결과 종합

TPC-E 성능평가모델은 이전 평가모델인 TPC-C에 비해 보다 현실적인 OLTP환경에 기반을 두고 있으며, 시스템성능에 영향을 미치는 다양한 요소들을 반영하고 현실화한 것이 특징이다.

- 현실 지향적인 OLTP 애플리케이션 환경 설정
- 성능평가를 위하 부하 측정단위 변경
- 성능평가 기준 변경
- 데이터베이스 부하측정을 위한 시스템운영시간 변경
- 데이터베이스 규모 산정식 변경

TPC-C와 TPC-E의 비즈니스 환경과 애플리케이션 환경이 다르기 때문에, 두 평가모델에 사용되는 절대적인 수치를 비교분석한 결과는 큰 의미를 갖지 않지만, 전체 작업부하 정도를 측정하기 위한 다음과 같은 평가기준과 방법의 변화는 큰 의미를 갖는다.

- Primary Performance Metric: 분당 처리되는 트랜잭션의 수(tpmC)에서 초당 처리되는 유효한 트랜잭션 수(tpsE)로의 변경은 메모리 소요 공간

등의 성능평가 항목에 영향을 미친다.

- Price/Performance Metric: Primary Performance Metric의 측정단위 변경에 따라 Price/Performance Metric도 자동으로 변경된다.
- 트랜잭션 믹스 율 조정
- Keying/Think Times 항목삭제: 측정대상(순수 트랜잭션의 수) 변화
- Test Duration and Timing: 트랜잭션에 소요되는 시간, 소요 공간, 여유 율에 영향을 미친다.

측정 단위의 변화와 측정대상의 변경, 추가 및 삭제된 항목은 결국 트랜잭션의 수, 데이터베이스 소요 공간, 시스템 여유 율, 평가항목마다의 보정 값에 영향을 미치게 됨에 따라 작업부하별 적용되는 성능기준에 맞는 평가모델을 벤치 마크해야 할 필요가 있다. 뿐만 아니라 기존의 TPC-C를 기준으로 개발된 산정식을 통해서 이러한 변화된 요소들과, IT기술개발 속도와 복합적인 OLTP 비즈니스 및 애플리케이션 환경을 고려할 때 새로운 OLTP 애플리케이션 환경의 작업부하를 최적으로 평가할 수 있는지에 대한 평가가 이루어져야 한다.

평가모델분석과 절의분석결과를 토대로 종합적으로 판단하여 볼 때, 기존의 TPC-C를 기반으로 하는 산정식을 통한 성능평가는 현실적인 작업부하를 측정하기에 부적절하다는 결론이며, 따라서 TPC-E에 기반을 둔 새로운 산정식을 개발하는 것이 합리적이라 판단된다.

⊙ TPC-App

TPC-App 벤치마크는 데이터베이스 서버, 웹 서버, 여타 애플리케이션 서버들이 서로 통신하는 전형적인 작업을 얼마나 잘 수행하는지 측정하게 된다. TPC-App 벤치마크의 주요 지표는 초당 웹 서비스 페이지 당 응답 시간으로 측정되며, 애플리케이션 서버

에서 사용되는 두 가지 경쟁 기술인 MS의 닷넷과 선 마이크로시스템즈를 비롯한 연합 세력들의 자바를 서로 비교하는 기준으로 이용될 수 있도록 제안되어 있다. TPC-App 벤치마크는 기존의 TPC-W 벤치마크를 대신하게 되는데, TPC-W 벤치마크는 벤치마크를 수행하기 위해 보통 25대의 서버가 필요할 정도로 비용이 많이 들고 실제 국내의 대규모 트래픽이 발생하는 포털사이트 등에서나 볼 수 있는 캐시 서버나 이미지서버는 국내의 일반적인 웹 서비스 환경에서는 쉽게 구성하기 어려운 환경이라고 할 수 있다. 또한, TPC-W 벤치마트는 애플리케이션 서버 성능뿐 아니라 정보를 저장하는 서버의 성능은 물론 다른 보조 머신은 성능측정에 포함되지 않아서 TPC-W 벤치마크과정은 상당히 포괄적이어서 측정결과를 적용하고 반영하는데 어려움이 있었다. TPC-App 벤치마크는 이러한 TPC-W 벤치마크의 한계를 개선하고, 주요 지표인 SIPS(Web Service Interaction per Second)는 H/W 제품군에서 동일한 사양의 서버에서의 성능비교를 위해 사용할 수 있을 것으로 전망된다. 또한, 한정된 예산 내에서 적절한 H/W를 도입해야 하므로, 성능대비비용을 비교하는데 사용하여 예산이나 비용의 효과적인 집행에 도움을 줄 수 있을 것으로 예상된다. 그러나, TPC-App 벤치마크는 Web Service를 기반으로 하고 있는데, 현재 국내 공공부문 및 민간 기업에서 Web Service를 도입하고 있지만, 아직까지 Web Service가 보편화되어 있다고 보기 어렵다. 향 후 TPC-App 벤치마크 방법론을 참고하여 일반적인 서비스나 정보시스템의 성능평가 모델 개발이 가능할 것으로 전망되며, 제품 구매나 용역 사업 발주 시에 활용방안이 함께 제공된다면, 조만간 Web Service가 보편화되고 확산될 때 효과적으로 사용될 것으로 전망된다.

⊙ SPECWeb2005

SPECWeb2005는 세 가지로 분류된 평가 기준(Banking Workload Design, Ecommerce Workload Design, Support Workload Design)을 도입함으로써 이전 버전의 웹 서버 벤치마크에서 사용하는 일괄적 평가 방법보다 상세한 평가 기준을 제시함으로써 보다 명확한 웹 서버 성능을 평가할 수 있는 방법을 제시한다. 3가지의 평가기준은 다음과 같은 특성으로 요약된다.

⊙ SPECWeb2005 Banking Workload Design

Banking Workload Design은 SPECWeb2005 Workload Design 중 가장 복잡한 구조로 설계된 것으로 은행 업무에 대한 상황을 근거로 웹서버 로그에 대한 연구를 기반으로 한다. SPEC은 Banking에 관련된 철저한 분석을 위해 텍사스 지역에 근거한 은행에서 웹서버 로그에 대한 실질적인 데이터를 이용하여 분석하고, 은행업무의 특성상 기밀성이 요구되는 데이터들을 분석하기 위해 임시의 거래를 가상으로 실시하여 은행업무 웹 사이트에 처리를 요구함으로써 자료 정보를 분석하여 설계되었다.

⊙ SPECWeb2005 Ecommerce Workload Design

일반적인 Ecommerce Web stores에서 요구되는 점을 조사해보면 Ecommerce transaction에 있어서 명확히 세 가지 파트로 구분된다.

① 브라우즈

브라우즈는 사용자가 선택한 타입에 맞추어 적절한 섹션으로 이동하여 관심 물품을 선택 할 수 있도록 제공한다.

② 상품의 조합

상품의 조합은 유저가 제안 또는 요청하는 부분이다.

③ Ecommerce

Ecommerce 는 실질적인 부분으로 구매자가 그들이 원하는 조합을 선택하여 쇼핑카트에 올리고, 카트페이지는 유저가 선택제품을 추가하고, 교체하고, 삭제하고, 카트내용을 세이브 하는 것이 가능해야하며, 유저가 “Checkout"을 클릭했을 때 세션은 보안이 이루어지는 부분이다.

⊙ SPECWeb2005 Support Workload Design

SPECWeb2005 Support Workload Design은 Vendor's Support Web Site를 가정하여 설계되었다. 이용자는 특정제품 (혹은 파일) 등을 검색하고, 리스트를 보고, 각 항목에 따른 필터링이 가능하고, 또한 파일들을 다운로드 받을 수 있다. Support Workload는 대형공급사의 실제 회사들의 평균적인 페이지 사이즈, 이미지사이즈, 접속빈도 (브라우저의 캐싱 때문에 재접속 되는 것 또한 감안하여) 등을 로그파일의 수집을 통해 분석함으로써 더 향상되었다. Support Workload 에서는 파일 다운로드를 위한 실제의 로그파일들에서 보여 지는 액세스 패턴들이 모델이 되었다. Support workload 클라이언트로부터 다이내믹 페이지가 요구되는 반면, 다른 두개의 Workload(뱅킹과 전자상거래)에 비하여 페이지가 더욱 심플해졌다. 예를 들어, Across Page의 Request(세션쿠키 안에 저장되어야 됨)가 계속 유지되어야하는 사용자 데이터가 없다. 그 대신, 이 Support Workload에서는 대용량의 파일 다운로드가 강조되는 특성이 있다.

이와 같은 3가지로 분류된 평가기준을 가진 SPECWeb2005는 기존 99와 99_SSL과 완전히 다른 방식을 설계되었으며 평가 방법

또한 기존 버전들과 완전히 다르다고 볼 수 있다. 즉, 새로운 개념으로 설계 및 개발 되었다. 또한 자바코드를 이용하여 완전히 새롭게 작성되었기 때문에 운영체제에 제한 받지 않고 설치함으로써 기존 시스템에 영향을 받지 않는 뛰어난 이식성이 지니고 있다. 이러한 SPECWeb2005는 웹 서비스를 제공하는 서버 측정에 있어 서비스 분류를 통한 벤치마크 도구로 활용이 매우 용이하다고 할 수 있다.

13) OLTP용 서버 적용 시 고려사항

TPC-E는 현실세계의 OLTP환경을 재구성한 시스템을 통해서 트랜잭션이 수행되는 동안 시스템에 영향을 미치는 다양한 요인들을 고려하여 데이터베이스에 어느 정도의 작업부하가 걸리는지를 측정하려는 성능평가모델이다. 즉, 트랜잭션 처리 및 데이터베이스 성능 벤치마크를 정의하고, 정의된 벤치마크 기준에 따라서 객관적인 성능데이터를 생성하기 위한 평가모델이다. TPC-E 벤치마크모델에서는 트랜잭션처리를 비롯한 데이터베이스의 속도 및 비용이 주요 평가대상이며, 벤치마크 평가 결과는 IT 전문 업체에서 실제적인 OLTP 시스템 성능을 나타내는 유효한 지표로서의 역할을 한다.

현실 세계의 비즈니스 및 애플리케이션 환경(OLTP환경)은 대상 분야 및 상황에 따라 다르기 때문에, 획일적으로 재구성하여 일률적인 기준을 제시할 수는 없으나, 최종적으로 제시되어야할 성능평가결과(속도와 비용)에 영향을 미치는 다음과 같은 다양한 요인들을 고려할 필요가 있을 것이다.

⊙ 서버의 용량 및 성능

비즈니스 및 애플리케이션 환경의 변화로 실제 OLTP환경에서 처리되는 트랜잭션수가 엄청나게 증가하고 있는 사실을 반영하여 TPC-E에서는 성능평가 측정단위를 tpmC에서 tpsE로 변경하였다. 이는 처리해야할 트랜잭션 수의 증가를 의미하기 때문에, 트랜잭션처리수를 측정하는 시간을 단축시킴은 당연한 결과라 할 것이다. 그러나 엄청나게 증가하고 있는 트랜잭션을 처리할 수 있는 용량과 성능을 구비한 서버의 선택은 가장 먼저 고려해야할 사항이다.

⊙ 데이터베이스

TPC-E 벤치마크에서 온라인 트랜잭션 작업부하를 처리할 수 있느냐는 결국 탑재된 데이터베이스에 달려 있기 때문에, 저렴한 비용과 빠른 속도를 고려하여 모든 애플리케이션 종류 및 규모에 적합한 데이터베이스를 선택하는 문제를 고려해야 한다.

⊙ 애플리케이션 규모 및 복잡도

애플리케이션 환경은 상황에 따라 다르기 때문에, 절대적인 기준은 제시할 수 없으나, OLTP환경 자체를 구체적으로 파악하여 어느 정도로 현실적으로 재구성하느냐에 따라서 성능평가결과가 달라질 것이기 때문에, 애플리케이션 자체를 이해하는 것 또한 주의사항에 포함될 것이다.

⊙ 기타 고려사항

TPC-E 벤치마크 모델 성능평가결과에 영향을 미치는 기타 사항들을 정리하면 다음과 같다.

- 운영체제: 데이터베이스 서버가 가장 효율적으로 운영될 수 있는 운영

체제

- CPU
- 메모리
- 시스템 및 데이터 디스크
- 네트워크 구성 및 체계
- 사용자 환경
- 비즈니스 환경

⊙ TPC-E 벤치마크 모델 활용 방안

TPC-E 표준, TPC's Use Policies, Guideline 등에서 사용되는 TPC-E 성능평가 기준(Primary Metrics)은 속도(tpsE)와 비용 대비 속도(price/tpsE)로 정의된다. 최종 평가 결과는 엄격한 요구사항에 따라서 개별적인 내부 감사를 거치고 인증과정을 거쳐 보고서 형식으로 TPC에 보고된다. 결과보고서에 수록된 데이터는 다음과 같이 시스템성능을 평가하기 위한 업계 표준지표로 활용되는 것이 가장 기본적인 활용방안이며, 이 지표를 활용하는 주체에 따라서 다양하게 활용 가능하다. 예를 들어, OLTP환경의 하드웨어 혹은 소프트웨어를 구축하기 위한 기초 자료로 활용될 수 있으며, 시스템의 무 결성을 테스트하기 위한 자료로 활용할 수도 있을 것이다.

⊙ 시스템 성능을 평가하기 위한 업계 표준 데이터로서의 역할

TPC-E 성능평가 결과 Performance Metric (tpsE), Price/Performance Metric (price/tpsE)의 가장 기본적인 적용방안은 OLTP시스템 성능을 평가하기 위한 업계 표준 데이터로 활용 가능하다는 것이다. 이는 트랜잭션 처리 및 데이터베이스 성능 벤치마크를 정의하고 이러한 벤치마크를 기준으로 객관적인 성능

데이터를 공시하기 위한 TPC(Transaction Processing Performance Council)의 근본 사업목적과 부합된다.

⊙ OLTP시스템(H/W, S/W) 개발을 위한 기초 데이터로 활용

TPC-E 벤치마크에 참여하는 업체들은 OLTP환경에서의 시스템 성능(속도) 및 가격 대비 성능(비요)에 대한 객관적인 정보를 얻을 수 있으며, 업체들은 이러한 정보를 활용하여 비용 대비 고객에게 더 큰 만족을 줄 수 있는 견고하고 확장 가능한 시스템을 개발하기 위한 기초 자료로 활용할 수가 있다.

⊙ 시스템 신뢰성 확보를 위한 무결성 테스트를 위한 데이터로 활용

TPC-E 벤치마크에서는 트랜잭션을 비롯한 다양한 데이터베이스 기능의 테스트까지를 포함하고 있기 때문에, 제공되는 자료는 시스템의 신뢰성을 확보하기 위한 목적으로 수행되는 무 결성 테스트 자료로서도 활용할 수가 있다.

⊙ 시스템 확장을 위한 기초 데이터로 활용

OLTP 환경에서 애플리케이션 및 비즈니스 환경은 시간이 지남에 따라서 수시로 변하게 되는데, TPC-E 벤치마크에서 평가되는 모든 성능평가는 1초당 처리되는 트랜잭션수를 기반으로 하기 때문에, 이를 기준으로 시스템의 확장여부를 판단하는 기초 자료로서 활용 가능하다.

⊙ 시스템 하드웨어 규모를 산정하는 기준 데이터로 활용

TPC-E 성능평가 결과에는 TPC-E Throughput,

Price/Performance 이외에 Operating System, Database Server Configuration, Database Manager, Processor, Memory, System Hardware Component, initial DB Size, Server Storage, Server Hardware, Server Software, Client Hardware, Client Software, Infrastructure, Response Time, Transaction Mix 등 데이터들이 포함되어 있어 이들 데이터를 이용하여 시스템 하드웨어 및 소프트웨어 구성을 위한 기초 자료로 활용할 수가 있다.

⊙ TPC-E 벤치마크 모델 발전 방안

Transaction Processing Performance Council (TPC)는 정보기술의 발달과 실제 OLTP환경의 애플리케이션 및 비즈니스의 변화를 고려하여 TPC-C의 후속모델인 TPC-E 벤치마크 모델을 개발하고 있으나, 아직 완전한 상태의 벤치마크 모델을 선보이지 못하고 있다. 본 보고서는 2005년 8월 17일 발표된 TPC-E Draft Revision 0.30과 2006년 5월 5일 발표된 Draft Revision 0.32.2d를 기준으로 하였으나, 2006년 7월 17일 Draft Revision 0.32.2g가 발표된 상황에 있다.

TPC-E 벤치마크 모델이 아직 최종 모델로 완성된 상태가 아니기 때문에, TPC-E 벤치마크 모델의 발전방안을 논의하는 것은 시기상조일 것이다. 그러나 현재의 TPC-E V0.32.2G를 기준으로 판단할 때, TPC-E 벤치마크 모델개발자들이 최종 모델을 완성해가는 과정에서 고려하거나, 포함시켜야할 내용들을 정리하면 다음과 같다.

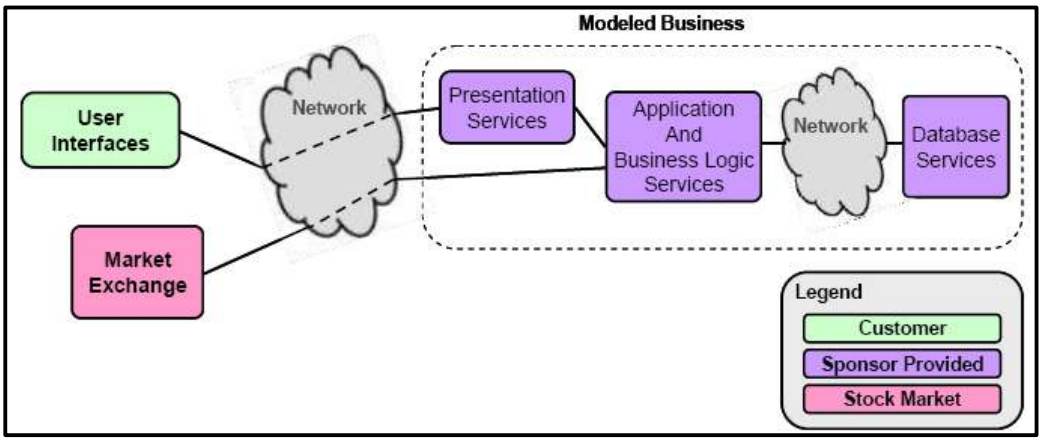
⊙ OLTP 환경의 다양성 반영

TPC-C 벤치마크 모델은 대규모 유통업체의 주문처리 프로세스

처리환경을 내용으로 하고 있으며, TPC-E 벤치마크 모델의 경우에는 주식 및 증권거래 중개회사의 트랜잭션 처리환경을 내용으로 하고 있다. TPC-C와 TPC-E의 비즈니스 및 애플리케이션 환경이 상이하며, 각각의 다른 환경의 트랜잭션을 처리하기 위한 테이블 및 데이터구조로 OLTP환경을 시스템으로 재구성하여 성능을 평가하고 있다. 근본적으로 테이블 및 데이터구조, 트랜잭션처리 구조가 상이한 다른 OLTP환경의 평가결과를 다른 환경에 일괄적으로 적용하기에는 한계가 있다. 같은 OLTP환경이라 하더라도 제조업, 금융/서비스업, 인터넷비즈니스 등의 분야는 트랜잭션처리구조가 다름은 물론, 사용자환경, 트랜잭션 발생 환경 등 성능에 영향을 미치는 요인들이 달리 작용하기 때문에, 다양한 OLTP환경을 반영할 수 있도록 하거나, 혹은 트랜잭션 구조나 분야별로 차별적인 Reference Benchmark Model를 개발할 필요가 있을 것으로 판단된다.

⊙ 네트워크 환경변화 고려

TPC-C 벤치마크 모델은 물론, TPC-E 벤치마크 모델의 경우에도 기본 환경은 아래 그림에서 보는 바와 같이 클라이언트가 서버에 접속하여 원하는 트랜잭션을 요구하고, 서버에서 이를 처리하여 그 결과를 클라이언트에게 보내는 형식으로 되어 있다.



<그림46> TPC-E 벤치마크 모델의 기본 환경

즉, 현재의 OLTP환경의 기본 Node는 Client/Server/Database Server이다. 이들 Node의 연결은 네트워크를 통해서 이루어지는데, OLTP환경이 변하는 것처럼, 클라이언트와 서버, 데이터베이스 서버를 연결하는 네트워크 기술 환경도 급속하게 변하고 있는 실정이다. 물론 Client/Server의 근본적인 구조가 변하지는 않는다 하더라도 네트워크 기술의 변화에 따라 서버에서의 처리결과가 클라이언트에게 응답되기까지 소요되는 시간에는 결정적인 영향을 미칠 것이 자명하다. 따라서 클라이언트와 서버 그리고 데이터베이스 서버가 연결되는 네트워크 기술 환경변화와 상관없이 객관적인 성능을 평가할 수 있는 새로운 평가기준을 마련하기 위한 노력도 필요할 것으로 보인다.

⊙ SPECWeb2005 적용 시 고려사항

SPECWeb2005를 이용한 웹 서버 벤치마크를 실행할 때 기본적으로 고려해야 하는 사항으로 SPECWeb2005의 평가 기준, 네트워크 구성, 평가 도구의 특성에 대한 분석이 필요하다. 자세한 내용

은 다음과 같다.

⊙ Session 기반 성능평가

SPECWeb2005는 동시 사용자를 세션기반에서 페이지 접근을 감시함으로써 성능 평가를 측정한다. 이는 접속기반에서 페이지 기반으로 평가 기준을 변경함으로써 기존의 쿠키 기반의 사용자 접근과는 다른 성능평가 결과로써 보다 명확한 서버 성능 평가가 가능하다.

⊙ 네트워크 구성

SPECWeb2005 벤치마크 테스트를 하기 위해서는 반드시 서버와 Client 사이에 하나 이상의 네트워크를 통한 연결이 이루어져야 한다. 벤치마크 코드는 각각의 측정자에게 분산되어 배포되고 Fileset은 서버에 의해 생성된다.

⊙ Back-End Simulator(BeSim)

BeSim은 웹 서버가 HTTP 응답(예를 들면 고객 자료)을 완료하기 위해 필요로 하는 특정의 정보를 회수하기 위해 의사소통해야만 하는 back-end 과정에 필요한 적용 서버를 에뮬레이터 하는데 사용된다. 즉, BeSim 타입의 웹 서버와 back-end 과정에 필요한 서버 사이의 통신을 에뮬레이터 하기 위해 존재한다. 이 프로그램은 데이터베이스 서버가 별도로 구성되었을 경우 웹 서버와 데이터베이스 서버 간 통신을 감시하여 실질적인 웹 서버의 성능만을 평가하기 위한 도구로 사용된다.

⊙ 장치 소프트웨어 디자인

SPECweb2005는 Prime Clienet 코드에 포함된 일반 벤치마크

장치와 클라이언트 작업부하 코드에 포함된 workload-specific 기능성으로부터 나온 클라이언트 베이스 코드를 분리한다. 이 방법으로 구성하면 작업부하는 좀 더 쉽게 증가하고, 제거되며 기본 장치 코드에서 변화 없이 교체가 된다. SPECweb2005를 위한 Workload-Specific 클라이언트 코드는 세 개의 클래스 파일들로 구성되어있다. (SPECweb_Banking, SPECweb_Ecommerce, and SPECweb_Support).

⊙ SPECWeb2005 구성

SPECWeb2005는 다음과 같은 내용으로 구성되어 있다.

- 자바 클라이언트 장치를 위한 소스코드와 클래스
- file set 생성 프로그램 소스 코드와 클래스
- backend 시뮬레이터의 소스코드와 클래스
- 동적 콘텐츠 실시를 위한 소스코드
- 문서

⊙ SPECWeb2005 모델 적용 방안

SPECWeb2005는 앞 절에서 언급 했듯이 Banking Workload Design 로그 분석과 Workload Design 분석을 통한 포털 서비스 서버 적용, Web2.0 기술도입에 따른 리치 클라이언트 성능고려 등을 통해 벤치마크를 효과적으로 적용할 수 있을 것이다. 이러한 기술들은 업계표준의 벤치마크 도구로서 활용이 가능하며, 서비스별로 사용자에게 성능 좋은 서비스와 전송 능력을 보유한 서비스 기관임을 평가하는 도구로 활용이 가능하다.

⊙ 신뢰성 분석 지표로서의 활용

SPECWeb2005를 이용한 벤치마크를 통해 웹 서비스를 제공하는 기관의 신뢰성을 증명하는 객관적인 분석 데이터 평가 자료를 제공함으로써 서비스를 안정적으로 제공할 수 있는 기술 보유의 근거 자료로 제시할 수 있을 것이다. 은행, 전자상거래의 경우 서비스에 대한 신뢰성은 매우 중요한 자료로 활용되므로 서비스에 대한 평가 기준을 제시하고 성능을 증명할 수 있는 도구로 매우 유용할 것이다.

⊙ 하드웨어 성능 평가 도구로서의 활용

SPECWeb2005를 이용하여 웹 서비스를 제공하는 웹 서버의 성능을 평가하여 분야별 최고의 성능을 제공하는 서버를 평가 할 수 있는 도구로 활용이 가능하다. 이와 같은 도입은 기업 또는 공공기관에서 최적의 자원으로 웹 서버 효율성이 가장 좋은 서버를 도입함에 있어 객관적인 지표로 사용될 수 있을 것이다. SPECWeb2005를 통한 감사로 성능평가를 확실하게 인정받기 위해서는 SPECjapp Server2004등과 같은 벤치마크 도구를 함께 활용하여 성능 치 계산 값을 다양하게 확보하여 업계 표준 벤치마크를 통한 성능을 확실히 인정받는 것이 중요하다. 또한, 하드웨어의 특성상 성능평가와 동시에 실질적인 도입 시 중요한 전력 이용, 도입 기기의 가격 효율성 등을 추가적으로 평가 하여 데이터화함으로써 자료를 활용 할 수 있을 것이다.

⊙ 서비스 동시접속 능력 평가

SPECWeb2005는 사용자를 세션 별로 동시 접속하여 서비스를 처리 하는 것이 평가의 가장 큰 목적이며, 이러한 평가에 있어 서비스별로 분류하여 서버의 성능을 평가할 수 있는 장점이 있다. 업계에서 서버를 도입함에 있어서 서비스의 성능별 평가를 도입하

여 측정 결과를 얻음으로써 적합한 서버를 선택할 수 있는 지표로 활용할 수 있을 것이며, 복합적인 서비스를 제공 할 경우 예상되는 종합 성능 평가 자료를 기존 서비스별 동시접속 데이터를 분석함으로써 제공 할 수 있을 것이다.

⊙ SPECWeb2005 모델 평가

SPCEWeb2005는 다음과 같은 특징으로 웹 서버 성능평가 모델로서 적절하다고 평가할 수 있다.

- Banking, Ecommerce, Support 로 구분되는 Workload Design 평가 방법.
- 기존 버전에서 평가하지 못했던 작업 파일의 Size Range 데이터 구분 및 자료 수집.
- Markov Chain 을 통한 사용자의 요청 데이터 추측을 위한 확률 적용.
- 구성파일 분석을 통한 벤치마크 테스트 환경의 이해.

⊙ Workload Design 평가 방법

SPECWeb2005 Workload Design의 구분은 앞에서도 언급한 바와 같이 특정 서비스 영역에 있어 매우 효과적인 평가 기준으로 적용될 것이다. 이러한 평가기준 적용은 응용을 통해 다양한 서비스 분야에 적용이 가능하다. 예를 들어 쇼핑 서비스를 제공하는 대부분의 웹 서버의 경우 Ecommerce Workload Design을 적용하여 평가를 할 수 있을 것이며, 국내 대다수의 은행의 경우 Banking Design 을 적용하여 서버의 성능을 테스트 및 평가 할 수 있을 것이다. 또한 하드웨어 공급 업체에서 새로운 서버 제품을 출시할 때 성능을 평가하는 자료로 각 Workload Design 별 분석 결과 값을 활용하여 서버의 우수성을 홍보하는 자료로 활용이 가능하다.

⊙ Size Range 데이터 구분 및 자료 수집

SPECWeb2005는 기존 SPECWeb 버전에서 평가하지 못했던 작업 요청 파일의 size 구분 범위를 확대함으로써 아주 적은양의 패킷의 이동까지도 분석할 수 있게 되었으며 기존 버전들에서 무시되던 데이터들을 모두 분석함으로써 보다 구체적인 평가가 가능하다. 국내의 경우 네트워크 기반 환경이 매우 좋기 때문에 대용량의 파일 전송에 대한 서버 벤치마크가 필수적이다. 이미, 포털사이트에서 제공되는 메일은 Gigabyte 단위의 대량의 파일 전송을 지원하며, 파일 공유 사이트는 1인당 하루 100GB이상의 파일을 송. 수신하고 있다. 이러한 고용량의 데이터 전송에 따른 서버 성능평가를 위한 도구로서 SPECWeb2005의 Support Design는 적절한 서버 벤치마크 도구로서 활용이 용이하다.

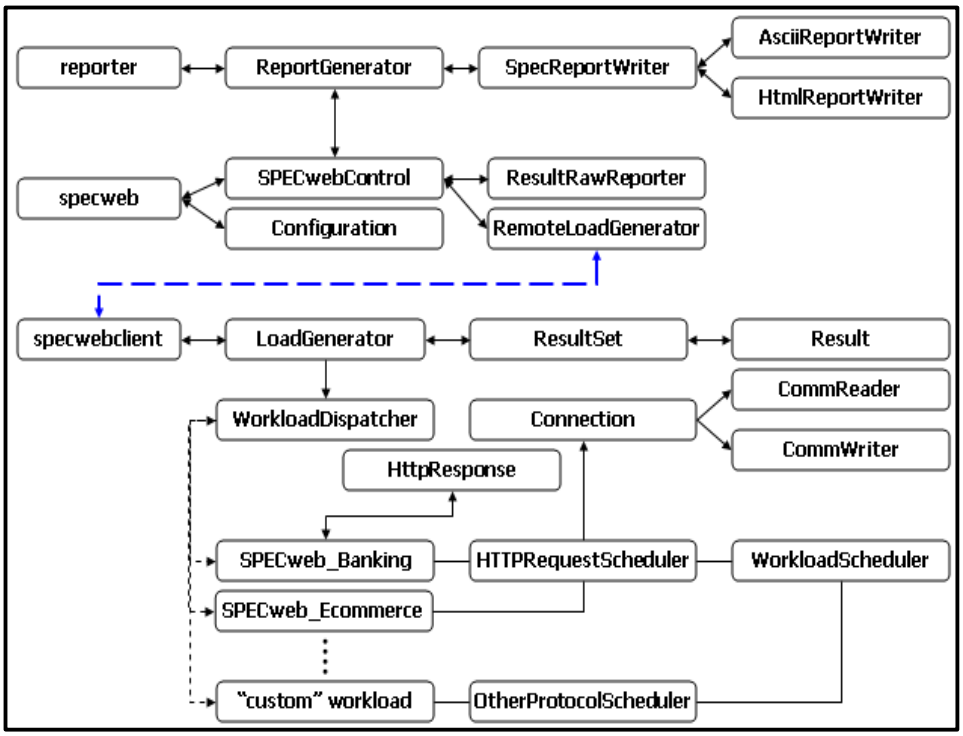
⊙ Markov Chain 을 통한 사용자의 요청 데이터 추측을 위한 확률 적용

Markov Chain을 통한 분석 자료 적용은 사이트 이용에 따른 사용자 이동 확률 값을 제공함으로써 사용자가 요청하게 될 데이터들을 페이지 별로 상세히 분석 할 수 있게 지원한다. 이와 같은 분석 자료는 많은 양의 페이지로 구성되어 있는 사이트에서 사용자가 이동하는 경로를 가상으로 적용하여 실제 서비스 이용 시 서버에 요청하는 상황과 유사한 환경을 조성하여 평가에 도움을 준다.

⊙ 구성파일 분석을 통한 벤치마크 테스트 환경의 이해

SPECWeb2005의 구성파일 분석을 통한 벤치마크 테스트 환경

을 이해하여 적용할 수 있다. SPECWeb2005에서 제공하는 기본적인 자바 클래스는 아래 <그림47>와 같다.



<그림47> SPECweb2005 Key Java Classes

이와 같은 실행 구조의 분석은 벤치마크 실행의 이해를 돕고 추가적인 모듈 설계와 확장 가능성을 제공한다. 특히, SPECWeb2005 academe 버전의 경우 소스의 수정을 통해 새로운 벤치마크 실험을 지원 하고 있다.

⊙ SPECWeb2005 모델의 발전 방안

SPECWeb2005는 전반적으로 웹 서비스의 환경 및 기술의 발달

에 따라 성능 평가를 객관적으로 실행하기 필요조건을 갖추고 있다. 특히, 은행 및 전자상거래에 적용되는 서버의 성능 테스트를 분류하여 실행하는 것은 매우 뛰어난 평가 기준이라고 할 수 있다. 하지만 SPECWeb2005 평가를 진행하는 웹 환경은 매우 빠르게 변화 하고 있다. 따라서 SPECWeb2005의 웹서버 성능 평가 도입 시 국내 실정에 적합한 형태, 즉 은행과 전자상거래 및 대용량 파일을 제공하는 웹 서버 평가 기준에서 사용자의 부하를 더욱 많이 필요로 하는 게임 서버, 포털 사이트 서버 등에 대한 대안으로 응용 벤치마크 기술로의 발달 및 도입이 필요하다. 웹 서비스 환경은 기존의 서버/클라이언트 구조 모델로서 크게 변화하지 않았지만 대부분의 서비스 처리를 전담하는 서버의 작업량이 급격히 증가함을 방지하기 위한 rich client 개념의 도입 및 적용으로 웹 서비스의 제공 및 처리 환경의 변화가 진행 중이다. 이는 서버의 성능 평가뿐만 아니라 클라이언트가 자체적으로 작업을 처리하는 시간, 서버로 전송 및 제공 받는 시간의 과정을 분류하여 벤치마크 함으로써 rich client환경에 따른 서버만의 성능을 평가하기 위한 연구가 필요하다. 이와 같은 웹 서비스 기술은 최근 주목 받고 있는 AJAX(Asynchronous JavaScript And XML), 태그, RSS(Rich Site Summary) 등의 기술을 이용하는 사이트들에 대한 작업 요청 및 처리요구를 집중적으로 벤치마크 할 수 있는 또 하나의 Workload Design이 필요할 것으로 예상되며, 서버에서 처리하는 작업량과 더불어 클라이언트에서 처리되는 작업량을 이중으로 평가함으로써 보다 복잡한 성능 평가 도구를 요구하게 될 것이다. 물론 SPECWeb2005를 바탕으로 한 응용 기술 개발이 좋은 대안이 될 것이다.

14) WAS 서버

⊙ TPC - App

TPC(Transaction Performance Processing Council)에서 TPC-W를 대체하기 위하여 새롭게 제안한 TPC-App는 Application Server와 Web Service의 성능 측정을 위한 벤치마크 표준이다. TPC-App 벤치마크는 데이터베이스 서버, 웹 서버, 여타 애플리케이션 서버들과 데이터를 교환하는 전형적인 전자상거래 작업을 서버가 중간에서 얼마나 잘 수행하는가를 측정하게 되고, 점수는 1초당 처리한 웹 서비스의 수로 처리가 된다.

⊙ 모델 적용 방안

TPC-W 벤치마크나 TPC-App 벤치마크 모두 H/W의 성능과 가격대비 성능 등을 비교 할 수 있도록 도와준다. 또한 TPC-App 벤치마크에서 제안하는 SUT 환경은 기존의 타 벤치마크에 비해 매우 현실적으로 제안되고 있다. TPC-App 벤치마크의 가장 중요한 지표인 SIPS(Web Service Interaction per Second)는 H/W 제품군에서 동일한 사양의 서버에서의 성능비교를 위해 사용할 수 있을 것으로 전망된다. 또한, 한정된 예산 내에서 적절한 H/W를 도입해야 하므로, 성능대비비용을 비교하는데 사용하여 예산이나 비용의 효과적인 집행에 도움을 줄 수 있을 것으로 예상된다. 그러나 실제로 현업에서는 H/W간의 성능평가도 중요하지만, 무엇보다 업무나 서비스에 근거하여 필요한 기능 요구사항들이 구체화되고 이를 근거로 하여 이러한 기능 요구사항들이 원활하게 서비스될 수 있도록 필요한 H/W의 용량산정이 필수적이지만, 현실적으로 필요한 용량산정이 정량적으로 접근하기 보다는 업무 담당자들의 경험에 근거하여 용량을 산정하게 된다. 기존에 몇몇 시스템 통합업체 등에서 제안되던 TpmC를 추정하는 방식 역시 사용자의

추정치에 근거를 하거나 Web 서버나 WAS 서버의 용량산정의 유일한 지표는 전체사용자 및 동시사용자를 근거로 하여 용량을 산정하기에 구축 완료 이후 터무니없이 큰 용량의 서버로 예산이 낭비되거나, 용량 부족으로 서비스에 문제가 발생하는 경우가 자주 발생하였다. 또한, TPC-App 벤치마크는 Web Service를 기반으로 하고 있다. 현재 국내 공공부문 및 민간 기업에서도 Web Service를 도입하고 있으며, 이미 많은 서비스와 시스템이 Web Service로 구축되고 있지만, 아직까지 Web Service가 보편화되어 있다고 보기 어렵다. 향 후 TPC-App 벤치마크 방법론을 참고하여 일반적인 서비스나 정보시스템의 성능평가 모델 개발이 가능할 것으로 전망되며, 제품 구매나 용역 사업 발주 시에 활용방안이 함께 제공된다면, 조만간 Web Service가 보편화되고 확산될 때 효과적으로 사용될 것으로 전망된다.

⊙ SPECjAPPServer2004

SPECjAPPServer2004은 SPEC의 Java/Client를 위한 벤치마크 기준으로 SPECjAppServer2001과 그 뒤를 이어서 발표된 SPECjAppServer2002의 후속 모델이다. SPECjAPPServer2004이외에도 현존하는 Java/Client 벤치마크 기준으로는 SPECjbb2005이 있다. SPECjAppServer2004는 J2EE1.3의 사양을 근간으로 J2EE 서버들과 컨테이너들의 성능(performance)과 확장성(scalability)을 측정하기 위한 것이며, SPECjbb2005는 서버 측면의 자바 성능을 평가하기 위한 벤치마크 기준이다.

⊙ 모델 적용 방안

SPECjAPPServer2004에 대한 산정기준으로의 적용은 벤치마크

의 활용도 측면에서 고려해 볼 수 있다. 앞서 언급한 바와 같이 SPECjAPP Server2004 벤치마크는 표준작업부하 등과 같은 벤치마크의 구조, 다양한 H/W 벤더 플랫폼 지원 불가, 낮은 범용성 및 적용상의 한계, 벤치마크 활용 부족 등의 제약을 갖고 있다. 따라서 단기적(1~2년 內)으로 적용하는 것은 현실적으로 실효성이 낮을 것으로 보이며, 향후 시장에서의 활용도 등을 감안하여 적용여부를 결정하는 것이 바람직할 것이다. 특히 전략적인 측면에서 장기적으로 규모산정지침의 민간부문 정보화사업의 산정지침으로의 확산을 위해서는 SPECjAPPServer2004와 같이 민간부문의 비즈니스 모델에 적합한 벤치마크를 적극적으로 고려할 필요가 있다. 한편, 적용 방안을 모색하기 위해서 현행 WEB/WAS 산정기준을 토대로 판단해 볼 때, 동시사용자, 사용자당 오퍼레이션 수, 인터페이스부하보정, 피크타임부하보정, 시스템 여유율 등 5개의 산정항목 중 사용자당 오퍼레이션수와 인터페이스 부하보정, 피크타임 부하보정 등의 산정항목에 영향을 미칠 수 있어 이에 대한 검토가 필요하다. 산정식의 핵심항목인 사용자당 오퍼레이션 수는 SPECjAPPServer2004의 오퍼레이션의 정의를 수용하여 재정립하여야 하고 사용자 당 오퍼레이션 수 역시 해당 벤치마크를 반영하도록 하여야 할 것으로 보인다. 앞서 언급한 바와 같이 SPECjAPPServer 2004에서 제시하고 있는 비즈니스별 트랜잭션 응답시간을 보장하기 위해서는 피크타임 부하보정이나 혹은 추가적인 보정항목의 설정을 통해 반영하여야 한다. 산정기준 수립 시 현행의 WEB/WAS 산정기준과 같이 서로 다른 업무특성을 대상으로 동일한 기준을 적용하기 보다는 벤치마크나 혹은 업무특성에 따라 별도의 산정기준을 수립하여 제시하는 것이 바람직할 것이다.

⊙ SPECjbb2005

SPECjbb2005 (자바 서버 벤치마크)는 서버 측면의 자바 성능을 평가하기 위한 SPEC의 벤치마크이다. 이전(以前) 벤치마크인 SPECjbb2000과 같이 SPECjbb2005는 중간층(middle tier)을 중요시하는 3-tier 클라이언트/서버 시스템을 에뮬레이션하여 서버 측면의 자바 성능을 평가한다. 이 벤치마크는 JVM (Java Virtual Machine: 자바가상머신), JIT(Just-In-Time: 즉시실행) 컴파일러, 폐영역 회수(garbage collection), 스레드(threads)의 실행 및 운영체제의 일부분을 시험한다. 이것은 또한 CPU, 캐쉬, 기억 계층(memory hierarchy)의 성능 및 공유메모리 프로세서(shared memory processors: SMPs)의 범위성(scalability)을 측정한다.

⊙ 모델 적용 방안

SPECjbb2005는 실세계의 애플리케이션을 설계하는데 좀 더 객체 지향적인 방법으로 구현된 새로운 고도화 작업부하(enhanced workload)를 제공하고, 현재의 애플리케이션을 좀 더 실질적으로 반영하는 벤치마크에 XML 프로세싱과 BigDecimal 연산처리와 같은 새로운 특징들이 있다.

SPECjbb2005는 하드웨어, 운영체제, 자바실행환경 등 애플리케이션의 모든 측면을 고려한 대표적인 자바 서버 애플리케이션을 시험하기 위해 고안된 것이다. 여기서는 네트워크과 디스크 I/O는 측정되지 않는다. SPECjbb2005를 조직이 자바 서버의 사이즈를 결정하는데 이용할 수 없다. 왜냐하면 이것은 특정 작업부하에 기반하고 있기 때문이다. 작업부하가 사용자 애플리케이션에 적용되는 적용되지 않든 작업부하를 결정하는 데는 다양한 가정이 존재한다. 또한, 데이터베이스를 실행하는 모든 운영들은 SPECjbb2005

내에서 메모리 운영이다. SPECjbb2005는 서버 환경에서 JVM 생산품을 비교하는 분야에 영향을 끼치는 수준을 제공하는 도구이다.

벤치마크의 애플리케이션 코드는 고도의 범위 평행 애플리케이션으로 기술되어야 한다. 특정 구성의 벤치마크 범위를 잘 기술하는 것은 하드웨어와 소프트웨어의 성능에 달려 있다.

15) 결론

정부에서는 하드웨어 도입자원의 효율적인 산정을 위한 필요성을 인식하고 정보통신부와 한국전산원을 중심으로 “H/W 용량산정 기법연구”(2002, 한국전산원), “정보시스템 용량산정 기술 및 프레임워크연구”(2003, 한국전산원), “정보시스템 규모별 용량산정 연구”(2004, 한국전산원) 등의 연구들을 수행한 결과, 정보시스템 하드웨어 규모산정 지침(2005, 한국전산원), 정보시스템 하드웨어 규모산정 가이드라인(2006, 한국전산원)에 관한 연구를 현재 진행하였다. 이들 H/W 규모산정 연구는 국제적인 공인인증기관인 TPC와 SPEC의 벤치마크를 토대로 하고 있다. 그러나 2005년 이후 이들 기관의 벤치마크 기준은 정보기술의 발전에 따라 다양하게 변화하고 있다. 따라서 본 연구에서는 기존 H/W 규모산정에 근간이 되고 있는 TPC와 SPEC의 TPC-E, TPC-APP, SPECjbb2005, SPECjappserver2004, SPECWeb 2005 등 신규기준의 적용을 위한 사전연구로서 다음의 연구를 수행하였다.

⊙ 정보시스템 성능평가 모델 관련 국내외 동향
- 국외 동향
- 국내 동향

⊙ TPC 및 SPEC의 신규벤치마크 기준 분석

- TPC-E, TPC-APP, SPECjbb2005, SPECjappserver2004, SPECWeb2005 모델분석
- 모델의 시사점과 적용방안

⊙ 기존 규모산정지침으로의 적용방안 제시

- 적용 시 고려사항
- 활용방안

본 연구에서 제시하고 있는 결과물은 성능벤치마크에 대한 문헌적인 연구가 미약한 국내 상황에서 관련 연구자 및 업계종사자들에게 좋은 지침서로 활용될 수 있음은 물론 향후 정보시스템 규모산정 지침을 보완을 위한 기본적인 추진 방향을 결정하는데 기본자료로 활용될 수 있을 것이다. 또한 이를 한국정보사회진흥원에서 운영 중인 e-Sizing 포탈을 이용해서 제공함으로써 정책담당자뿐 아니라 현업의 사용자들이 『정보시스템 구모산정 지침』에 관심을 증폭시키는 계기가 될 것으로 기대하며, 아울러 본 연구결과가 바람직한 홍보자료로 활용될 수 있을 것이다.

본 연구의 한계로는 전문가를 중심으로 한 제한적 범위 내에서 벤치마크기관에서 제시하는 모든 문서를 대상으로 분석을 수행하지는 못했다. 따라서 본 연구결과를 기초로 향후 기존 지침에 대한 변경 시 추가적인 검토가 필요할 것으로 보인다.

사. 소프트웨어 제품을 위한 평가 선정 모형[박호인97]

1) 연구의 배경

⊙ 소프트웨어 평가 및 선정의 어려움

- 다양성으로 인한 선택의 범위가 넓다
- 급격한 정보기술의 변화 속에서 다양해진 하드웨어, 운영체제, 소프트웨어 간의 호환성 부족
- 소프트웨어 제품 및 품질의 평가·선정에 관한 전문적 기술 부족
- 소프트웨어 제품 및 품질을 측정·평가할 수 있는 구체적인 기준이나 표준 부족
- 소프트웨어 평가에 사용되는 자료의 주관적이고 정성적임

⊙ 소프트웨어 제품의 특성

- 다양성과 무형성
- 상충성(conflictivity)

⇒ 다속성 의사결정(multiple attribute decision making)

2) 소프트웨어 평가속성 및 평가체계

⊙ 소프트웨어 제품평가를 위한 일반속성

: 경제적 요인, 자원적 요인, 품질적 요인의 혼합 형태로 구성

※ 소프트웨어 제품(software product)

사용자에게 인도할 것으로 지정된 컴퓨터 프로그램, 절차, 관련 문서, 그리고 자료의 전체집합 (ISO 9000-3(1991))

<표50> 평가 속성

연 구	주 요 속 성	특이사항
Brownstein & Lerner(1982)	기능적 요구사항 특성 설계개념 특성 비용 특성 공급자 특성 지원 특성	요구사항분석 및 문서 검토
Sanders, Ghandforoush & Austin(1983)	핵심요인 주관적 요인 객관적 요인	Brown-Gibson의 다차원 입지모형 사용

⊙ 소프트웨어 품질중심의 평가속성

※ 소프트웨어 제품(software product)

명시적 그리고 묵시적 요구를 만족시키는 소프트웨어 제품의 능력과 관련된 소프트웨어 제품의 특성 전체 (ISO 8402(1994))

- Boehm의 품질모형 : 최초로 소프트웨어 제품의 품질을 정량적으로 측정·평가할 수 있는 품질모형 제시

<표51> 평가체계

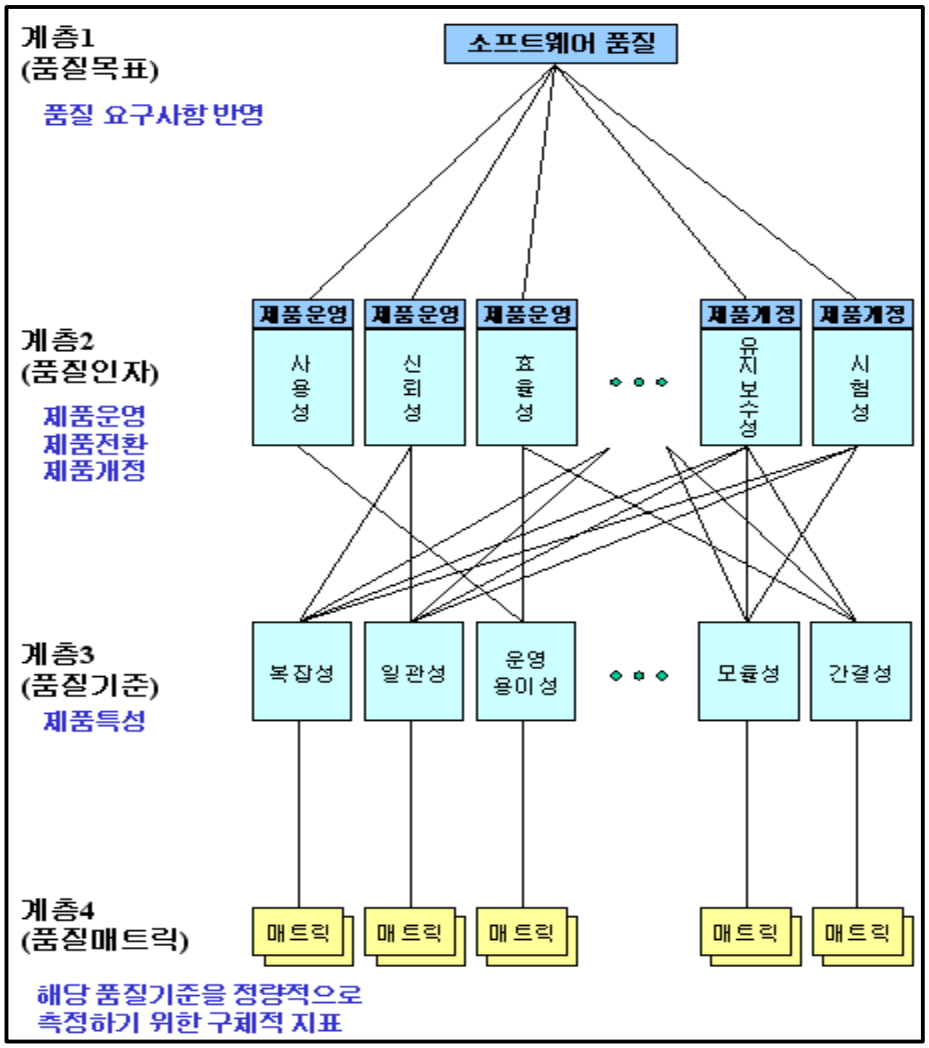
연 구	주 요 속 성	특이사항
Tally(1983)	문서 입출력 전체적 특성(설치용이성, 데이터보안, 데이터접근성, 데이터 손상 등)	워드프로세싱, 스프레드쉬트 소프트웨어별 필요한 세부속성 제시 데이터관리
Zahedi(1985)	기능적 요인 물리적 요인 비용 편익	계층적 분석과정에 의해 가중치 산정
미 상무성 Frankel(1986)	사용자 인터페이스 기술적 특성 문서, 교육 및 훈련	상세한 산정절차를 제시하고, 기회비용 산정 및 비용-편익분석을 과정에 포함시킴
NASA Kuan(1986)	사용편의성 사용자친숙성 공급업자 지원 문서, 기계호환성	최종 사용자 관점 강조
Visker & Bree(1987)	가용성 시간유효성 재무적 요인 조직 적합성	중·소규모 기업을 위한 평가속성을 제시
Lucas, Walton & Ginzberg(1988)	소프트웨어 품질 사용자 요구사항 조직적 특성	
Davis(1989)	유용성 사용편의성	특성별 6개 부특성으로 구성된기술수용모형(TAM:Technology Acceptance Model) 제시
USACERL Meier & Williamson(1989)	하드웨어 요구사항 공학기능 시스템 요구사항 공급자 지원, 가격	토목기사를 지원하는 마이크로컴퓨터용 소프트웨어에 초점을 둠
Subramanian & Gershon(1991)	성능 조직적 적합성 시스템 품질 사용자 친숙성 인터페이스	
Adeli & Wilcoski(1993)	사용자 인터페이스 대화식 그래픽 이식성 재설계 관리 프로그래밍 환경	CAD용 소프트웨어에 한정하여 평가속성을 제시
Jung & Yoon(1996)	기능성, 신뢰성 사용성, 효율성 이식성, 유지보수성	ISO 9126 표준적 품질모형을 평가속성으로 사용함

<표52> 평가 결과

품질특성	이식성 (portability)	활용성 (as is utility)			유지보수성 (maintainability)		
중간구조 / 기초구조	이식성	신뢰성	효율성	인간공학	시험성	이해성	변경성
장치독립성	○						
자기포함성	○	○					
완전성		○					
확고·무결성		○		○			
일관성		○				○	
설명성			○				
장치효율성			○				
접근성			○	○	○		
통신성				○	○		
자기기술성					○	○	
구조성					○	○	○
간결성						○	
명료성						○	
확대성							○

출처 : Gilles, A.C., Software Quality: Thoery and Management, Chapman and Hall, 1992, p.25.

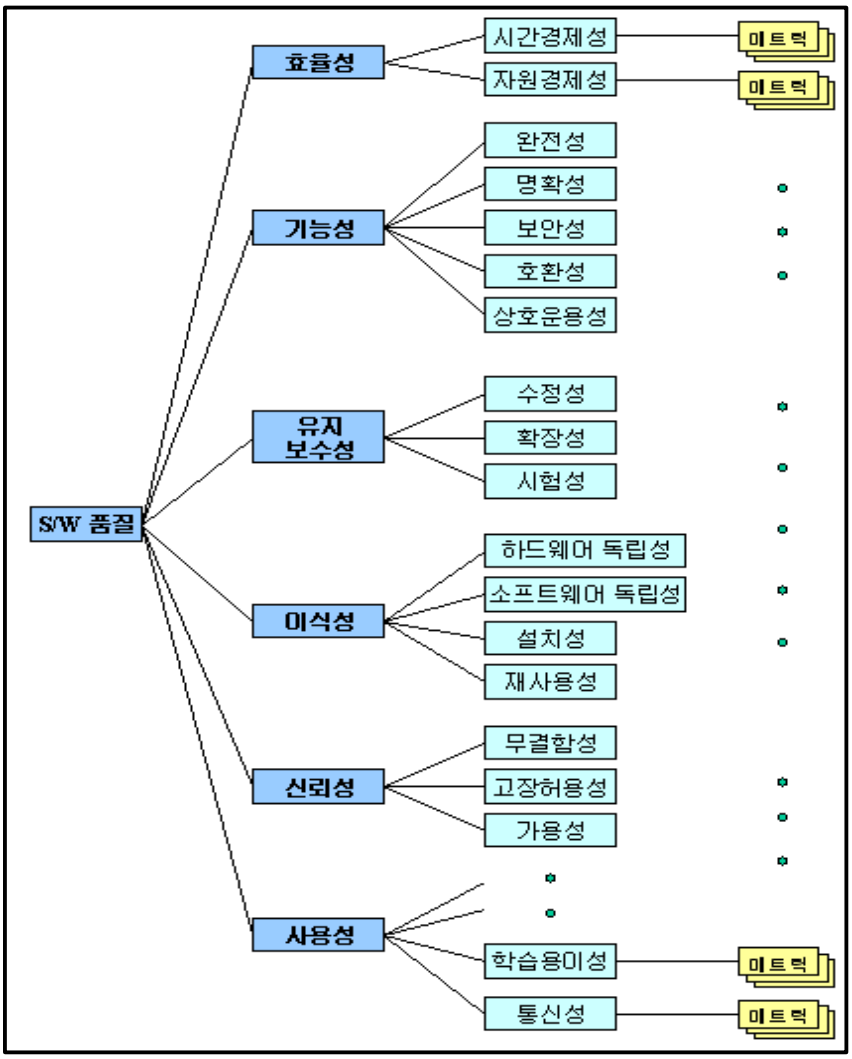
▪ MaCall의 품질모형 : 구매자 관점에서 요구된 품질 수준을 구체화하고, 소프트웨어 개발과정 중에 요구된 품질의 평가 가능성을 판단하는 지침으로 사용



<그림48> MaCall의 품질모형

▪ IEEE 품질모형 : IEEE 1061(1992)에서는 표준은 아니지만 정보참조로

서 사용가능한 품질 모형 예시, ISO 품질표준모형과 유사



<그림49> IEEE 품질모형

출처 : IEEE Standard for a Software Quality Matrics Methology, 1992, pp.19-20.

- ISO 품질모형 : 공정한 품질 측정 및 평가를 위한 객관적 표준 시 ISO 9126(정보기술-소프트웨어 제품평가-품질특성과 그 사용을 위한 지침)

<표53> ISO 품질모형

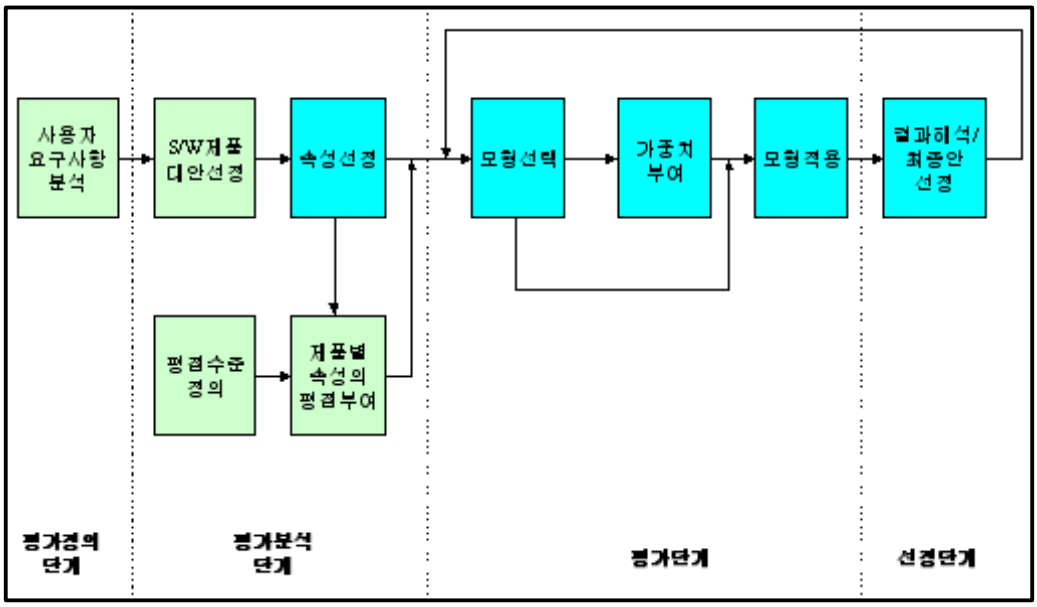
계층 1 (주품질특성)	계층 2 (부품질특성)
기능성	적절성, 정밀성, 상호운용성, 순응성, 보안성
신뢰성	성숙성, 고장허용, 회복성
사용성	이해성, 학습성, 운용성
효율성	시간행동, 자원행동
유지보수성	분석성, 변경성, 안정성, 시험성
이식성	적응성, 설치성, 적합성, 대체성

- 내부품질특성 제시 : 측정·평가에 이용(완정성, 추적성, 일관성 등)

3) 소프트웨어 평가 및 선정시스템

⊙ 소프트웨어 제품의 평가·선정시스템

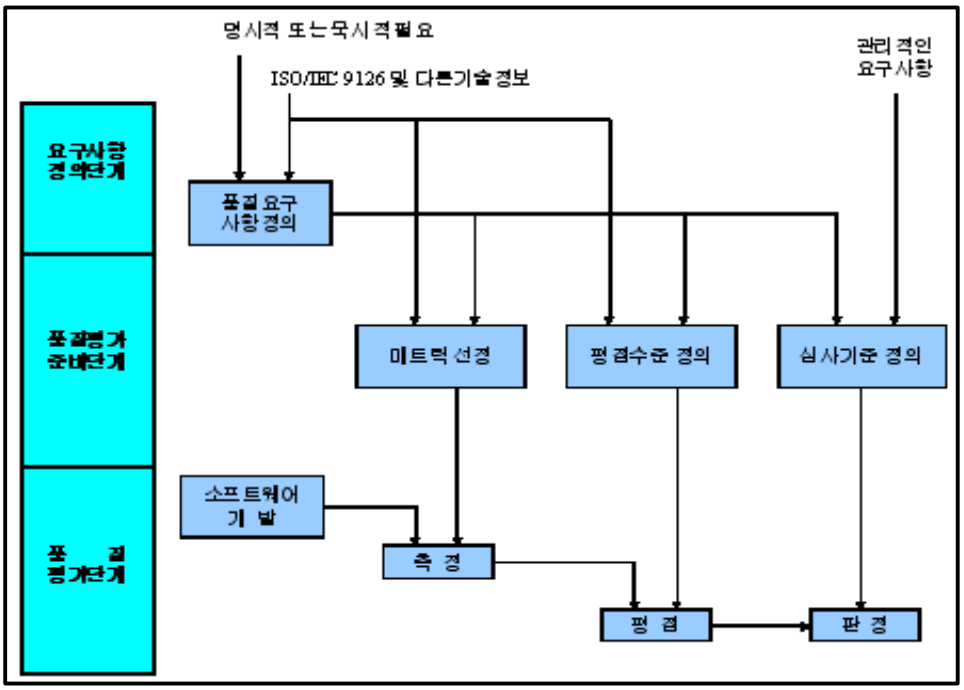
- 상용화된 범용의 소프트웨어 제품의 선정 시 이용 가능한 다수의 제품 중 사용자의 요구사항이나 평가목적을 만족시키는 단일제품 선정(Jung and Park, 1996)
- 다수의 제품 중에서 단일의 대안 선정에 초점을 맞추어 평가 진행



<그림50> 소프트웨어 평가 단계

⊙ 소프트웨어 품질의 평가 시스템

- 구매자 입장에서 계약에 의해 특정 소프트웨어를 외부에 용역 의뢰하여 인수에 앞서 계약내용을 준수하고 있는지를 확인·검증하거나, 개발자 입장에서 특정 소프트웨어를 개발하여 배포하기에 앞서 품질 요구사항을 만족시키고 있는지를 확인·검증(ISO 9126, 1991)



<그림51> 평가 시스템

출처 : ISO/IEC 9126, Information Technology – Software Product Evaluation – Quality Characteristics and Guidelines for Their Use, ISO. 1991, p6.

4) 계층적 분석과정(AHP)

복잡한 문제를 작은 부분으로 나누어 단순화시키고 체계화시키는데 적합한 의사결정 방법. (Analytic Hierarchy Process)

※ 가중치 : 각 속성의 다른 속성에 대한 상대적인 중요도를 표시 가치판단의 근본

⊙ 적용분야

- 사무정보시스템의 선정(Seidmann and Arbel, 1983)
- 마이크로 컴퓨터의 선정(Seidmann and Arbel, 1984)

- 원격통신 기술의 평가 및 선정(Douligeris, 1994) 등에 적용

⊙ 연구분야

- Zahedi(1985) : 마이크로 컴퓨터용 데이터베이스 관리시스템 평가
- Zahedi, Ashrafi(1991) : 사용자의 소프트웨어 효용(utility)을 Ashrafi로 프로그램과 모듈의 상대적 중요도 산정
- 정호원, 이종무(1996) : 품질특성의 가중치 산정에 활용한 연구

⊙ 적용절차

- 문제의 계층적 표현 : 문제의 목표를 정의하고 목표에 영향을 미치는 관련 속성들을 계층적으로 세분화하여 의사결정구조를 설정한다.
- 속성(요소)간 쌍대비교(pairwise comparison) : 작성된 계층구조를 바탕으로 계층별로 쌍대비교를 시행한다. AHP는 동일한 단계에 있는 요소들 사이의 중요도를 측정하는 방법 및 척도로 특징지어질 수 있다.
- 고유값에 의한 가중치 계산방법 : 추정된 가중치와 추정된 최대 고유값 산정

⊙ 일관성 측정

쌍대비교에서의 일관성 정도를 측정하는 도구로 일관성 비율 (Consistency Ratio: C.R.) 제공

5) 평가모형의 이론적 배경

<표54> 평가 모델 이론적 배경

연 구	적 용 모 형	적 용 제 품
Brownstein & Lerner(1982)	선형가중모형 (속성별 허용기준 제시)	모든 소프트웨어 제품
Sanders, Ghandforoush & Austin(1983)	다차원 입지선정모형	재무 및 회계용 소프트웨어
Edmonds & Urban(1984)	선형가중모형	요구문석 및 명세작성을 위한 도구
Zahedi(1985)	AHP	PC용 DBMS
NASA Kuan(1986)	선형가중모형	통계패키지, 통신용 소프트웨어, 문서작성 프로그램 등
Anderson(1989)	휴리스틱 모형	스프레드쉬트 소프트웨어 재무관리 소프트웨어
Eskenasi(1989)	최대최대모형과 유사	
USACERL(1989)	선형가중모형 (속성별 허용기준 제시)	토목공학용 소프트웨어
Anderson(1990)	선형가중모형 선형배정모형 최대최대모형 EBA 사전식 순위법	문서작성, DBMS 스프레드쉬트, 재무계획용 소프트웨어
Subramanian & Gershon(1991)	ELECTRE I	CASE 도구
Adeli & Wilcoski(1993)	선형가중모형	DOS용 CAD 소프트웨어
Jung & Park(1996)	타협해법 휴리스틱 모형	데이터베이스 모델러

6) 평가모형의 유형 : **평점간의 절충 허용여부에 따라 구분.**

⊙ 보상모형: 각 속성간의 절충이 허용되는 모형으로 대수적 접근방법 이용 소프트웨어 제품의 전체적인 품질이나 성능 중시되는 경우 사용

- 선형가중모형(linear Weighted attribute model): 소프트웨어 제품에 대한 대표적 평가모형. 모든 속성과 사용자가 부여한 가중치를 함께 고려하여 최종안 선정
- 선형배정모형(linear assignment model): 모든 속성과 모든 소프트웨어 제품을 동시에 고려하여 각 제품의 유일한 순위 결정

- 가중차이모형(additive-difference model): 대안간의 평점을 쌍대비교하여 차이의 합계를 구하고, 그 결과를 비교하여 최종안 선정
- 타법해법(compromise programming): 이상적 지점(ideal point)으로부터의 거리개념에 기초한 모형 이상적 지점으로부터의 거리가 최소인 대안 선정

⊙ 비보상모형 : 사용자의 요구사항을 반영하기 위해 최소 허용 한계값을 정하여 평가하는 기술적 모형(descriptive model)이 주를 이룬다. 평가속성의 수가 증가되고 있는 추세에서 비보상모형 필요성 증대

- 지배 모형(Dominance Model): 대안의 초기 비교과정에서 주로 이용. 지배모형은 지배당하는 대안을 기존의 대안집합에서 제외함으로써 비열등해 집합을 구한다. 축소된 대안은 그 밖의 다른 모형에 적용 최적안 선정
- 최대최대 모형(Maximax Model) : 가장 간단하고 쉬운 비보상방법, 모든 속성의 가중치는 동일하다고 가정. 속성평점의 변이가 적을 때, 속성의 중요도가 거의 같을 때, 비교적 낮은 하나의 속성이 사용자 만족에 심각한 영향을 주지 않을 때 적합.
- 사전식 순위법(Lexicographic Ordering) : 속성의 우선순위에 따라 순차적으로 한 번에 하나의 목표만을 최대화
- Elimination By Aspects(EBA) : 모든 속성에 최소 허용기준을 설정하고, 중요도에 따라 차례로 대안의 평점과 허용기준을 비교함으로써 최종안 탐색
- 접속모형(conjunctive model) : 모든 속성에 대해 허용기준을 설정하여 허용기준과 대안을 비교하여 대안을 축소해 나간다. 모든 허용기준을 만족하는 최종안 채택
- 비접속모형(disjunctive model) : 한 개이상의 속성에만 최소한의 허용기준을 지정, 중요도에 따라 허용기준을 비교해가면서 대안 탐색
- 휴리스틱 모형(heuristic model) : 여러 차원을 동시에 고려하여 대안을

평가하는 방법 일치도(concordance), 불일치도(discordance), 중요도(magnitude)

7) 모형의 비교

<표55> 모형비교

모형 \ 장단점		장 점	단 점
보상모형	선형가중모형	모형에 대한 이해와 적용이 용이함	모형을 정당화하기 위해 속성의 선정에 신중해야 함
	선형배정모형	유일한 순위의 부여가 가능	관련정보의 활용율 저조
	선형차이모형	대안간 비교에 의한 평가가 가능함	선형가중모형과 결과가 같은 반면 계산절차가 번거로움
	타협해법	관련정보의 활용도가 높고 비교가능한 복수의 차원 제공	복수의 최종안의 존재로 인한 혼란의 가능성 있음
비보상모형	지배모형	열등한 대안의 제거로 대안집합의 단순화가 가능	최종안의 탐색 불가능
	최대최대모형	대안을 신속하게 선정	사용자의 다양한 요구사항을 반영하지 못함
	사전식 순위법	소수의 속성만이 중요할 때 신속한 결정 가능	허용기준이 없어 관련정보의 활용이 저조함
	EBA	평가결과에 대한 이론적 정단화가 용이	초기의 대안이 선정되면 관련정보의 활용도가 감소할 수 있음
	비접속모형	소수의 속성의 만족이 필수적인 상황에 알맞음	다수의 최종안이 선정될 가능성이 높음
	접속모형	최소한 기준을 갖는 대안의 선정 가능성이 높음	모든 허용기준을 만족하는 대안이 없을 수 있음
	휴리스틱모형	다양한 차원의 종합적인 평가가 가능함	계산절차가 복잡하고 허용기준의 부여가 까다로움

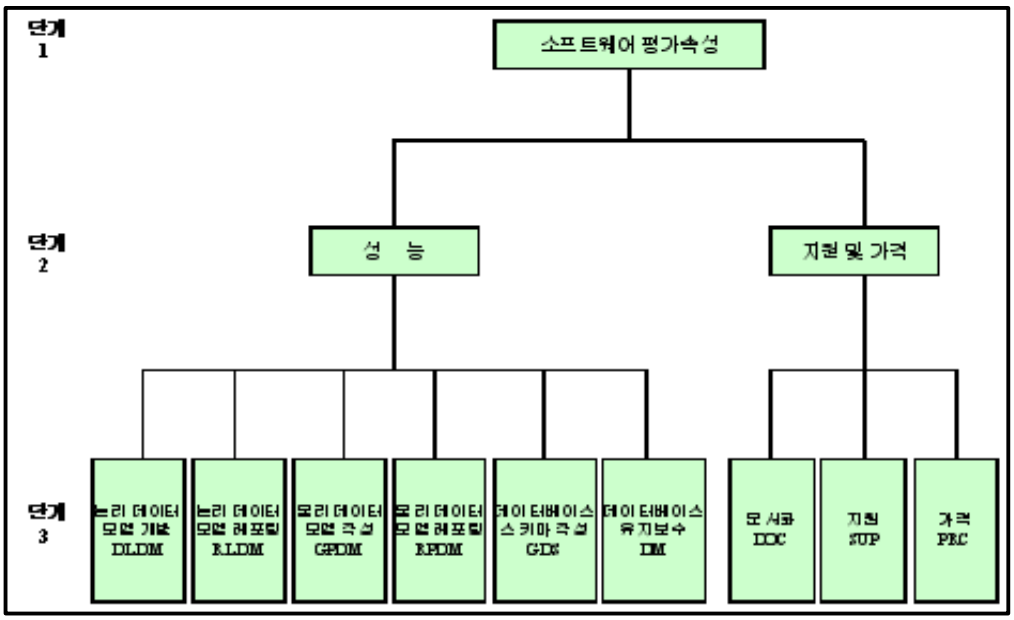
8) 모형적용 절차

⊙ 평가정의단계

- 평가대상 : 데이터베이스 데이터 모델러(database data modeler)
- 소프트웨어에 대한 품질, 성능, 제약요인, 예외적 요구 등을 조사

⊙ 평가분석단계

- 소프트웨어 제품 대안 선정
  - Bachman Information System사의 Bachman's Solution
  - Logic Works사의 Erwin ERX 2.5
  - Powersoft사의 S-Designor Enterprise 4.2.1
  - Intek Technologies사의 Vivid Clarity 1.1
- 속성의 선정



<그림52> 속성의 선정

- 평점수준의 정의 : 월등함, 매우좋음, 좋음, 만족함, 불만족함, 허용 불가함으로 구분

⊙ 평가단계

- 제품별 속성의 평점 부여 : Infoworld(Feb. 5, 1996)에서 발췌한 컴퓨터 관련제품의 비교 난에 수록된 평점자료 사용
- 모형의 선정 : 가중치 결정모형과 평가모형 선정, 모든 모형을 적용하여 결과 분석
- 가중치 부여 : AHP에 의해 산정된 가중치 사용
- 모형의 적용 : 선택된 평가모형에 주어진 평점자료를 적용하여 평가결과 산출

⊙ 선정단계 : 모형의 적용결과를 바탕으로 최종안 선정

9) 모형 적용상의 비교

– 보상모형 : 비교적 일관적인 결과 표출, 대부분의 관련정보 모두 활용

– 비보상모형 : 다양한 대안 선정됨, 다양한 선정규칙과 허용기준 존재

<표56> 비보상 모형

항 목 / 모 형	선 택 안	비 고
선형가중모형	Erwin ERX	
선형배정모형	Bachman's Solution	
가중차이모형	Erwin ERX	
타법해법	Erwin ERX	p=1,2, ∞를 사용
지배모형	Bachman's, Erwin, Vivid	
최대최대모형	Vivid Clarity	
사전식 순위법	Erwin ERX	DLDM,GPDM,GDS,DM,DOC,RLDM, RPDM,SUP,PRC
EBA	Bachman's Solution	DLDM(6),GPDM(6),GDS(6),DM(5), DOC(6),RLDM(6),RPDM(6),SUP(5), PRC(6)
접속모형	Erwin ERX	DLDM(6),GPDM(6),GDS(6),DM(5), DOC(5),RLDM(6),RPDM(6),SUP(5), PRC(6)
비접속모형	Bachman's Solution	DLDM(6),GPDM(6),GDS(6),DM(5), DOC(5),RLDM(6),RPDM(6),SUP(5), PRC(6)
휴리스틱 모형	Erwin ERX	T_s=0.5, T_g=0.05, T_d=0.5

10) 결론

⊙ 본 연구를 통해 제시된 시스템과 모형은 소프트웨어 평가·선정 문제해결을 위한 객관적이고 종합적인 기본 틀을 제공

⊙ 가중치는 일관적이고 객관적인 결과를 얻기 위해 중요

⊙ 적용환경에 알맞은 방법의 선정이 요구됨

⊙ 모형의 결정요인은 사용자의 요구사항과 평가목적이다.

⊙ 본 연구에서 제시한 선정기준은 사용자의 모형에 대한 활용성을 제고할 수 있는 지침역할

⊙ 다양하고 객관적인 모형 사용으로 요구사항에 근접하는 대안 선정

가능. 평가과정의 객관성과 최종대안의 신뢰성 향상

아. 소프트웨어 품질평가를 위한 최적화 모델[최석진92]

1) 서 론

⊙ 연구의 배경 및 목적

개발제품의 품질을 높이기 위하여 다양한 개발 기법과 개발 도구 및 개발 체계 등의 개발환경에 대하여 연구가 계속 수행되었으며 개발 제품의 품질 평가에 대한 연구도 계속 되어왔다. 이 논문에서는 소프트웨어 제품을 평가하기 위한 최적화 모델을 제안하며 제안된 모델은 기존의 모델과는 달리 소프트웨어 제품에서 요구되는 품질 특성을 정확히 정의하고 설계한 다음 이러한 특성을 만족하는 제품 평가 모델을 설계 적용하여 제품 특성을 충분히 구현한 소프트웨어를 개발하기 위함을 목적으로 한다.

일반적인 모델에 사용자의 요구를 반영하는 것은 일반적 특성의 제품 개발은 유효하지만 절실히 요구되는 것은 제품의 특성과 사용자 환경 및 요구 조건에 최적한 평가 모델을 설계하고 설계된 모델에 적용하여 제품을 정확히 평가하여 소프트웨어 제품개발의 생산성 향상과 고품질의 제품을 개발하는데 있다

⊙ 연구의 범위 및 방법

소프트웨어 품질보증 및 품질평가에 대한 기술, 방법 등을 조사하고 기존의 개발된 소프트웨어의 품질평가 모델을 연구 분석하여 소프트웨어 제품의 품질을 예측하기 위하여 제품의 특성 및 사용자의 요구 사항에 최적한 품질 평가 모델을 설계하여 제품의 의미

를 정확히 전달할 수 있는 고품질의 제품을 개발하기위한 최적 평가 모델을 설계하고자 한다.

2) 소프트웨어 품질보증 및 품질평가

⊙ 품질 보증의 개념

소프트웨어 품질 보증은 소프트웨어 라이프사이클의 각 단계에서 수행되는 인증 및 검증에 밀접한 관계가 있으며 품질 보증을 성공적으로 이끌기 위해서는 인증과 검증에 관련된 품질을 측정할 수 있는 표준이 제정되어 있어야 하고 사용자의 요구사항이 적절히 만족되었는지에 관한 측정도 함께 이루어 져야한다.

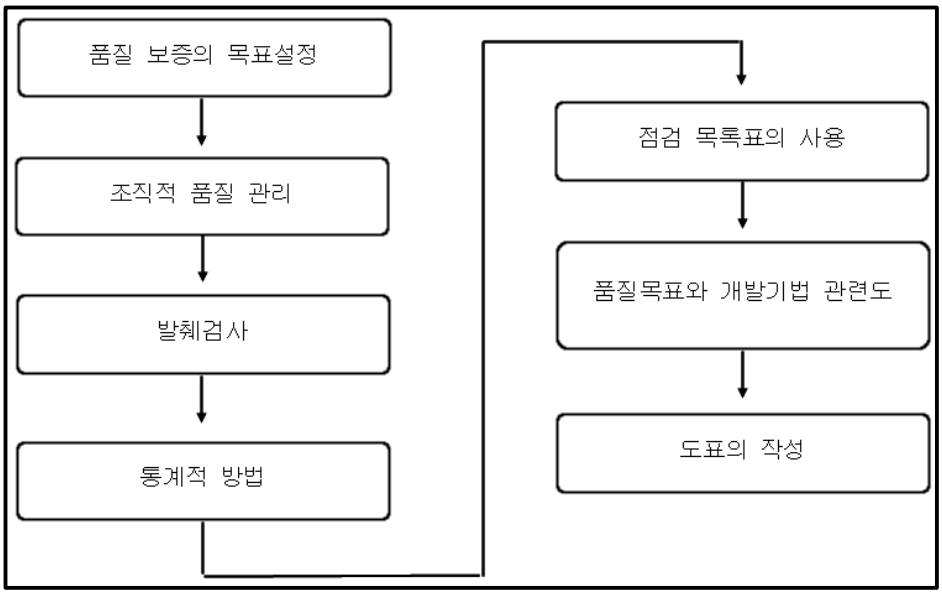
⊙ 품질 보증의 평가 목적

개발 하고자 하는 소프트웨어가 목표로 하는 품질 특성을 말하며 개발 제품 품질 평가 척도의 최상위 레벨인 품질 인자중 적용 가능한 것들의 집합이다. 품질 목표는 시스템의 라이프사이클 전체에 걸쳐서 개발 및 평가의 지표가 되므로 프로젝트의 특수성 등 여러 요인을 고려하여 실현 가능하도록 설정되어야 한다.

⊙ 품질 보증의 계획

소프트웨어 라이프사이클 동안의 품질 보증 목적은 품질에 영향을 주는 모든 활동의 권리를 위한 근본과 관련된 모든 것에 대한 보증을 하는 것이고 설정된 품질이 달성되어지는 것을 조정하는 것이다.

⊙ 품질 보증 실시 절차



<그림53> 품질 보증 실시 절차

⊙ 소프트웨어 품질평가

소프트웨어의 품질을 평가하기 위하여 그 품질에 미치는 모든 분야를 포괄하는 평가 척도를 통해 평가되어야 하며 소프트웨어 품질의 평가는 개발주기 전 단계에 걸쳐 이루어지므로 특정 개발 단계의 특성에 맞는 적절한 기법의 사용이 요구된다. 또한 품질평가 방법에는 여러 가지가 있으나 이 논문에서는 체크리스트에 의하여 기준 및 목표를 설정하고 소프트웨어의 품질 정량화를 이용하여 제품의 특성을 계량된 수식으로 표현하였다.

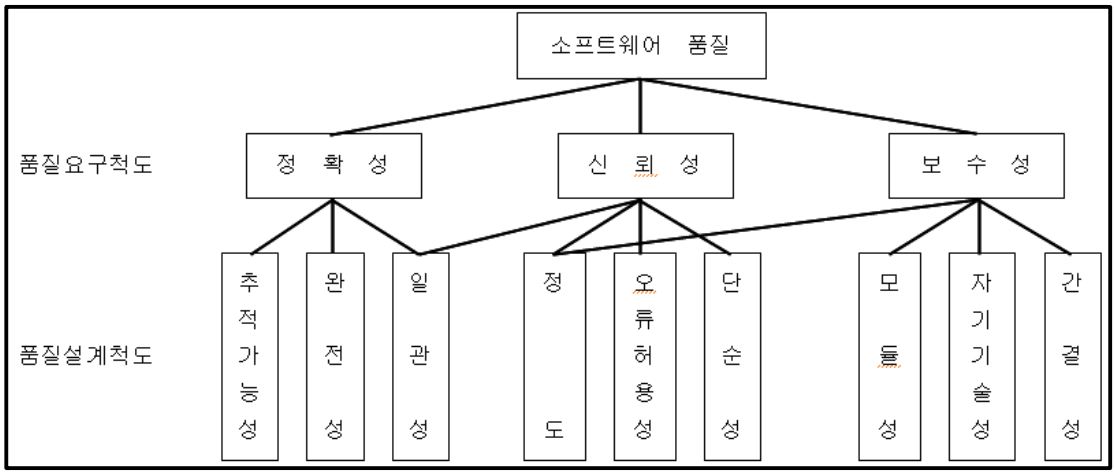
① 개발과정 품질평가 척도

개발 과정에 대한 평가 척도는 소프트웨어 품질에 영향을 주는 프로젝트 기반구조의 확립 수준을 파악하는 것으로 시작되며 개발과정 품질 척도는 이들 기반구조의 형성 수준과 적용수준에 대한 평가항목으로 구성된다. 소프트웨어 품질 평가 시 이들 평가 항목

을 적용함으로써 품질에 영향을 미치는 개발 과정의 특성을 정량적으로 측정할 수 있다

② 개발제품 품질평가 척도

개발 제품에 대한 평가 척도는 소프트웨어 개발주기의 각 단계별로 생산되는 모든 산출물을 대상으로 품질을 평가하기 위한 척도이다. 개발제품 품질 평가 척도는 개발제품의 품질평가를 위한 개발제품의 품질계층구조의 최하위 레벨로 소프트웨어 품질 특성을 정량적으로 측정하기 위한 수단이다.



<그림54> 품질척도의 구조

⊙ 소프트웨어 품질평가 방법

품질평가의 일반적 방법 중 Walkthrough 테스트를, 그리고 체크리스트에 대해 이들 품질 평가 방법들이 품질평가를 대신할 수는 없으나 품질 보증을 위한 과정 중에 이용될 수 있으며 이용의 필요성에 충분한 가치가 있으므로 이에 대한 고찰할 필요가 있다 하겠다.

⊙ 품질평가 정량화

소프트웨어 품질보증이 활성화되기 위해서는 품질의 정량화를 통해 소프트웨어의 가시성을 높여 품질을 정확히 표현하고 품질평가 활동의 객관성과 공정성을 높여야 한다. 소프트웨어 품질의 정량화란 불가시적인 소프트웨어의 품질을 가능한 한 가시화시키기 위하여 계량화된 모델을 통해 품질을 표현하고 평가하는 것을 말한다.

3) 기존 소프트웨어 품질평가

품질에 대한 평가 척도, 품질목표 및 품질평가 기준에 따라 품질을 평가하기 위하여 평가모델이 필요하며, 품질을 평가하는 요소들은 품질을 보는 사람에 따라 많은 차이를 나타내고 있다.

⊙ SQMAT 모델

SQMAT는 신뢰성이 높은 품질의 제품을 생산하기 위하여 개발된 소프트웨어 품질계측, 보증기술이며 소프트웨어 품질을 다각적인 시점에서 정량적으로 평가하고 보증하기 위한 개발 기술로써 소프트웨어 개발공정을 정의 공정, 설계공정, 제조공정, 검사공정으로 구분하고 있다.

▪ 요구조건

SQMAT에서 품질계측, 보증기술에 요구되는 조건은 적용성, 용이성, 신뢰성, 적극성, 정량성 등이 있다.

▪ 모델의 특징

- 품질요구척도, 품질설계척도 및 품질평가척도의 3가지 레벨로 이루어지는 구조화된 척도를 이용한다.
- 종래의 검사공정별로 리뷰라고 하는 피이드 백 제어에 더하여 피드포워드의 원리에 근간한 목표관리의 사고가 이용되고 있다.

- 소프트웨어 목적에 따라 중요한 품질특성은 상세히, 그다지 중요하지 않은 품질특성은 간단히 평가할 수 있도록 3가지 평가방법을 마련하고 있다.
- “눈으로 보는 관리”를 용이하게 하기 위해서 목표와 계측결과를 그래프로 그린다.

▪ 품질척도
- 품질요구척도는 소프트웨어 이용자 관점에서 보아 어떠한 품질을 얼마만큼 충족시키면 될 것인지를 검토해서 품질목표를 정하고 또한 각 공정별로 품질목표를 충족하고 있는가 아닌가를 판정하기 위한 8가지 척도로 구성된다.
- 품질설계척도는 이용자 쪽에서 본 품질목표를 달성하기 위해서 소프트웨어를 어떻게 만들어야만 하는가를 설계자 시점에서 여러 가지로 구체적으로 검토한 후 목표를 정하려 목표대로 작업이 행하여지는지 여부를 판정하기 위하여 사용하는 23가지 척도로 구성된다.
- 품질평가 척도는 각 공정에서의 작업결과를 실제로 계측하고 평가하는 척도이다. 또한 이 척도는 작업을 실시할 때에 품질목표를 달성하기 위한 지침으로서도 이용할 수 있다.

▪ 계측방법
- 정밀계측 : 중요도가 높은 품질척도에 대응한 품질의 계측에 적합하다. 평가척도별로 해당하는 곳에 각 건에 대하여 조건을 충족시키고 있는가 어떤가를 조사한다. 평가척도에 의해 [Yes] [No]로 평가하는 것과 4:대단히 만족, 3;만족, 2:약간 만족, 1:불만의 4단계 중 어는 쪽이든 간에 평가하게 된다.
- 일괄계측 : 중간적 중요도의 품질척도에 대응한 품질계측에 적합하다. 평가척도별로 해당하는 곳을 통해 평가대상 전체에 대해서 앞서 언급한 4단계 평가를 실시한다.
- 간이계측 : 중요도가 그다지 높지 않은 품질척도에 대응한 품질의 계측에 적합하다. 품질평가척도를 고려하면서 평가 설계척도의 레벨에서 4단계 평가를 실시한다.

▪ 문제점

- 적용범위와 품질평가자의 범위가 제한되어 일반성이 결여되어 있다.
- 사용자 요구품질이 잘못 나타날 수 있다.
- 품질기준이 모델 간에 상이하며 유동적이다.
- 정확한 품질검사가 어렵다.

⊙ 후지쯔 모델

후지쯔사의 모델은 품질의 기본이 초기공적에서 결정되는 것이고 개발계획, 개발목표라 하는 소프트웨어 개발의 원류에서 관리를 중시한다. 소프트웨어 개발의 각 단계에서 목표치를 설정하고 소프트웨어 개발과정에서 얻어진 정보를 품질 데이터로 파악하여 분석한다.

▪ 요구조건

후지쯔 모델의 요구조건은 성능, 신뢰성, 조작성, 호환성, 정합성, 보수성, 정보의 양등이 있다.

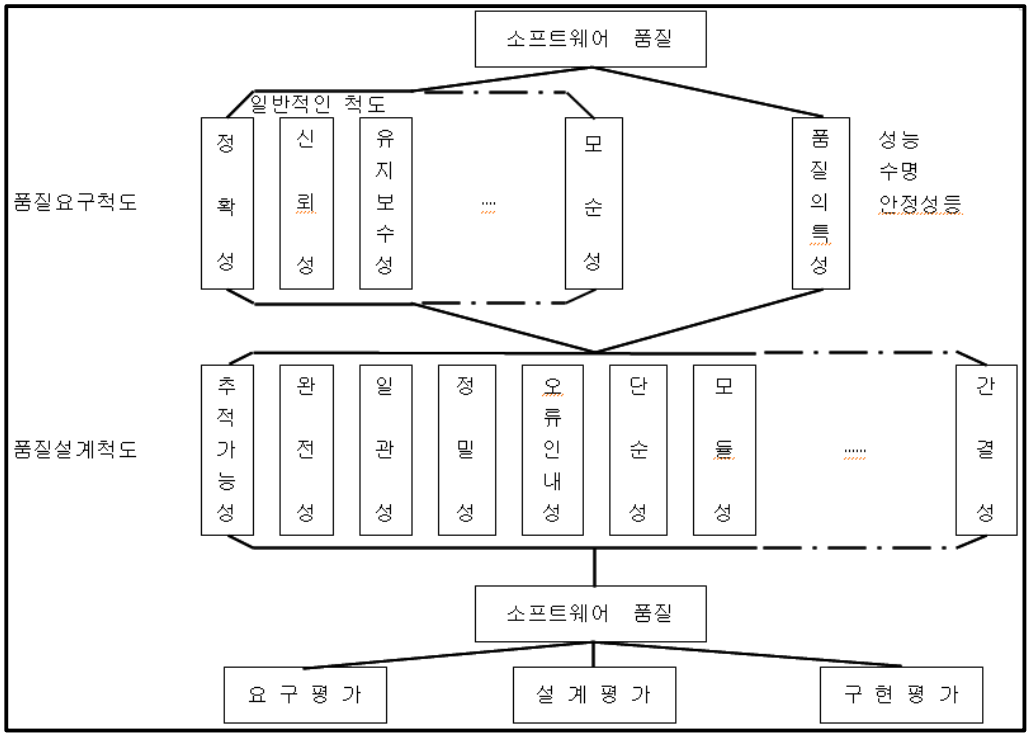
▪ 모델의 특징
- 품질특성과 품질수준 그리고 품질요소와의 관계를 나타내는 형태의 모델이다.
- 적합품질, 설계품질, 매력적 품질, 최초의 품질 관계를 나타내는 형태의 모델이다.
- 품질특성의 분류가 다른 모델보다 적다.
- 각각의 품질특성이 갖는 의미는 상당히 포괄적이다.

▪ 문제점
- 품질요구척도와 품질설계척도간의 구분이 없다.
- 개발자 스스로 품질을 설정, 평가하는 형태이다.
- 요구에 대한 정확한 품질특성을 판단하기 어렵다.

4) 제시된 소프트웨어 품질평가 모델

⊙ 모델의 설계

품질은 사용자의 요구사항을 만족시켜 주는 제품 또는 구성품의 성능, 수명, 안정성등 특성의 총체로 이루어진다. 이와 같은 품질을 확보할 수 있도록 각각의 유형별 제품의 특성에 맞는 최적의 모델을 설계하였다.



<그림55> 새로운 품질평가모델

소프트웨어의 품질계측, 보증기술에 요구되는 조건은 <표57>에 나타나 있으며 미국 품질보증연구소의 품질요구척도를 기준으로 제시하였다.

<표57> 품질목표의 정의

품 질 목 표	정 의
정 확 성	소프트웨어가 사용자 요구와 사용자요구상세서를 만족시킬 수 있는 정도
효 율 성	시스템적 입장에서(하드웨어포함)소프트웨어가 충분한 기능을 수행하는지의 정도
유 연 성	소프트웨어를 개선하기 위해 요구되는 노력의 정도
무 결 성	소프트웨어의 DATA CONTROL이나 자료검색시 보안을 완전하게 해주는 정도
결 합 성	INTERFACE 기능을 강화하여 프로그램 및 시스템간의 결합을 쉽게 해주는 정도
유 지 보 수 성	소프트 웨어의 오류를 찾아내고 이를 보수하는데 요구되는 노력의 정도
이 식 성	하드웨어를 바꾸었을 때 소프트웨어가 수행될 수 있게하는데 요구되는 노력의 정도
신 뢰 성	시스템의 실패없이 예상된 기능을 정확하게 수행할 수 있게 하는 노력의 정도
재 사 용 성	소프트웨어의 표준화를 통해 모듈의 재상용이나 다른 PACKAGE에서 사용될 수 있는 정도
검 사 성	코드의 구조와 정확성을 검사하기 위해 요구되는 노력의 정도
유 용 성	시스템의 입, 출력을 배우고 이를 사용하기 위해 사용자가 들어야 할 노력의 정도

품질설계척도는 소프트웨어의 품질을 가시적으로 파악하기 어려운 품질목표를 보다 기술적인 측면으로 구체화시킨 것이다.

<표58> 품질기준의 정의

품 질 기 준	정 의
추 적 가 능 성	작업 적용환경에 관한 요구에서 구현된 프로그램으로 link를 제공하는 속성
완 전 성	요구되는 기능의 완전한 구현을 준비하는 속성
일 관 성	일관성 있는 설계, 구현기법, 문서화를 준비하는 속성
정 밀 성	계산값과 결과치들의 요구된 정밀도를 제공하는 속성
오 류 인 내 성	실제의 작업환경하에서는 약간의 오류에도 작업의 계속성을 제공하는 속성
단 순 성	가장 알기쉬운 방법으로 기능들을 수행하는데 제공되는 속성
모 듈 화	고도로 독립된 모듈들의 구조를 제공하는 속성
일 반 성	실행되어진 기능들에게 약간의 여유폭을 제공하는 속성
확 장 성	data나 program function들의 확장을 위해 준비되는 속성
도 구 성	소프트웨어의 사용평가나 오류의 명시를 제공하는 속성
자 기 표 현 성	어떤 하나의 기능수행에 대한 설명을 제공하는 속성
실 행 효 율 성	처리시간을 최소화 하는 속성
보 안 성	소프트웨어, DATA CONTROL과 PROTECTION을 위해 제공되는 속성
감 사 성	소프트웨어, DATA 결과들의 쉬운검사를 준비하는 속성
운 용 성	소프트웨어의 운용에 관한 작업계획과 과정을 결정하는데 제공되는 속성
훈 련 성	현재의 환경에서 새로운 시스템의 변화를 준비하는 속성
S/W 시스템 독립성	소프트웨어 환경에 의존도를 결정하는 속성
H / W 독 립 성	하드웨어에 의존도를 결정하는 속성
전 달 공 통 성	표준 프로토콜과 인터페이스 루틴을 사용하는 속성
자 료 공 통 성	표준 DATA 표현의 사용을 제시하는 속성
간 결 성	최소량의 코드내에서 한기능을 수행하는 속성

⊙ 모델의 평가방법

품질인자 또는 품질기준들 간의 상대적인 중요도 수준에 따라 가중치를 부여하고, 평가기준별로 요구정도, 설계내용 및 구현시의 평점을 계산하여 품질을 평가하고 가시성을 높이기 위하여 레이더 그래프로 표시한다.

▪ 가중치 부여

중요도 순위에 따라서 매우 중요한 요구척도는 2.0, 중요한 요구척도는 1.5, 일반적인 요구척도는 1.0으로 부여한다.

▪ 평점

각 평가 기준별 평점은 요구적합도, 설계적합도 및 구현적 합도에

따라 평점을 계산하여 산출한다.

▪ 적용 요소의 평점산출

각 제품의 평가척도에 따르는 평가 요소는 매우 적합한 경우 1, 적합한 경우 0.8, 보통은 0.5, 부적합은 0, 미적용은 공백으로 나타낸다.

▪ 품질목표 달성 수준

품질목표 달성 수준은 적용된 평가기준의 평점의 합계를 가중치 합계로 나누어서 구한다.

⊙ 기존모델과의 비교평가

기존의 모델들은 품질의 특성을 고려하지 않고 설계되어진 반면 제시된 모델의 경우 품질의 특성을 기초로 함으로써 평가의 기준 지침을 제공하고 있다.

<표59> 기존 모델과 제시된 모델과의 비교표

비 교 내 용	SQMAT	후지쯔	제시모델
1. 제품특성에 따른 정확한 평가	×	×	○
2. 개발환경 변화에 따른 상태 파악	△	△	○
3. 품질평가 기준 지침의 유용성	△	△	○
4. 적용범위 - 일반성 - 전문성	 ○ △	 ○ △	 ○ ○
5. 품질평가 모델의 용이성 - 사용자 - 설계자 - 개발자	 △ ○ △	 △ ○ △	 ○ ○ ○
6. 사용자의 기능적 요구사항의 정확한 평가	×	×	○
7. 요구품질 특성의 정확한 평가	△	△	○
8. 품질척도의 정량화 기법 사용	△	△	○
9. 유사제품의 결과 계속 적용	×	×	○
10. 사용자 요구, 설계 구현시 품질평가를 통한 제품품질 예측	△	△	○

○ : 양 호 △ : 보 통 × : 불 량

5) 결 론

품질 평가에 있어 품질 평가 요소별 가중치의 역할은 소프트웨어 특성별로 품질을 평가하는데 매우 중요한 역할을 하지만 정확한 가중치를 부여하는 방법은 수치적으로 표준화 되지 않아 가중치에 의해 품질의 평가가 달라질 수 있다. 가중치의 정확한 수치 표준화 작업이 계속 연구 되어야 할 분야이다. 소프트웨어의 품질 평가를 정확히 하기 위해서는 가능한 한 모든 측정 자료와 도구들을 전산화 시켜야 하며 소프트웨어 평가 모델이나 평가 방법 등을 구현한 품질 측정 도구의 개발과 적용은 소프트웨어 생산성과 품질을 높일 수 있는 최선의 방법이다

자. 소프트웨어 품질평가 투입요소 결정[이종무97]

1) 서론

⊙ 연구의 배경

소프트웨어 품질 표준에 근거한 구체적인 측정과 평가 방법의 제시와 활은 아직 미흡한 실정이다. 또한 제한된 예산 하에서 합리적이고 객관적인 품질평가를 위해서 요구되는 해당 투입요소(target input)의 우선순위는 기존 연구에서 제시된 바 없다

⊙ 연구의 목적

소프트웨어 품질평가의 객관성 확보를 위한 품질의 이해와 개발과정상의 품질 예측력 향상 방안의 마련, 소프트웨어 품질평가에 적합한 합리적인 가중치 결정방법의 선정, 그리고 체계적인 투입요소(target input) 결정을 위한 종합모형의 구체적인 제안과 사례적용 및 결과분석 등에 둔다.

⊙ 연구의 방법 및 구성

기업정보시스템 활동의 주된 목적은 대내적인 의사결정의 합리화와 대외적인 전략상의 경쟁우위 달성에 있다. 즉 경영 관리자 혹은 일반관리자들의 소프트웨어에 대한 요구와 필요성은 기업의 전략적 차원의 가치창조를 위한 목적으로 이해되어야 한다.

따라서 본 연구에서의 소프트웨어 품질평가에 대한 기본관점의 초점은 사용자의 의사결정 합리화와 기업 목표달성을 위한 대외 전략적 품질확보에 맞춘다. 이를 위해서 본 연구에서 채택한 구체적인 연구방법을 기술하면 다음과 같다.

첫째, 소프트웨어 품질평가를 위한 표준 혹은 지침과 품질 모형

및 품질특성에 관한 기존 문헌 연구결과들을 분석해본다.

둘째 사용자관점의 소프트웨어 외부품질특성과 개발자관전의 공학적 내부 품질특성들의 연결방법을 제안하여 이를 산업현장의 사례에 적용한다.

셋째 평가 및 선정의 의사결정과정에서 매우 중요한 가중치 경정방법의 객관적 선정을 위해서 기존의 다중목표 의사결정과 소프트웨어 평가에 관한 문헌연구들을 분석하여 소프트웨어 품질평가에 적합한 합리적인 가중치 결정방법을 선정 제시한다. 특히 소프트웨어 품질평가와 같이 가치판단의 주관성이 큰 경우 이를 정량적으로 우선순위화 하기에 적합한 가중치 결정이론을 확장하여 제안한다.

넷째, 기업의 가치창조 차원에서 소프트웨어 제품의 개발 생명주기 과정상의 특징을 고려한 품질 평가관점들을 체계적으로 구분하여 제시한다. 이의 현실적 적용을 위한 틀로서 투입요소 결정모형을 기존 문헌연구의 분석을 통하여 제안한다.

다섯째, 투입요소 결정모형을 특정사례에 적용시킨 후, 정량화 절차를 거쳐 구체적인 소프트웨어 투입요소의 우선순위를 정한다.

2) 가치기반 소프트웨어 품질과 품질모형

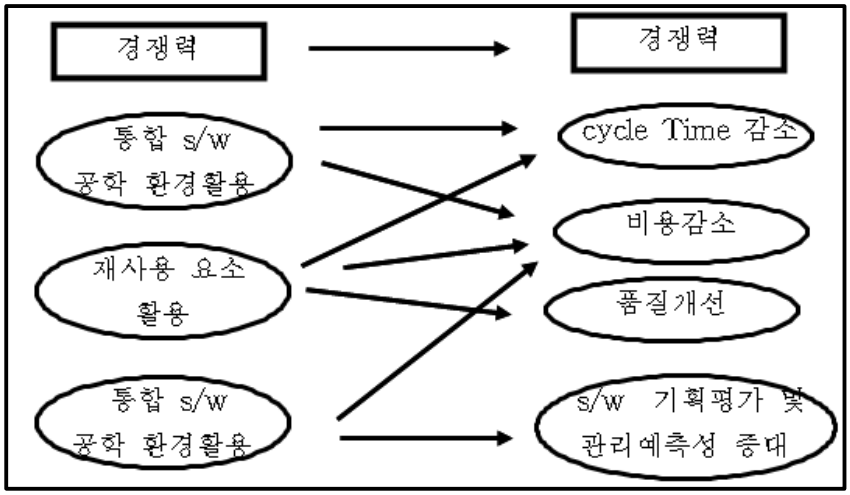
기업조직의 가치 극대화를 통한 경쟁력 강화를 위한 가치기반 소프트웨어 품질의 중요성과 표준 품질모형과 품질특성의 구성에 대해 기술한다. 특히 품질모형의 구체적 활용방법으로 개발자관점의 품질과 사용자관점의 품질을 연결하는 방법을 제시한다.

⊙ 가치기반 소프트웨어 품질

▪ 가치기반 품질의 중요성

과거와 달리 현대의 소프트웨어 개발 경영전략의 주된 특징은 복잡한 소프트웨어 제품 및 개발과정에 관한 이해와 경영목표 달성을 위한 품질관리의 필요, 관리 과정상 객관적 품질평가의 중요성, 그리고 이에 따른 품질개선의 노력 등으로 요약할 수 있다

특히 과거 소프트웨어 개발의 성공적 경영전략인 장기적 투자나 최고경영층의 이해와 지원의 단순한 특징에서 벗어나, 기업과 고객의 적극적인 요구 수용뿐만 아니라 가치에 기반한 일치된 목표의 달성을 중시한다. 그리고 개발 단계별 전사적 품질관리를 강조하는 과정 주심적 경쟁력 확보전략으로 변해 가고 있다. 결국 이러한 경쟁전략은 개발자 및 공급자 관점뿐만 아니라 고객 관점에서의 품질만족을 목표로 한다. 이는 가치기반의 품질관리의 개념을 실행함으로서, 경영가치의 극대화를 구체화시키려는 기업의 노력이라고 평가할 수 있다.



<그림58>소프트웨어 개발 경쟁력과 전략적 능력

- 가치창조와 전략적 품질

- 가치창조와 소프트웨어 품질관리

기업전략으로서의 가치창조의 의미는 고객의 제품 요구에 대한 기업의 보다 적극적인 공급자로서의 역할을 강조하는 것이다. 가치창조의 구체적인 개념 정의를 위해서는 다음과 같은 가치등식(value equation)을 활용할 수 있다.

고객가치= (품질 * 고객서비스) / (원가 * 시간) = 기업가치

* 품질과 고객서비스는 고객가치 창조에 정(+) 효과를 가져오는 반면에 원가와 시간적 요소는 부(-)의 효과를 가진다고 본다.
* 고객가치와 기업가치의 구분은 무의미하며 기업의 공급자가 추구해야할 기업 가치와 소비자가 느끼는 고객가치는 상호 연결되어 경쟁적 우위를 달성하도록 해야만 한다.

이는 새로운 시장전략으로서 과거의 단순원가, 생산성 및 품질 등의 일차원적 전략개념에서 대 고객 가치창조와 연결된 종합적인 품질개념의 중요성과 이를 구체화할 수 있는 품질관리 방안의 도입을 강조하고 있다. 이러한 관리방안으로는 품질 기능전개도 혹은 품질 가계도 등을 예로 들 수 있다.

이 개념은 올바른 고객의 요구사항을 규정하고 제품과 서비스의 개발주기 및 각 단계에서 이를 기업 내부의 요구로 변환시키기 위해 필요한 시스템이다. 즉 고객의 요구에 기초해 제품이나 서비스를 설계하고, 생산자나 공급자 조직의 모든 구성원들을 포함시키기 위한 시스템이다.

- 전략적 품질과 가치시스템

전략적 품질이란 고객관점의 제품과 서비스, 그리고 고객이 원

하는 품질을 제공함으로서 보유하게 되는 전략적 이익을 의미한다. 가치시스템은 구체적으로 공급자 가치사슬, 기업가치사슬, 유통경로 가치사슬, 그리고 구매자 가치사슬로 연결된다.

가치사슬상의 경영활동은 원자재구매, 재고관리 등 내 부적 물류활동, 제조, 생산, 운영등의 활동, 그리고 구매자, 즉 대 고객 서비스등의 활동으로 구성된다. 그리고 지원활동은 기업의 기반구조적 활동, 인전자원관리, 그리고 기술개발과 구매 활동 등으로 이뤄진다.

가치시스템과 관련된 품질 개선을 위한 국제적 지침으로는 ISO 9004-4을 참고할 수 있다.

품질관리 및 품질시스템에 관한 지침중의 품질개선을 위한 ISO 9004의 정의하고 있는 공급자사슬이란 공급자로부터 입력을 받아들이고, 여기에 가치를 부여하여 고객을 위한 최종산출물을 만들어 내는 일련의 공급자와 고객의 상호 관련된 가치부여 과정(프로세스)를 의미한다. 이는 공급자, 프로세스, 고객으로 구성된다.

즉, 가치시스템에 따른 가치사슬상의 제 활동은 기업 경영가치의 극대화를 위한 결합 활동으로 이해할 수 있다. 또한 전략적 기업 목표의 달성과 기업의 가치와 고객의 가치 창조를 연결 지어 구체적인 품질추구의 목표로 정의하여 활용할 수 있는 장점이 있다.

3) 소프트웨어 품질관리

⊙ 소프트웨어 품질표준과 관리의 필요성

고객서비스와 만족을 의식한 전략적 품질의 추구는 고객과 공급자가 공통의 품질관리를 행할 수 있는 객관적 표준이나 공통 기준의 마련을 통

해 더욱 구체화될 수 있다. 이러한 이유 때문에 품질표준의 이해와 구체적인 활용방안의 마련은 품질평가와 관리에 앞서 매우 중요하다. 즉 공급자 입장에서 객관적인 고객만족과 품질관리를 행할 수 있는 최상의 방안은 고객의 요구를 객관적인 품질표준과 품질모형에 입각해 수용해 나갈 수 있기 때문이다. 한마디로 올바른 소프트웨어의 품질평가를 위해서는 명확하고 객관적인 방법과 표준 혹은 기준의 제공이 요구되며, 제품의 사용목적이나 요구사항 및 표준과의 일치가 무엇보다 중요하다.

소프트웨어 품질과 특성에 관한 표준 용어의 정의는 품질측정과 평가의 객관성을 확보하고, 나아가 소프트웨어 개발 생명주기상의 설계와 검사의 일관성을 보장하기 위해 절대적으로 필요하다.

⊙ 체계적인 품질관리 및 평가방법의 요구

전략적 차원의 소프트웨어 품질개선과 관리방법에 관한 체계적 연구는 방법론 차원에서 다양하게 이뤄지고 있다.

체계적인 품질관리를 위해서는 사용자 요구사항의 정형화 명세방법에서 평가도구와 방법까지 매우 다양하지만 체계적이고 일관성 있는 관리방법이 요구된다

소프트웨어 품질관리방법 및 개선과정으로서 대표적인 방법은 품질개선 패러다임(Quality Improvement Paradigm:QIM)이나 전사적 품질관리(TQM)를 들 수 있다. Basili95품질개선 연구에서 제시된 QIM의 Shewart-Deming주기의 계획,실행,검사,통제활동을 이용한 소프트웨어 품질개선패러다임이며, TQM은 벤치마킹을 통한 전사적 품질관리 방안이다

사용자 요구사항의 정형화는 개발과정상의 품질평가의 기본 요소이므로 중요하다. 따라서 소프트웨어 품질관리방법으로서 사용자의 정형적 명세화

방법과 이를 위한 도구의 개발은 중시되어야 한다. 이에 관한 품질 관리적 차원에서의 여러 방법들은 개발목표나 가용 자원 등을 고려해 적절한 기준에 따라 채택여부가 판단되어야 한다.

4) 소프트웨어 품질평가 가중치 결정방법

⊙ 품질평가와 객관적 가중치 결정

▪ 품질평가의 객관화

인간의 합리적이고 객관적인 의사결정을 위해서는 자료의 불명확성뿐만 아니라 가치판단 및 과정상의 주관성이 제거되어야 한다.

특히 소프트웨어 품질평가와 제품선정을 위해서는 소프트웨어의 복잡한 특성 때문에 일반화된 방법을 찾기는 쉽지 않다. 또한 품질특성들이나 혹은 대안들 간의 우선순위 혹은 상대적 가중치의 결정은 직접적으로 평가와 선정의 최종결정에 영향을 줄 수 있으므로 선택에 신중을 기해야 한다. 현재 객관적 자료 및 경험과 소프트웨어 공학적 방법을 통한 많은 평가의 객관화에 관한 연구가 진행 중이다. 특히 안정성이 우선되는 소프트웨어 제품, 예를 들어 원자력 발전소 관리 분야의 품질평가와 인수문제는 주관적 평가의 신뢰성문제와 관련되어 더욱 복잡하다. 이러한 주관적 평가의 객관화 문제를 해결하기 위해서는 가급적 가치판단을 위한 정량적 방법의 활용과 결과 제시가 요구된다. 또한 실제 자료에 근거한 시험 및 평가와 합리적이고 직접적인 적용이 가능한 이론의 개발과 활용이 이뤄져 보다 일관성 있는 품질평가와 결과의 검증이 가능해야 한다.

▪ 객관적 가중치 결정방법의 요구

기차판단과 평가에서 가중치의 문제는 다양한 가중치 결정방법에 따른 평점화 결과의 차이와 각 방법의 주관적 적용으로 인한 특징에서 기인한다. 지나치게 단순화한 가중치와 평점의 결정은 오히려 잘못된 고객만족과

품질평가 결과를 도출하게 된다 따라서 평가대상이나 평가자의 특징에 따른 다양한 가중치 결정방법의 채택이 중요하다. 소프트웨어 품질표준의 적용을 통한 품질평가와 가중치결정에 관한 연구로는 정호원과 이종무의 품질특성의 선정이나 소프트웨어 제품 전정모형에의 응용연구와 jung 과 Yoon의 품질평가와 자원배분 모형연구 이단형 외 3인의 품질관리에의 실제 적용사례, 그리고 기타 응용 연구가 있다.

⊙ 객관적 가중치 결정방법

▪ 다중 목표 의사결정과 소프트웨어 제품평가 방법

다중 목표 혹은 다속성의 최적 안 선정에 관한 연구는 최근 의사결정 분야에서 많은 진척을 이루어 왔다. 특히 시설이나 입지 혹은 자원 평가와 선정에 관한 연구는 경제적, 기술적, 환경적 그리고 사회적 요소 등을 고려한 다중의사결정의 대표적인 예로 꼽을 수 있다. 다중 목표 의사결정의 방법으로는 가중치법, 연속제거법, 수학적 프로그래밍법, 기타 공간적 근접법으로 구분할 수 있다. 이 가운데 다속성 의사결정 분야에서 흔히 사용되는 의사결정 규칙은 가중치 합계 혹은 선형모형 방법이다. 이는 각 요소의 가치 평가값에 중요도 혹은 가중치를 선형적으로 곱하여 전체 가치를 평가하는 방법으로 사용이 용이한 장점을 갖는다. 가중치 결정을 중심으로 다중 목표 의사결정의 다양한 방법들을 저리해 보면 소프트웨어 제품 평가와 선정 문제와 관련된 연구에서는 순위법과 AHP방법을 이용하여 소프트웨어 제품평가 및 선정에 관한 모형들의 특성별 구분을 통한 합리적인 평가 및 선정모형의 활용방안을 제시하고 있다.

▪ CASE 도구평가 및 선정에 관한 표준의 평가방법

<표60> CASE 도구 평가 및 선정방법

평가 및 선정 방법	주요 특징
비용 위주의 평가법	경제적 기준에 따른 평가
점수 위주의 평가법	정량화된 평가값의 부여
Borda방법	순위위주의 평가법 예)순위합계법
특성별 평가법	특성별 우선순위에 따른 평가
Condorcet 방법	이원비교에 의한 평가
Fishburn 방법	선호정렬법
Dodgson 방법	선호측정법
나열식 평가법	기준비교 평가방법
구조적 방법	계층적 분석방법 예)AHP

Case도구 평가 및 선정에 관한 표준지침에서 제시한 방법 외에 객관적인 가중치 결정방법에 관한 연구로는 다목표 의사결정에서의 가중치 방법의 비교에 관한 Hobbs의 연구를 참고할 수 있다. 이 연구는 가중치 결정방법의 중요성을 강조한 것으로, CASE 도구 평가 및 선정에 관한 표준지침 및 기타의 평가방법들을 포함하는 비교적 광범위한 구분이 가능하다. 따라서 이를 기초로 소프트웨어 품질평가와 선정에 관한 연구들의 가중치 결정방법의 주요 내용을 정리하면 위 표와 같다.

▪ 소프트웨어 품질평가를 위한 객관적 가중치 결정방법

<표61> 소프트웨어 품질평가를 위한 가중치 결정방법

구분	연 구	주요특징과 예
순위 혹은 범주 부여법	Edwards와 Newman연구 Sanders외 2인 연구 이종무와 정호원 연구	.평가자의 주관적 중요도에 따른 순위 혹은 범주별 구분과 상용의 용이함 특징 .가중치 부여법, 순위합계법, 순위역수법
평점 부여법	Brownstein과 Lerner연구 이단형 외 3인 연구	.정확한 가중치 결정가능, 직접 평점 부여(예, 1-10점 부여) .평가자의 명확한 전문지식 요구
이원비교 비율 질문법	Saaty연구, Zahedi 연구 이종무와 정호원 연구	구조적 방법, 1-9척도 선호, 상대적 이원비교의 가능함 견제 AHP 방법
비율 비교 질문법	Hobbs 연구 Comrey 연구	100만점 비율 질문 Metfessel Allocation방법
무차별 상쇄법	Edwards와 Newman연구 Hobbs 연구	효용함수적 무차별 상쇄이론 근거, 현실의 제한적 가정, 주관적 판단 일관성 가정, 효용가치에 의한 가중치 결정
선호법	Wong과 Lingras 연구 정호원과 이종무 연구 위금숙과 이금석 연구	.선호의 순위이용, 순위부여법의 특성활용, 주관적 강약의선호표현 .정성적 선호법
기타 차별 비교법	Fritz와 Carter연구 King과 Schrem연구	비용우위의 경제적 기준 비교 사전식 나열 비교 평가 및 선정 비용우위법, 사전편찬식 방법

⊙ 정량적 가중치 방법의 특징

▪ 순위법

가중치의 결정의 가치판단의 기본이다. 모든 속성이 동일한 중요도를

가질 수 없으므로 상대적 중요도를 가중치화 하는 것은 중요하다. 중요도의 우선 순위를 부여하여 이를 정량적 가중치로 활용하는 순위법으로 인간 가치판단의 역사상 매우 익숙한 방법의 하나이며, 사용상의 용이한 특징 때문에 다른 방법에 비해 보편적으로 활용될 수 있다.

- 주관적부여법

평가자의 주관적 판단에 따른 임의적 가중치를 직접 부여하는 방법

예를 들어 10점을 기준으로 순위에 따라 최하위 순위의 가중치까지 직접 부여한다. 그러나 이 방법은 순위를 우선 고려한다는 차원에서 순위법의 유형일 뿐이고, 단순평점부여법에 의한 직접적 가중치 부여방법과 동일하다.

- 순위합계법

가장 중요한 속성의 순위값을 가장 높게 부여한다. 이는 소위 역순위를 부여하는 방법으로서 가장 중요치 않은 속성의 순위값은 1이다. 동일순위의 경우에는 동일 순위 값들의 평균값을 동일 순위값으로 부여한다.

- 순위역수법

순위합계법과는 달리 가장 중요한 속성의 순위값을 1부터 부여한다. 그리고 모든 속성의 각 순위값의 역수를 계산한 후 정규화 과정을 통하여 가중치를 결정한다.

▪ AHP

다속성 의사결정의 상대적 중요도 결정을 위한 AHP는 해결해야 할 대상 문제가 복잡하고 비구조적인 경우에 합리적 가중치 결정의 한 방법으로 활용된다. 이 방법은 문제가 복잡하고 비구조적인

경우에 문제를 계층적으로 단계화하고 각 단계별로 중요 의사결정 요인들을 선정한다. 그리고 각 요인들의 상대적 중요도를 이원비교를 통하여 일관성 있게 가중치로 제시할 수 있다. 또한 계층적 분석을 통하여 각 계층의 합리적인 가중치 배분이 가능하다.

▪ 정상적 선호 방법의 특징

Wong과 Lingras의 정성적 신호이론은 신뢰함수에 입각하여 주관적 신호를 정량화된 수치로 객관화 할 수 있는 특징을 갖고 있다. 즉 전문 평가자로부터 주관적 선호관계만을 입력받아 이를 상대적인 가중치의 개념으로 객관화하여 정량적 평가가 가능케 할 수 있다. 대부분의 가치판단이나 평가는 사실상 주관적인 선호에 바탕을 두는 경우가 대다수이므로 이러한 정성적 선호법에 의한 가중치 결정방법은 체계적인 확장에 따라 보다 활용가치가 매우 높다.

- 신뢰함수

일반적인 확률적 방법이론으로는 표본 도수에 의해 실제 확률을 추정하는 객관적 도수 해석방법과 주관적 신뢰도에 의한 베이지안 접근방법이 있다. 객관적 도수 해석방법 : 객관적 도수 자료의 이용이 가능한 일반적인 의사결정이 주관적 신뢰도에 의한 베이지안 접근방법 : 전문가의 신뢰도를 의사결정의 바탕으로 비교적 간단한 의사결정에 주로 이용

- 정상적 선호의 표현

평가자가 주관적 선호를 수량화하여 표현하는데 익숙치 않은 경우 이를 확률값으로 나타내기는 더욱 어렵다. 이 때 증거이론의 신뢰함수를 이용하여 한 요소가 다른 요소보다 더 중요하거나 혹은 선호됨을 쉽게 표현할 수 있다. 정성적 선호이론에서는 선호관계의 표현(이하 선호명제)를 위해 이진적 관계기호 >와 무차별 관계기호 ~를 사용한다. 예를 들어 선호명제가 A>B로 표시되면 이는 평가

자가 문제에 대한 가능한 해 가운데 A가 B에 비하여 정답일 것이라고 믿는 정도가 크다 혹은 더 선호한다는 의미이다. 또한 상호관계가 동일하게 정답으로 믿어지는 즉, 동일한 선호명제의 경우에는 A~B로 표시한다.

5) 소프트웨어 투입요소 결정 모형

⊙ 체계적 구분의 필요성과 특징

▪ 체계적 구분의 필요성

정보시스템의 발전은 컴퓨터 하드웨어와 소프트웨어의 발전을 가속시켜 왔다. 특히 소프트웨어 분야에서의 발전은 응용분야나 사용목적 등에 한정되지 않고 광범위하고 다양하게 이뤄져 왔다. 이러한 소프트웨어의 복잡한 발전과 제품 특징에 따른 다양한 요구사항은 정보시스템의 품질관리에 중요한 요소로 작용하여 초기 개발단계에서부터 시험, 검사, 이도 후 실제 사용현장에서도 정확한 실행을 보장할 필요성이 증대하게 되었다. 그리고 소프트웨어 제품 개발 생명주기상의 구현 이후의 유지보수 단계에서도 보다 장기적인 소프트웨어 응용의 성공 가능성을 확신할 수 있는 요구가 또한 증대하고 있다. 따라서 이러한 요구는 첫째, 소프트웨어 제품에 관한 요구사항의 체계적인 문서화와 또한 요구사항의 수용여부에 관한 단계적인 확인을 통해 구체적으로 해결해야만 한다. 둘째, 해당 소프트웨어 품질을 올바르게 설명할 수 있는 투입요소의 선정과 합리적인 품질측정과 검사를 통한 객관적인 품질 평가가 이뤄져야 한다. 따라서 사용자 요구사항의 올바른 파악과 이의 체계적인 문서화는 중요하며, 이를 바탕으로 소프트웨어 제품의 품질을 측정 평가하는 것이 필요하다.

▪ 품질평가 방법상의 특징

– 3차원 품질평가 방법의 특징

Sanders의 3차원 평가요소을 바탕으로 품질평가를 위한 중요 관점들을 살펴보면 다음과 같은 3가지 관점의 평가요소의 구분이 가능하다.

* 시스템의 호환성 여부를 확인하는 결정적 요소가 있다. 세부적으로는 하드웨어 시스템의 호환성과 영상 호환성과 같은 특정 응용영역의 호환성 여부를 확인하는 요소 등이 있다

* 정량화할 수 있는 경제적 측면의 객관적 평가요소를 들수 있다. 예로는 초기 구입비용, 운용과 유지보수 및 변경비용, 그리고 사용 운영자의 교육훈련 비용 등의 제비용적 요소가 있다.

* 정보시스템 활용에 따른 정보의 질적인 차원의 주관적 평가요소를 들 수 있다. 이는 구체적으로 문서화의 품질,사용의 용이성, 정보제공의 품질 등 기타 세부 평가요소들을 나눠 살펴볼 수 있다.

기본적으로 평가대상의 호환여부를 우선 확인하기 위해 결정적 평가요소를 적용한다.

– 소프트웨어 개발과정 관리에 관한 접근방법

소프트웨어 제품의 특성 때문에 개발과정의 품질관리를 위해서는 개발단계별 혹은 관리과정별 접근방법이 요구된다.

소프트웨어 개발과정 관리에 대하여 Doucek 의 개발과정의 관리방법에 관한 연구을 적용하면, 일반적으로 프로젝트 관리방법과 활동과정 관리방법의 두 가지 방법의 적용이 가능하다. 이 가운데 활동과정별 품질관리방법의 내용은 개발 각 단계의 활동별로 적합한

관리활동이 요구된다는 차원에서 3가지 유형의 개발활동을 품질관리의 주요대상으로 삼는다. 즉 분석결과 설계된 내용이 구현된 완료활동과 분석 설계과정은 확정되었으나 완성되지 않은 계획활동 등으로 구분된 관리과정별 활동을 대상으로 품질관리를 행할 수 있다. 그러나 이는 관리과정별 소프트웨어의 품질관리가 동시 병행적으로 이뤄져야 하므로 실제적 적용에는 상당히 복잡한 문제가 대두될 수 있다

- 소프트웨어 품질공학적 개방과정과 응용영역에 따른 방법

일반적으로 소프트웨어 품질공학 측면의 개발과정 모형의 내용은 개발단계의 분석과 설계, 구현의 특정에 따라 기본모형과 구현모형으로 나눠볼 수 있다. 기본모형은 분석과정의 산출과 요구사항 명세서 작성을 위한 기본적 모형으로서 정보적, 환경적, 조직행동적 관점의 3관점으로 구성된다. 구형모형은 설계과정의 산출, 구조도 등의 모듈명세 형태의 도구 활용등의 내용을 포함한다. 이를 활용한 소프트웨어 개발연구로는 안전성을 고려한 소프트웨어 개발과정과 인간-기계 접속 시스템의 구현 연구을 들 수 있다. 소프트웨어 제품의 개발과정과 아울러 사용자의 요구사항의 차이에 의해서도 상이한 품질특성의 요구가 나타난다. 특히 구체적인 소프트웨어 응용분야별 특징은 품질특성에 영향을 미치게 된다. 따라서 사용자 요구사항과 응용분야별 특징은 체계적으로 구분되어야 한다. Lac과 Raffy의 연구를 참고해 보면, 실증적 사례자료로서 응용분야별 소프트웨어 품질특성의 상대적인 중요도를 구분 정리할 수 있다.

⊙ 제품 품질평가를 위한 지침

▪ 제품 품질 평가를 위한 지침의 구성

본 연구에서는 소프트웨어 품질평가 과정에 관한 지침과 ISO 9126의 표준 정의를 인용하여 사용한다. ISO 14598 시리즈는 소프트웨어 제품의 품질측정과 심사 및 평가를 위한 방법을 제시한다. 그러나 이러한 방법들은 소프트웨어 생산과정 혹은 비용예측등을 위한 목적으로 사용될 수 있지만, 이를 위한 직접적인 평가방법에 관한 내용은 제시되지 않고 있다. ISO 14598시리즈는 ISO9126 시리즈에 정의 및 예시한 품질모형, 품질특성 및 척도등을 바탕으로 소프트웨어 품질평가과정의 개관과 평가지침 및 요구사항들을 제공한다. 14598-2와 14598-6은 기업 혹은 전체 부서 수준의 평가 경영 및 지원의 내용을 기술한다. 또한 14598-3,14598-4,14598-6는 각각 세부 프로젝트 수준에서의 평가 요구사항 및 지침에 관한 내용을 제공한다.

▪ 제품 품질평가를 위한 지침의 주요 내용

- 품질평가 지원:ISO 14598-2,14598-6

14598-2 : 소프트웨어 품질평가를 위한 계획 및 경영의 중요 내용 기술

14598-6 : 품질평가 모듈의 문서화를 위한 지침 제공

- 품질평가 과정 : ISO 14598-3,14598-4,14598-5

14598-3 : 조직내의 인적 기술적 자원을 동원하여 소프트웨어 제품을 새로이 개발하거나 기존 제품을 향상시키기 위한 계획을 가지고 품질 평가를 하려는 경우에 활용

14598-4 : 이미 개발 완료된 기성 소프트웨어 제품의 구매획득이나 기존 제품의 재사용을 계획하는 경우에 적용

14598-5 : 소프트웨어 제품의 객관적이고 독립적인 품질평가를 위해, 외부의 제3자에 의한 평가를 행하는 경우에 해당

- 평가과정의 가치사슬의 연결

ISO 14598 시리즈에서 제시한 평가 대상자별 제 관점의 상이함은 관점의 특징, 즉 요구사항, 관심정도 혹은 가용 투입요소 등에 따라 품질특성상의 중요도가 상이하게 나타난다. 예를 들어 개발자의 경우 조직내부의 가용 투입요소와 개발과정중의 중간제품을 중심으로 품질평가를 하는 반면, 구매자는 기존 제품 혹은 기성의 제품을 중심으로 사용자의 응용영역이나 공급자에 의해 제시된 투입요소를 통해 간접적으로 개발된 품질을 평가해야 하기 때문이다. 따라서 개발자의 관심과 구매자의 관심은 서로 다르며, 또한 제 3 평가자의 관심과 이해는 더욱 상이할 것이다. 그러나 이러한 상이한 평가관점별 투입요소와 품질에 관한 중요도는 가치사슬상의 기업조직의 가치극대화를 위해서 체계적인 구분이 이뤄져야 한다. 또한 이러한 구분과 관점별 중요도의 차이는 품질평가의 객관성 및 평가결과의 타당성을 높여줄 수 있을 것이다.

6) 소프트웨어 개발 생명주기와 가치시스템

⊙ 소프트웨어 개발 생명주기에 관한 표준상의 개발과정

▪ 기본과정

주요 기본과정은 소프트웨어 생명주기상의 주요당사자(이하 사용자)들과 관련되는 직접적인 5개 세부공정단계로 구성된다. 여기서 사용자들이란 시스템이나 소프트웨어 제품의 획득자 뿐만 아니라 소프트웨어 공급자, 개발자, 운영자, 그리고 유지보수자들이 포함된다. 세부공정단계로는 획득공정 공급공정, 개발공정, 운영공정, 그리고 유지보수공정 등이 해당된다.

<표62> 기본과정의 세부 활동

공정활동	세부활동
획적공정	착수,제안요청서 준비, 계약준비 및 수정, 공급자 감시,수락 및 완료
공급공정	착수,응답준비,계약,계획수립,실행 및 관리,검토 및 평가, 인도 및 완료
개발공정	개발공정구현, 시스템요구분석,시스템구저설계,S/W 요구분석, S/W 구조설계, S/W 상세설계, S/W 코딩 및 시험, S/W 통합, S/W 자격시험,시스템통합, 시스템자격시험, S/W 설치, S/W 수락지원
운영공정	운영공정구현, 운영시험, 시스템운영, 사용자 지원
유지보수 공정	유지보수 구현, 문제 및 수정분석, 수정구현, 유지보수 검토/수락, 전환, S/W 폐지

▪ 지원과정

지원과정의 8개의 지원 공정으로 구성되며, 기본과정이나 다른 공정에 의해서 사용될 수 있다. 따라서 지원과정은 다양한 생명주기에서 이용되며, 조직이나 응용목적, 프로젝트와 고객의 특성 등에 따라 다양하게 수행될 수 있다.

<표63> 지원과정의 세부 활동

문서화공정	문서화공정구현,설계 및 개발, 생산, 유지보수
형상관리공정	형상관리공정구현, 형상식별,형상통제,형상상태기록관리,형상평가,공표관리와 인도
품질보증공정	품질보증공정구현, 제품보증,공정보증,품질시스템보증
검증공정	검증공정구현,검증
확인공정	확인공정구현,확인
합동검토공정	합동검토공정구현,프로젝트관리검토,기술적검토
감사공정	감사공정구현,감사
문제해결공정	문제해결공정구현,문제해결

▪ 조직과정

조직과정은 관리, 요원 교육훈련, 또는 공정개선등과 같은 조직적 기능의 수행을 위해 조직에서 활용된다. 이들 공정 활동들은 보통 전사적 수준에서 수행되며 특정프로젝트나 계약의 범위에 국한되지 않는다.

<표64> 조직과정의 세부 활동

경영관리 공정	착수 및 범위정의, 계획수립,실행 및 통제, 검토 및 평가, 마감
기반구조공정	기반구조공정구현,기반구조의 설정, 기반구조의 유지보수
개선공정	개선공정설정, 공정평가,공정개선
교육훈련공정	교육훈련공정구현, 교육훈련 교재개발, 교육훈련 계혹 구현

⊙ 소프트웨어 개발주기 공정 관점과 사용자 연결

소프트웨어 개발생명주기 과정에 관한 지침상의 관점들은 기본적으로 계약관점, 공학적 개발관점, 운영관점, 관리관점, 그리고 지원관점등으로 나눠볼 수 있다. 소프트웨어 사용자들은 대체로 관련되는 가치사슬를 고려해 구분해 보면, 계약 및 구매과정의 경우에는 주로 구매자 혹은 공급자들이 관련된다. 또한 공학적 기술이 중요시 되는 개발과정에서는 개발자와 유지보수자, 운영과정에서는 운영자, 그리고 최종 관리과정에서는 전사적인 관리자 혹은 해당 부서 관리자들이 관련된다.

⊙ 가치사슬과의 연결

기업사슬상의 기업가치 극대화를 위한 소프트웨어의 품질평가는 개발 생명주기상의 관련성과 기업활동 과정상의 중요도 등을 고려해야 한다. 즉 정보시스템을 활용하는 기업의 경영가치 극대화는 결과 사용자의 요구사항과 관점별 중요도에 따라 구매 혹은 개발 환경에 맞춰 품질관리가 이뤄져야 한다. 즉 사용자의 유형에 따라 개발 생명주기상의 관점의 관련성이 달라지고, 또한 이에 대한 품질 요구사항의 내용이 상이하며 품질특성의 중요도 역시 달라진다. 가치시스템은 공급자의 가치사슬과 유통경로상의 가치사슬은 기업의 직접적인 가치창조에 이바지한다. 본 연구에서는 이를 소프트웨어 개발 생명주기 과정에 관한 표준상의 관점별 유형으로 정의해 투입요소 결정에 활용한다. 예를 들어 요구분석 단계에서 품질 중요도는 구매자 혹은 공급자, 설계 및 구현단계에서의 중요도는 개발자나 유지 보수자의 이해 관계를 고려해서 품질평가 투입요소의 우선순위를 결정한다. 운영/공학적 관점에서는 개발과정주기 단계별 품질특성 중요도를, 계약 관점에서는 고객차원의 사용품질을, 그리고 관리 관점에서는 전 개발 과정에 걸쳐 전반적인 품질 중요도를 고려하여 최종 결과를 도출한다.

7) 소프트웨어 품질평가의 제 관점

⊙ 개발 과정을 고려한 평가 관점

소프트웨어 개발 생명주기상의 각 단계별 특성을 고려한 구분은 소프트웨어 제품의 특성상 당연히 필요하다. 우선 주관적 평가요소와 객관적 평가요소가 개발 과정상에서 분석되어야 한다. 이에 참고할 수 있는 연구 가운데 품질평가와 예산추정요소의 관련성을 개발단계별로 분석한 평가요소 연구가 있다. 이는 구체적으로 경제적 관점에서 평가요소 상호간의 중요도를 연구하면서 제품 품질평가의 객관적 기준으로 예산 추정요소들을 프로젝트 관점, 조직

관점, 인적 자원 관점에서 제시하고 있다. 또한 Huff의 연구에서 제시한 주요 비용관련 요소들은 프로젝트관점(환경복잡도,현공학기술,기술성장도,심사), 조직관련(대상조직규모, 조직문화변화, 현재관례), 인적자원관점(적용범위, 현 기술수준, 구현기술)등이다.

⊙ 소프트웨어 제품 품질평가 관점

▪ 직관적 소프트웨어 평가요소의 필요성

소프트웨어 제품의 선정요소 가운데 기업조직 차원의 관심은 전사적인 품질관리와 평가의 측면에서 궁극적으로 결과에 많은 영향을 미친다. 예를 들어 소프트웨어를 개발하거나 구매하려는 기업조직의 규모에 따라 선정요소와 정책 자체가 상이하다. 특히 중소기업 조직의 소프트웨어 선정요소와 대기업 조직의 선정요소 및 정책에는 많은 차이가 있고, 평가와 선정결과에도 차이가 크다. 성공적인 소프트웨어 제품선정의 기본 요소로는 이러한 기업조직의 구조와 규모, 기업정책의 정형화 정도, 그리고 공식적 소프트웨어 선정 정책 등을 들 수 있다. 따라서 이러한 조직관점의 평가와 선정을 위한 구분은 중요하며, 종합적인 품질평가의 틀 속에 포함되어 있다.

8) 결론 및 향후 연구방향

⊙ 연구결과의 요약 및 의의

첫째, 본 연구에서는 기업의 가치 극대화 차원에서 대외 경쟁력강화를 위한 가치기반 소프트웨어 품질의 중요성을 강조하였다. 이는 본 연구의 기본사항이며 소프트웨어 활용과 품질평가의 기본 전제라 할 수 있다.

둘째, 표준 품질모형과 품질특성의 내용을 기존 주요 연구 내

용과 함께 설명하고, 표준 품질특성의 구체적인 활용방법을 제시하였다. 특히 계층적 품질모형의 특징을 이용하여 개발자 관점의 내부품질특성과 사용자 관점의 외부품질특성을 객관적으로 연결할 수 있는 방법을 개발하고, 이를 산업현장에 적용한 결과를 예시하였다. 이 결과 전체 40개 내부품질특성 가운데 주요 28개 특성을 통하여 상위90%의 우선순위를 모두 포함시킬 수 있었다. 이는 예를 들어 시간과 예산상의 제약 하에서 상위 28개 내부품질특성을 통하여 전체 외부품질특성의 90%이상을 간접 측정해 낼 수 있음을 의미한다.

셋째, 합리적인 소프트웨어 품질평가에 적합한 가중치 결정방법을 제시하였다. 소프트웨어 품질평가 뿐만 아니라, 인간의 모든 의사결정이나 가치판단 과정에서 적용된 가중치방법이 결과에 미치는 영향은 매우 크다. 본 연구에서는 주관적인 평가값을 합리적으로 정량화할 수 있으며, 사용의 용이성과 합리성, 그리고 범용성 등의 특징을 갖는 순위법, AHP방법, 그리고 정성적 선호법등의 세 가지 가중치 결정방법을 제시하였다. 그리고 각 가중치방법을 표준 품질특성의 평가에 적용시켜 그 결과를 제시하였다. 특히 평가자의 주관적 선호만으로 정량적 가중치 결정이 가능한 정성적 선호법을 '계층적 선호'관계의 선호법으로 확장하여 제안하였다. 이는 인간의 주관적 의사결정의 모든 분야에 두루 적용이 가능한 매우 유용한 가중치 결정방법의 하나이다.

본 연구의 품질특성 선정 결과를 종합 비교해 보면, 세 가지 가중치방법 모두 평가결과를 객관적으로 정량화하기에 적합하였다. 구체적인 우선순위 값의 차이는 있으나, 최종 결과는 모두 일치하였다. 이는 가중치 결정방법상의 특징에서 비롯되는 것으로 이해된다. 정량화 결과와 관련해서는 정성적 선호법은 비교적 안정적인 가중치 결정에 적합하며, 평가자의 정확한 정량적 이원비

교가 가능한 경우에는 AHP방법이 보다 바람직하다고 판단된다.

넷째, 소프트웨어 품질평가에 필요한 평가 관점들을 체계적으로 구분하고, 이를 종합하여 투입요소 결정을 위한 종합모형을 제안하였다. 소프트웨어 품질의 올바른 평가를 위해서는 예를 들면 조직의 목표, 고객의 요구사항, 사용품질, 그리고 대상제품의 무결성 수준 등에 따라 품질특성의 상대적 중요도가 각기 다르게 고려되어야 한다. 이를 근거로 본 연구에서는 평가관점들에 관한 기존의 구분연구들을 분석하고 체계화하여 세 가지 평가관점 즉, 조직, 고객, 그리고 프로젝트 관점을 도출하였다.

또한 조직 관점을 구성하는 세부 차원으로는 제 관점간의 중요도와 비용-품질효과간의 중요도 결정과 품질관리 회의 등의 활용결정을 위한, 소프트웨어 전략과 관리정책 등의 두 가지 차원을 제안하였다. 고객 관점의 세부 차원으로는 사용자 차원과 평가자 차원을, 그리고 프로젝트 관점에서는 문제영역 ,제품, 그리고 외부환경등의 세 가지 세부 차원을 도출 제안하였다.

다섯째, 이상의 연구 결과 도출된 평가 관점과 해당 차원별로 개발 생명주기별 특징을 고려하여, 품질특성에 관한 전략적 중요도를 고안하여 제시하였다. 그리고 이를 일반 상용소프트웨를 구매하고나 혹은 기존의 다른 소프트웨어의 구성요소를 재 사용하는 경우에, 결정모형에 적용시킨 결과를 가중치 방법별로 비교 제시하였다.

최종 제시된 투입요소간의 우선순위는 가중치 결정방법에 관계없이 동일한 결과를 보였다. 그러나 투입요소의 절대적 수치는 가중치 결정방법에 따라 상이한 결과로 나타났다.

여섯째, 조직 관점에서의 비용-품질효과간의 상이한 중요도에서 기인하는 투입요소 우선순위의 변화를 민감도분석을 통하여 제

시하였다.

본 연구의 경우 조직 관점의 전략목표에 따라 가중치 방법별로 민감도분석의 결과에 분명한 차이가 나타났다. 비용우위의 소프트웨어 전략에서는 차이가 없으나, 품질우위의 전략의 경우에는 가중치 결정방법에 따라 투입요소의 최종 우선순위에 차이가 나타났다. 이는 조직관점의 소프트웨어 전략 혹은 관리정책의 상대적 우위 정도에 따라서 상이한 우선순위의 결정이 가능함을 의미한다. 이상의 결과를 종합해볼 때 본 연구의 의의로는 개발과정을 고려하고 가치 프로세서 차원의 품질 추구를 통한 기업의 경쟁우위 달성을 강조하였다. 이를 위한 구체적인 방안으로서 개발과정상의 품질관리와 품질평가나 외부의 품질감사등에 대비한 투입요소의 결정을 객관화하여 제시하였다. 또한 표준 품질모형과 특성을 중심으로 개발단계별 품질관리와 표준적 품질평가 지침등을 고려하여, 투입요소 우선순위 결정을 위한 평가관점들을 종합적으로 체계화하였다. 이러한 소프트웨어 품질평가의 종합적 체계의 적용을 통하여 품질평가의 객관적 신뢰성 를 높일 수 있고, 세부적 평가단계의 결정을 보다 객관화할 수 있을 것이다. 특히 기업의 경쟁력 차원에서 가치기반 소프트웨어의 품질확보는 필수적인 요소인데, 본 연구에서 제안한 내부 품질과 외부품질에 관련성에 따른 객관적 연결방법은 개발과정에서의 사용자 관점의 외부품질을 예측해 볼 수 있는 방법으로서, 향후 실제 산업현장에서 그 활용 가능성이 매우 크다고 판단된다.

⊙ 연구의 한계 및 향후 연구방향

본 연구의 한계점과 일반적인 확대를 위한 향후 연구방향으로는 다음 내용들을 들 수 있다.

첫째, 본 연구에서 제안된 소프트웨어 개발자 관점의 내부품질을 통한 외부품질의 간접적 평가방법은 종합적 관점에 따라 다양해 질 수 있다. 즉, 예시한 소프트웨어 품질특성과 활용에 관한 표준의 내부품질과 외부품질의 관련성은 평가대상이나 평가 환경에 따라서 달라질 가능성이 크며, 품질특성 상호간의 상충성도 관련 중요도에 영향을 미칠 수 있다.

둘째, 선정 제시된 세 가지 가중치 결정방법 외에도 소프트웨어 품질평가자의 중요도나 선호표현의 특징에 따라서는 또 다른 최상의 가중치 결정방법이 활용될 수도 있을 것이다. 현실적으로 대부분의 가치판단이나 의사결정의 경우처럼 소프트웨어의 품질평가에서도 가중치 결정방법이 최종결과에 미치는 영향은 매우 크므로, 구체적인 품질평가와 선정에 앞서 보다 합리적인 가중치 결정방법의 신중한 선택이 요구된다.

셋째, 도출 제시된 품질특성에 관한 상대적 중요도의 객관적 타당성의 확인은 산업현장의 실제 적용 자료를 분석 검증해야 하는 한계점이 있다. 본 연구에서는 기존 외국의 산업현장에서의 주요 사용사례 연구를 중심으로 중요도를 도출하였으므로, 우리나라 현실에 적합한 품질특성 중요도의 도출과 활용이 요구된다.

넷째, 종합적 투입요소 결정모형의 타당성 역시 향후 산업현장의 적용을 통한 확인 및 검증이 필요하다. 또한 결정모형의 제 관점이 구분과 관련해서 가치기반 소프트웨어 품질의 보다 현실적인 추구를 위해서는 다양하고 구체적인 가치사슬상의 고객 및 중간 유통업자들의 관계를 고려하고, 보다 세부적인 관점 구분과 이에 따른 품질 중요도를 해당 관계자들로부터 도출해 활용할 수 있어야 한다.

다섯째, 투입요소 결정모형의 결과에 관한 일반성 확보를 위해

서는 실제 산업현장의 복잡하고 다양한 자료를 적용해야 하며, 이 경우 보다 다양한 결론의 도출이 가능하리라 판단된다. 또한 본 연구에서는 보다 더 세부적인 투입요소의 구분에 따른 우선순위의 제시를 못한 한계점이 있다.

요약하면 본 연구는 향후 연구결과의 확대와 관련해서 산업현장의 실제적용과 이에 따른 결과 분석이 뒤따라야 한다. 특히 개발과정의 품질관리 차원에서 개발자 관점의 공학적 품질특성과 관련되는 외부품질특성의 관계 파악은 매우 중요한데, 이러한 상호관련성 및 사웅성의 관계 정립을 위한 산업현장의 객관적 자료가 필요하다. 향후 이러한 산업현장이 자료수집에 적절한 내부품질척도의 개발과 적용이 가능하게 되면, 이는 품질관리차원에서의 현실적 가치뿐만 아니라 미래의 품질예측력 향상에 기여할 수 있을 것이다.

그리고 본 연구의 관리 정책적 차원의 품질평가 관점 외에도, 예를 들어 조직의 정보시스템 성장 단계별 비용-품질효과 중요도의 차이나 혹은 조직 관리정책 차원의 관점 간 중요도, 그리고 프로젝트 각 차원간의 중요도의 차이 등을 민감도분석에 의해 보다 세부적으로 분석한 연구도 가능할 것이다. 또한 투입요소의 체계적 구분에서 제시한 세부적인 투입요소들의 적용이 포함되면, 품질관리나 품질감사의 실행과정에서 더욱 유용하게 활용될 수 있을 것이다.

1.1.2. 국외 사례

가. 소프트웨어 품질보증 [정통기03]

1) 해외 S/W품질 시험·인증 기관 현황

세계 S/W제품(패키지S/W) 매출액은 2002년 2천1백억 달러 규모에서 연평균 13.5%정도 성장하여 2006년 3천8백억 달러 규모에 달할 것으로 예상된다. 또한 S/W의 매출에 따르는 품질비용은 2002년 매출액 대비 3.7%인 80억 달러에서 2006년 5.2%인 200억 달러 정도로 예상되며(자료출처 : IDC 2002), 이는 매년 큰 폭의 품질개선비용 상승을 예고하고 있다. 미국의 경우 S/W결함이 경제에 미치는 사회적 손실 비용이 GDP의 0.6%인 연간 595억 달러로 추정(출처 : NIST, 'The Economic Impacts of Inadequate Infra-structure for Software Testing', 2002. 5.)되며, 이를 S/W테스트 및 인증과정을 거쳐 줄일 수 있는 비용 절감 효과는 손실 비용의 37%를 절감할 수 있다고 한다.

이에 세계 여러 나라에서는 S/W품질 시험·인증제도 및 기관을 두어 S/W품질을 높여 경쟁력을 강화해 나가고 있다.

⊙ DELTA

Danish Academy of Technical Science의 지원을 받는 독립적인 비영리 조직으로 1969년에 설립되었으며 Software Engineering, Quality and Certification을 포함한 6개 분야에서 270여명의 직원이 일을 하고 있다. 덴마크, 노르웨이, 스웨덴의 인정된8개의 시험소에서 20여 개국 이상을 대상으로 서비스를 수행하고 있으며, 1982년부터 영리적 목적의 시험 수행으로 IEEE 488.2의 적합성을 위한 통신 프로토콜 시험과 ISO/IEC 9126의 품질 특성을 기반으로 한 Critical 응용S/W의 인증 서비스를 수행하고 있다.

평가 기간은4~5주부터6개월까지 다양하며, 보통 평가 소요기간은 2~3달 정도이다. 비용은 시장 기능에 따라 결정되며, 일반적으로 S/W개발 비용의 5?10%정도이다. 평가기준은 Delta에서 개발한 S/W제품 평가 체계로 ISO/IEC9126 및 ISO/IEC 14598-6을 기반으로 하여 S/W문서, 개발 문서, 소스코드를 통한 제품 평가를 실시하고 있다.

<표 65> 세계 S/W제품(패키지S/W) 매출액

2002	2003	2004	2005	2006	연평균 성장률(%)
213.6	247.2	287.8	335.6	381.0	13.5%

자료 : IDC, The Worldwide Black book(Version 4), 2002

⊙ Iqqa

한국의 TTA, 미국의 Globalyst 및 JCCSoft와 제휴관계를 맺고 있는 회사로 S/W품질 보증(Quality Assurance), S/W보안, 인증 프로그램 및 마케팅서비스를 제공하고 있다. 미국내 3개(본사는 네바다), 유럽 및 아시아에4개의 지사를 두고 있으며, 주로 기능성, 호환성, 확장성 및 성능과 마이그레이션, Interna-tionalization, Localization 분야에서 품질보증테스팅 서비스를 제공하고 있다. 인증서비스로는 인증 로고 부여를 위한 버그 테스트, 콘텐트 및 업무 프로세스 확인, 기본 표준 인증, 정의된 환경 내에서 기능성 및 안정성 검사 등을 하고 있다.

인증서비스 외에 컨설팅 서비스로 여러 보안솔루션업체를 연결하여 보안시스템 구축을 위한 One-stop 서비스를 제공하고 있으며 Mercury Interactive, Segue, Rational, RSW등의 주요 툴을 통해

진행 중인 QA 프로세스를 객관적으로 평가하고 각 환경에 적합한 툴, 유틸리티 및 기술을 추천하는 등의 컨설팅서비스도 제공하고 있고, 특히 Localization이 필요한 회사에는 현장지원 서비스를 제공하고 있다.

그 외에도 소프트웨어 개발 프로젝트의 프로그래밍 언어,개발 플랫폼,개발 기술 등 프로젝트에 특화된 벤치마킹테스트 서비스를 제공하고 있다.

⊙ NTS/XXCAL

1961년 설립되어 1998년도에 XXCAL과 합병되었다. 미국 LA,Boston, Texas등에30여개의 시험소를 운영하고 있으며, 유럽 시장을 위해 독일의 Siemens사와 업무 협약을 체결하고 있다. 아시아 시장 공략을 위해 일본에 지사를 운영하고 있으며, 주요 사업 영역으로는 Software Testing, Network Testing, WirelessTesting, Hardware Testing, Quality Assurance, Engineering for Automotive and Aerospace, Nuclear Plant등이 있다.

S/W인증서비스로 Microsoft사의 Xbox compatibility test, Taiwan government EMC standard compliance, UL 1950/60950, Microsoft사의 Windows Logo program for Hardware, Verizon사의 Product certification, NTSCertification 프로그램, PDA(Personal Device Assistance)에 탑재되는 응용 프로그램 시험·인증서비스를 하고 있고, 부가적으로 Benchmarking Testing, Comprehensive Testing등도 하고 있다. 평가 항목은Functionality, Compatibility, Interoperability, Usability, Performance등이 있다.

⊙ eTesting Lab.

Ziff David Media의 한 Besiness Unit으로 1999년에 설립되었으며 2002년에 LionBridge로 흡수되었다. LionBridge의 시험. 인증 유닛인 VeriTest와 연계하여 Performance & Benchmarking Test, Usability Test, Web-based Test 서비스 등을 제공하고 있으며, 고객으로부터 의뢰 받은 벤치마크테스트의 결과를 PC Magazine에 발표하는 등의 서비스도 하고 있다.

⊙ NSTL

1983년 NIST(미국기술 표준원)에서 분리되어 설립된 테스팅 및 품질 보증 전문 기관이다.

미국, 캐나다, 영국, 대만, 일본, 중국과 인도등에 연구실을 운영하고 있으며, 주요 고객으로는 Microsoft, Intel, Qualcomm, Nokia,IBM, HP, Dell 등이 있다. NSTL은 1990년부터 캐나다 정부를 대행하여 IT 조달 서비스를 제공하고 있는데 캐나다 정부에 납품하기 위해서는 NSTL 테스트가 필수 사항이다. 평가 항목은 제품의 성능, 기능적합성, 사용성 평가 등을 수행하고 있으며, 테스팅 결과는 캐나다 조달 담당관리가 웹에 게시하는 형태를 취하고 있다.

나. 소프트웨어 벤치마크테스트(BMT) 현황 [정통기b05]

1) 해외 현황

S/W 벤치마크테스트 초창기에는 하드웨어 및 소프트웨어 개발 회사들에 의해서 트랜잭션 처리와 데이터베이스 시스템의 성능 측정을 위한 벤치마크테스트가 수행되었다. 벤치마크테스트 결과를 악용하여 자사의 강점은

부각시키고, 약점은 숨기는 형태로 벤치마크테스트 결과를 마케팅에 이용하여 Benchmarketing이라는 새로운 용어가 출현하기도 했다. 또한, 자사 제품의 벤치마크테스트 결과가 나쁘게 나왔을 경우 또 다른 벤치마크테스트를 통해 자사 제품의 결과가 좋게 나올 때까지 벤치마크테스트를 지속한다고 하여 벤치마크 전쟁(Benchmark war)이라는 용어도 나타났다. 이러한 객관성이 결여된 벤치마크테스트 결과들이 오용되고 있는 상황에서 업체가 아닌 일반 고객에게 정확하고 객관적인 정보를 제공한다는 목적으로 1973년에 미국 시카고에 Neal Nelson Associate가 설립되어 Unix-based 벤치마크를 독점적으로 수행하기 시작하였다. 그 이후 80년대 전반부터 S/W품질 시험.인증과 더불어 벤치마크테스트를 수행하는 제3자 시험 전문기관들이 등장하였고, 80년대 후반에 벤치마크테스트 결과의 오용과 혼란을 막기 위해 관련 업체들로 구성된 TPC, SPEC 등과 같은 컨소시엄이 구성되어 각종 벤치마크 모델을 정립함으로써 벤치마크 전쟁(Benchmark war)이 감소하게 되었다. 주요기관은 다음과 같다.

미국 NSTL(National Software Testing Labs,http://www.nstl.com)은 1983년에 NIST(미국기술표준원)에서 분리된 후, 각종 인증 프로그램과 벤치마크테스트를 수행하고 있으며, 1990년부터 캐나다 정부의 IT 조달 벤치마크 서비스를 대행하면서 방대한 지식베이스를 구축하게 되어 이 분야에서 독보적인 시험기관으로서의 위상을 확립하게 되었다. 캐나다 정부는 IT 조달벤치마크 서비스 시행을 통해, 제품 품질 확인 기간을 6개월에서 2주 이내로 단축할 수 있었고, 제품 구입 가격을 낮추고, 구입 제품의 품질까지 개선되는 효과를 얻을 수 있었다.

미국 VeriTest(http://www.veritest.com)는 세계적으로 알려진 사설 시험.인증 전문기관으로 2002년 7월에 세계 최고의 벤치마크테스트 기관이던 Ziff Davis의 eTesting Lab을 합병하여 테스트 부분을 확대해 나가고 있다. 세계 최고의 벤치마크테스트 도구 개발 회사로, 자체 개발한

Business Winstone, IBench,WinBench, 3D WinBench와 WebBench 등의 벤치마크테스트 도구를 이용하여 인터넷, 하드웨어와 소프트웨어 제품의 성능에 대해 전문 테스터와 실제 사용자에 의한 사용성, 제품 비교 테스트 등 벤치마트 테스트를 수행하고 있다. 완제품 뿐만 아니라 개발 중의 제품에 대해서도 테스트를 수행하고 있으며, 100여종 250여 제품의 벤치마크테스트 결과를 홈페이지에 게시하여 신청 업체들이 홍보 및 마케팅에 활용할 수 있도록 하고 있다. 또한 1992년부터 약 500여 회사 제품에 대해 S/W 분야별로 벤치마크테스트를 실시하여 결과를 PC Magazine에 정기적으로 공표하는 등 S/W 벤치마크테스트 분야를 선도하고 있다.

TPC(Transaction Processing Performance Council,http://www.tpc.org)는 트랜잭션 처리와 데이터베이스 분야에서 벤치마크를 정의하는 벤더들의 컨소시엄으로, 대형 transaction-based 시스템의 성능을 측정하기 위한 사실상(de facto) 표준 기관으로, IBM, Dell,SUN, Microsoft 등 23개 기관이 member로 참여하고, 4개 기관이 Associate member로 참여하고 있다.정부, 비영리기관, 교육기관, 컨설턴트 등은 Associate member로 참여가 가능하여, TTA S/W시험인증센터가 가입을 추진하고 있다. 연 100여 건의 벤치마크테스트를 수행하여 결과를 홈페이지에 게재하고 있고, 2달에 한번씩 Benchmark Status Report를 발행한다.

SPEC(System Performance Evaluation Cooperative,http://www.spec.org)는 1988년에 설립되어 60개 이상의 회원사를 가진 성능 표준화 기구로 성장하였다. 3개의 그룹으로 구성되어 있으며, 28종의 벤치마크테스트 모델을 개발하여 활용하고 있다.

미국은 벤치마크테스트 활성화를 통해 제품 경쟁력이 강화됨에 따라 S/W 최강국이 될 수 있었으며, 현재는 세계 시장을 지배하는 S/W 최다 생산국으로서 벤치마크테스트 서비스가 더욱 활발히 이루어지고 있다. 또

한 지난 20여년 동안의 벤치마크테스트를 통해 벤치마크테스트 방법론과 프로세스가 명확하게 확립되어 시험 기관의 벤치마크테스트 결과에 대한 신뢰성이 확보되어 있고, 현재 테스트 기관에서 1차적으로 시험된 결과를 별도로 검증해 줄 수 있는 구조를 마련하여 잘못된 시험 결과가 공표되는 것을 예방하고 있어 이의 제기가 거의 없다. 또한 이의 제기가 있을 경우 동일한 시험 환경에서 재시험을 수행하여 결과를 확인할 수 있게 하고 있다.

상대적으로 유럽은 독일의 TUViT, 덴마크의 DELTA등 다수의 S/W 벤치마크테스트 기관이 존재하고 있지만 S/W 시장의 대부분을 미국의 S/W 제품이 점유하고 있어서인지 벤치마크 관련 수요가 많지 않은 형편이다.

1.2. 소프트웨어 성능 평가 및 검증 절차

1.2.1. 소프트웨어 벤치마크테스트

가. 벤치마크 테스트를 통한 공개소프트웨어 검증 절차[김두연06b]

최근 정부・공공기관의 각종 정보시스템 구축에 공개소프트웨어의 사용이 권장되고 있다. 그러나 현재 많은 공공기관의 중요한 시스템 구축 사업에서는 공개소프트웨어 도입의 기준, 절차, 성능 및 안정성 등에 대한 신뢰할 만한 데이터가 부족하여 공개소프트웨어 도입을 꺼리고 있다.

정책적인 고려만으로 공개소프트웨어를 도입하였다가 시스템 구축 후 기술적인 문제가 발생할 경우에 여러 가지 어려움에 직면할 수 있다. 따라서 시스템 구축 후 발생할 수 있는 제반 문제를 예방하기 위해서는 사전에

무엇을 어떻게 검증하는 것이 타당한가 하는 것이 담당자의 고민이다.

한편, 국내 S/W 산업이 점차 성숙해감에 따라 다양한 동종 제품 출시되고 있다. 구매자는 이 중에 최적의 제품을 어떻게 선택할 것인지가 새로운 과제로 등장했다. 점차 일반화되고 있는 S/W 벤치마크 테스트는 구매자가 객관적으로 각 제품의 장단점을 파악할 수 있게 도와줌으로써 구매 목적에 더욱 근접한 제품을 선택하는데 유용한 지침이 된다.

공개소프트웨어의 도입을 위한 의사 결정의 과정에 이러한 벤치마크 테스트의 절차와 기법이 적용된다면 보다 객관적이고 신뢰할 수 있는 데이터를 확보할 수 있다. 이에 본 논문에서는 일반 상용소프트웨어에 비해 공개소프트웨어가 가지는 특성과 위험요소를 고려하여 공공기관의 중요한 시스템에 공개소프트웨어를 도입하기 위한 벤치마크 테스트 절차를 제시하였다. 특히, 공개소프트웨어를 적용한 세계 최대 규모의 시스템인 교육행정정보시스템(NEIS : National Education Information System)의 사례를 중심으로 본 연구의 유효성을 검증하였다.

1) 서론

최근 정부・공공기관의 각종 정보시스템 구축에 공개소프트웨어의 사용이 권장되고 있다. 그러나 현재 많은 공공기관의 중요한 시스템 구축 사업에서는 공개소프트웨어 도입의 기준, 절차, 성능 및 안정성 등에 대한 신뢰할 만한 데이터가 부족하여 공개소프트웨어 도입을 꺼리고 있다.

정책적인 고려만으로 공개소프트웨어를 도입하였다가 시스템 구축 후 기술적인 문제가 발생할 경우에 여러 가지 어려움에 직면할 수 있다. 따라서 시스템 구축 후 발생할 수 있는 제반 문제를 예방하기 위해서는 사전에 무엇을 어떻게 검증하는 것이 타당한가 하는 것이 담당자의 고민이다.

한편, 국내 S/W 산업이 점차 성숙해감에 따라 다양한 동종 제품 출시

되고 있다. 구매자는 이 중에 최적의 제품을 어떻게 선택할 것인지가 새로운 과제로 등장했다. 점차 일반화되고 있는 S/W 벤치마크 테스트는 구매자가 객관적으로 각 제품의 장단점을 파악할 수 있게 도와줌으로써 구매 목적에 더욱 근접한 제품을 선택하는데 유용한 지침이 된다.

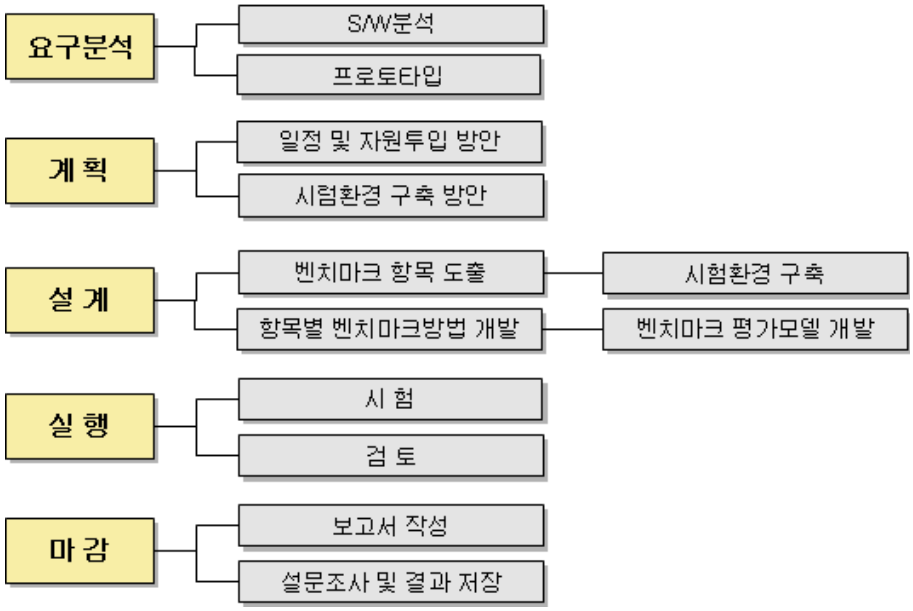
공개소프트웨어의 도입을 위한 의사 결정의 과정에 이러한 벤치마크 테스트의 절차와 기법이 적용된다면 보다 객관적이고 신뢰할 수 있는 데이터를 확보할 수 있다. 이에 본 논문에서는 일반 상용소프트웨어에 비해 공개소프트웨어가 가지는 특성과 위험요소를 고려하여 공공기관의 중요한 시스템에 공개소프트웨어를 도입하기 위한 벤치마크 테스트 절차를 제시하였다. 특히, 공개소프트웨어를 적용한 세계 최대 규모의 시스템인 교육행정정보시스템(NEIS : National Education Information System)의 사례를 중심으로 본 연구의 유효성을 검증하였다.

본 연구는 정부를 비롯한 공공기관에서 중요 시스템에 공개소프트웨어를 도입할 경우, 기술적 검토 절차 및 성능검증 방법에 대한 지침으로 활용할 수 있다.

2) 관련 연구

⊙ 소프트웨어 벤치마크 테스트

국내 S/W 산업이 점차 성숙해감에 따라 유사한 제품이 많이 개발되고 이들은 점차 제품군을 형성해 가고 있다. 이러한 상황에서 S/W 벤치마크테스트는 구매자가 객관적인 자료를 통해 구매 목적에 더욱 근접한 제품을 선택하는 유용한 지침이 된다. 반면 개발사 입장에서는 동종 제품들 간의 객관적인 비교평가를 통해 각 제품의 강점을 파악하고, 취약점을 알아내는 등 개선방안을 제시할 수 있으므로 S/W 품질 향상에 큰 도움을 주게 된다.



<그림59> S/W 벤치마크 테스트 프로세스

효율적인 벤치마크테스트 수행을 위해서는 벤치마크테스트 프로세스 정의가 선행되어야 한다. <그림59>는 일반적인 벤치마크테스트 수행을 위한 프로세스를 도식화한 것이다.

요구분석 단계는 시험 대상 항목과 벤치마크테스트 항목 리스트를 도출하기 위해 대상 S/W에 대해 철저한 분석을 한다. 분석 이후 항목 도출 시에는 벤치마크테스트에 참여하는 제품의 업체 개발자 및 담당자들과의 미팅을 통해 도출 항목에 대해 합의하고, 실제 벤치마크테스트 가능 여부를 검증하기 위해 프로토타입 단계를 포함하여 설계단계에 앞서 준비해야 할 조건을 파악한다. 계획단계에서는 요구분석 단계에서 도출한 결과물들을 토대로, 일정 및 자원 투입 계획을 구체적으로 산정한다. 별도의 교육이 필요한 경우 교육계획을 수립하며 전문가 그룹을 구성한다. 시험환경 구축

을 위한 환경설정과 시험도구 및 S/W 선정 등이 계획단계에서 이루어진다. 설계 단계에서는 계획 수립 이후 도메인별 전문가 그룹을 구성하여 시험 대상 주요 항목 및 부가항목을 선정한다. 벤치마크테스트 항목 확정 이후 항목별 측정기준 및 공식을 도출하여 메트릭을 작성하고, 벤치마크 테스트에 참여하는 업체들과 항목에 대한 합의를 진행한 후 벤치마크테스트 평가모델을 완성한다. 실행 단계에서는 개발된 평가모델의 항목별 벤치마크테스트를 수행한다. 비교대상 제품별로 동일한 조건에서 시험을 하고 시험결과가 도출되면 결과에 대한 문제점을 정밀 검토 후, 필요한 경우 보완시험을 수행하고 결과를 정리한다. 마감 단계에서는 벤치마크테스트의 절차, 방법 및 비교분석 결과를 정리 검토하고 결과에 대한 고객 의견을 조사한다. 최종적으로 벤치마크테스트 과정 및 결과를 저장한다.

⊙ 공개소프트웨어

공개소프트웨어는(Open Source Software) 1984년 자유소프트웨어(Free Software)운동으로부터 분화하여 현재에 이르렀으며, 소스코드에 접근할 수 있는 권리, 프로그램을 복제하여 배포할 수 있는 권리, 프로그램을 개선할 수 있는 권리를 보장한다. 이러한 공개소프트웨어는 소스코드에 대한 접근성이 보장되므로 시스템 간의 호환성을 확보 할 수 있을 뿐 아니라 사용자의 요구에 맞는 일관성과 함께 일치성을 보장받을 수 있다.

2002년부터 정부에서 공개소프트웨어 정책추진을 위한 본격적인 사업을 착수하여 현재 그 성과를 보이고 있으나, 이러한 적극적인 정책지원과 비용 절감에 대한 기대에도 불구하고 현재 중요 업무 시스템(Mission Critical System)에 활발히 도입되지는 못하고 있다. 공개소프트웨어 도입을 주저하는 대부분의 공공부문 기술 관계자들의 의견은 공개소프트웨어 도입에 대한 검증된 데이터가 부족하여 신뢰할 수 없다는 것이다. 또한, 공개소프트웨어에 대한 체계적인 기술지원, 안정적인 유지보수, 각종 응용

소프트웨어가 부족 등도 공개소프트웨어를 도입을 꺼리는 이유 들이다. 그러나 이것은 공개소프트웨어에 대한 오해와 편견에서 비롯된 것이며 이를 객관적으로 평가하면 다음과 같다.

첫째, 공개소프트웨어의 기술지원 문제이다. 공개소프트웨어에 대한 기술지원을 받고 싶으나 지원업체를 찾기가 어렵다고 호소한다. 그러나 대표적인 공개소프트웨어인 리눅스는 배포판 업체뿐만이 아닌 다국적 하드웨어 업체는 물론 국내 소프트웨어 개발 업체들로부터 기술지원을 직접 받을 수 있다.

둘째, 공개소프트웨어의 신뢰도의 문제이다. 독점소프트웨어 업계와 마찬가지로 공개소프트웨어도 매우 다양한 소프트웨어를 사용할 수 있다.

2004년 10월 포레스터 리서치가 미국 140개 대기업을 대상으로 조사한 결과에 따르면 조사 대상 업체의 53%가 자사의 주요 핵심 업무용 소프트웨어 운영환경으로 리눅스를 활용하고 있으며, 새로운 업무용 소프트웨어 도입에 리눅스를 선택할 의사 또한 52%에 달하는 것으로 나타났다.

셋째, 성능과 안정성의 문제이다. 리눅스는 유닉스보다 성능이 떨어진다는 인식이 일반적이나,TPC-C나 SPECfp등에서 제공하는 각종 벤치마크 결과를 보면 리눅스는 유닉스와 윈도우에 비해 높은 성능을 보이고 있다. 특히 국내 사례에서 운영체제 수준의 운용에서 안정성이 뛰어난 것으로 평가받고 있다.

⊙ 공개소프트웨어의 벤치마크 테스트

▪ 공개소프트웨어 도입 위험 요인

공개소프트웨어를 도입하는 방법에는 크게 3가지 방법이 있다. 첫째, 기관이 직접 선택하여 도입하는 직접선택, 둘째, 민간기업 등 외부의 추천에 의한 간접공급, 셋째, 기관 내부에서 공개소프트웨어를 개발 및 수정하

는 내부개발 등이다. 공공기관에서 이러한 3가지 방법의 하나를 선택하여 공개소프트웨어를 도입할 수 있지만 국내 공개소프트웨어 업계와 공공에서의 현실을 고려할 때에 가장 실용적이고, 일반적으로 사용할 수 있는 방법은 간접공급 방식이다.

따라서 간접공급에 의한 위험 요인을 분석해 본다. 먼저 하자보증에 대한 위험요인이다. 1차적으로 공급업체는 하자보증에 대해 지원을 해야 한다. 하지만, 공개소프트웨어의 GPL(general public license, 일반 공중 허가서) 라이센스가 적용된다면 근원적인 문제점에 대해서는 공급업체에서 보증에 대한 책임을 회피할 수 있다. 따라서 공개소프트웨어를 도입하는 기관 담당자는 보증과 관련한 위험요소에 대한 대비책을 세워두어야 한다.

다음으로, 일반적인 공급계약방식에서 발생할 수 있는 위험요소가 고려되어야 한다. 공급업체의 적격성에 대한 위험성 평가, 시스템의 확장성 가능 여부에 대한 위험요소 평가, 소스코드의 성숙도 및 신뢰도에 대한 위험요소 평가 등이 모두 고려되어야 한다.

또한, 공급업체가 지원하는 공개소프트웨어가 최종 시스템 환경에서 당초 요구한 성능과 안정적 운영을 보장하는지에 대한 위험요소이다. 이는 유사 사례분석과 벤치마킹 테스트 등 충분한 성능검증 절차를 거쳐 위험요인을 최소화해야 한다.

마지막으로 간접방식에서는 외부 공급업체의 기술지원 능력에 대한 위험요소가 존재한다. 즉, 공급업체의 기술지원 능력에 대한 객관적인 검증 없이 공개소프트웨어를 도입하였다면 도입 이후의 기술지원 문제가 발생할 수 있다는 점을 명심해야 한다. 아울러 비공개소프트웨어의 경우와 마찬가지로 공급업체의 파산 등으로 기술지원을 받지 못하는 사태가 발생할 수 있으므로 동일 솔루션의 기술지원업체에 대한 정보를 확보해 두는 것도 공개소프트웨어의 장점을 극대화하여 위험 요소를 줄일 수 있는 좋은 방법

이다.

⊙ 공개소프트웨어 벤치마크 테스트 절차

▪ 벤치마크 테스트 환경

시스템 테스트의 목적은 설치 후 시스템이 성공적으로 가동하는 데 있다. 시스템 도입 전 하는 벤치마크 테스트는 최종 시스템 성능에 직접 영향을 미치게 된다. 따라서 테스트 대상 업무, 방법 및 항목 등 테스트 환경은 최종 시스템 운영에 매우 큰 영향을 미치게 된다. 본 연구에서는 먼저 테스트 방법을 2개 영역으로 나누고 영역별 테스트 항목을 제시하고, 다음으로 테스트 대상 업무 선정을 위한 기준을 제시하였다.

시스템 테스트 방법은 성능 테스트와 과부하 테스트 등 2가지 영역으로 나누어 실시한다. 성능 테스트는 운영시스템 환경에서 시스템 구성요소별로 부하를 적절히 처리하는지 테스트하는 것이다. 확인하는 세부항목은 처리량, 자원 사용량 (CPU, Memory), Disk I/O, 핵심업무 응답시간등이다. 과부하 테스트는 요구사항의 한계 또는 그 이상에서 시스템 구성요소별로 스트레스를 적절히 분배하는지 테스트한다. 그 세부 점검 항목은 과부하상태에서 응답시간, 시스템 자원 사용량 등이다.

테스트 대상 업무는 시스템 테스트의 유효성을 높일 수 있도록 최대한 다양한 경우의 케이스를 선정한다. 본 연구에서는 다음과 같은 기준을 제시한다.

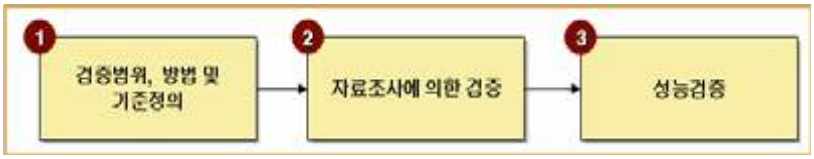
- 사용빈도가 높은 업무를 선정한다. 평상시 자주 사용되는 화면이 그 예 중 하나이다.
- 특정시간(Peak-time)에 많이 사용되는 업무를 선정한다. 학교는 학기 초에 집중적으로 사용되는 화면이 그 예이다.
- 복잡한 업무를 선정한다. 프로세스가 복잡한 업무, 여러 사이트가 공통으로 사용되는 업무가 된다.

- 많은 양의 데이터를 발생하는 업무를 선정한다. 데이터 처리량이 많은 업무가 그 사례이다.
- 네트워크에 많은 부하를 발생시키는 업무를 선정에 고려한다.
- 온라인 성능에 영향을 줄 수 있는 일괄처리 업무를 선정에 고려한다. 온라인 운영 중 실행되는 배치작업이 그 대상이다.
- 아키텍쳐 구성상 부하 경로를 검증할 수 있는 업무를 선정한다. Web만 부하가 발생하는 화면, Was까지 부하가 발생하는 화면, DB까지 부하가 발생하는 화면 등 각각 선정한다.

위에서 제시한 테스트 환경은 정부나 공공기관의 일반적인 업무프로세스를 고려했다는 점에서 기업 등 특수한 업무프로세스에는 일정한 한계가 있을 수 있다.

▪ 공개소프트웨어 검증 절차

기준과 검증 밥법은 각각 [그림 2] 및 <표 1>과 같다. [그림 2]는 공개소프트웨어의 도입에 따르는 위험 요인을 고려한 벤치마크 테스트 절차이다.



<그림60> 공개 소프트웨어 벤치마크 테스트 절차

공개소프트웨어 벤치마크 테스트 절차는 검증 범위와 방법 및 기준의 정의, 자료조사에 의한 검증, 벤치마크 테스트를 통한 성능검증 단계로 구성된다. 이는 목표 시스템(TO-BE 시스템)의 아키텍처(Architecture) 수

립에 방향을 제시하는 핵심 절차이다. 먼저, 검증 범위와 방법 및 기준을 정의 단계는 공개소프트웨어 전문가 참여하에 검증의 범위와 방법 및 기준을 상세히 정의한다. 자료조사에 의한 검증 단계는 공개소프트웨어 도입에 따르는 위험을 최소화하고, 직접 검증에 따르는 비용을 최소화하기 위한 사전 조사 작업으로 시장추세, 기능 및 성능, 지원체계, 적용사례(Reference Site) 등과 같은 관점에서 분석한다.

다음의 <표66> 은 자료조사를 통한 검증의 기준이다.

<표66> 검증의 기준

구분	내용
시장추세	▪ 공개 소프트웨어의 시장동향 및 향후기술 추세에 따른 지속적인 제품기능 강화 여부 ▪ 공개 소프트웨어 주 활용도 분석을 통한 새로운 시스템(TO-BE)에 적용 가능성 여부
기능 및 성능	▪ 공개 소프트웨어 보유기능 및 안정성 ▪ 성능과 안정성을 지원하는 주요기능 구비여부(컨넥션 풀링, 클러스터링, 보안성,DBMS 트랜잭션 등) ▪ 오픈 아키텍처 설계를 위한 표준화 지원 여부 ▪ 상용 소프트웨어와의 기능 비교 분석 ▪ 신뢰성 있는 기관의 공개 소프트웨어 성능 검증 결과를 통한 성능 분석
지원체계	▪ 공개 소프트웨어 공급 업체 현황과 체계적 기술지원 여부 ▪ 공개 소프트웨어의 법적 분쟁에 대비한 고객프로그램 보유 여부 ▪ 사후관리 분석
적용사례	▪ 주요기관의 공개 소프트웨어 적용 사례
전문기관 의견	▪ 정보화정책 및 정보시스템 전문기관 의견 ▪ 공개 소프트웨어 전문기관(한국소프트웨어 진흥원 등)의 평가의견 ▪ 정보보호 전문기관 및 기타 관련 기관 의견

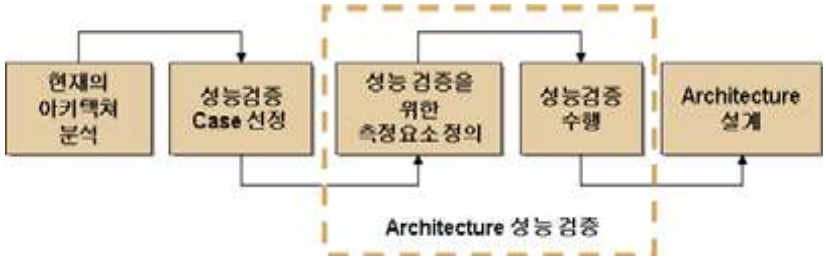
다음으로, 성능 검증 단계는 기술적 검증의 단계로 벤치마크 테스트를 통한 성능을 검증한다. 이를 위해서 사전에 도입 검토 후보 시스템을 대상

으로 핵심 업무 모델링을 통한 기초자료 확보와 테스트 환경 정의, 테스트 데이터와 애플리케이션 프로토타입을 개발하는 등 <표67>와 같이 성능 검증을 위한 기본모델을 수립한다.

<표67> 성능 검증 기본모델

단계	단계별 주요 수행업무	비 고
1	업무 특성 및 처리량 분석	▪ 업무 특성에 따른 트랜잭션을 모델링하여 데이터 용량(I/O) 산출
2	서버 용량 산정	▪ 서버의 용량산정(CPU 점유율, 응답시간, 개인식별정보 암호화)
3	서버 성능검증 및 결과 분석	▪ 리눅스, 유닉스 등 운영체제에 대한성능 검증
4	종합평가	▪ 안정성, 처리 성능, 경제성, 관리효율성, 보안성 등을 종합평가

기본모델 수립은 작업의 객관성 확보와 체계적 추진을 위하여 <표67>와 같은 4단계 절차에 따라 작업이 진행되며, 실질적인 성능검증은 다음의 <그림60>과 같은 절차에 따라 수행한다.



<그림60> 성능 검증 절차

현재의 아키텍처 분석은 현황분석 및 요구사항 정의 과정에서 도출된

사항을 바탕으로 현재의 IT 환경에서 구현 가능한 케이스(Case)를 개발하는 과정이다. 이는 시스템구현 이미지(Image)를 구성하고, 시스템 구성 명세(Draft)를 정의하는 단계이기도 하다. 다음으로, 성능검증 대상(Case)을 선정하고 측정요소를 정의한다. 성능측정 요소에 대한 정의는 성능검증 시나리오에 반영하며, 이 시나리오에 따라 아키텍처(Architecture) 성능 검증을 수행하며, 그 결과에 대한 분석을 토대로 아키텍처를 설계한다. 여기서 제시한 공개소프트웨어 벤치마크 테스트 절차는 공개소프트웨어의 위험요소와 특성을 고려한 결과 자료조사에 의한 검증 단계가 추가되었다. 즉, 이 본문에서 제안하는 벤치마크 테스트 절차는 공개소프트웨어 도입에 가장 적합한 품질평가 기준 중 하나이다. 반면 현재의 아키텍처 분석은 현황분석 및 요구사항 정의 과정에서 도출된 사항을 바탕으로 현재의 IT 환경에서 구현 가능한 케이스(Case)를 개발하는 과정이다. 이는 시스템구현 이미지(Image)를 구성하고, 시스템 구성 명세(Draft)를 정의하는 단계이기도 하다. 다음으로, 성능검증 대상(Case)을 선정하고 측정요소를 정의한다. 성능측정 요소에 대한 정의는 성능검증 시나리오에 반영하며, 이 시나리오에 따라 아키텍처(Architecture) 성능 검증을 수행하며, 그 결과에 대한 분석을 토대로 아키텍처를 설계한다. 여기서 제시한 공개소프트웨어 벤치마크 테스트 절차는 공개소프트웨어의 위험요소와 특성을 고려한 결과 자료조사에 의한 검증 단계가 추가되었다. 즉, 이 본문에서 제안하는 벤치마크 테스트 절차는 공개소프트웨어 도입에 가장 적합한 품질평가 기준 중 하나이다. 반면 관련 연구가 많지 않고 적용사례가 부족하여 신뢰성을 입증하기 어려운 한계도 있다.

⊙ 적용 사례

▪ 적용 대상과 검증 환경

우선 본 연구의 적용사례인 NEIS의 교무업무 시스템은 16개 시?도 교

육청 단위로 단독 또는 그룹서버로 운영한다. 특수학교와 고등학교는 1개 학교당 1대의 단독 서버로, 초.중학교는 15개 학교당 1대의 그룹서버로 구성되었다. 목표 시스템의 서버 구성은 단독DB 서버 2,338대, 그룹 DB 서버 606대, WEB/WAS 서버 393대 등 3,300여대에 달한다. 여기에 학교의 교무/학사, 입(진)학,보건 업무가 운영되며 인증서를 발급받은 이용자는 교사 등 약 432,000여 명, 학생 약 7백만여 명의 자료가 DB화되었다. 테스트를 위한 단독서버 사용자는 최대 120명, 15개교를 기준의 그룹서버는 최대 이용자 800명을 기준으로 하였다.

성능 검증을 위한 시험시스템은 실제 학교에서 운영하는 시스템과 같게 구성하였다. DB서버는 단독서버 1대, 그룹서버 1대 그리고 WEB/WAS 서버로 구성하였다. 단독서버는 CUP 2개와 메모리 4GB이며 그룹서버는 CPU 4개와 메모리 8GB이며, 시스템 소프트웨어는 운영체제로 리눅스와 유닉스, DBMS, WAS 등으로 구성하였다. 응용소프트웨어는 NEIS 교무/학사 3영역이다.

성능검증은 하드웨어나 소프트웨어의 기본성능을 평가하는 목적이 아니라 교무업무시스템에 필요한 응용소프트웨어를 탑재하여 어느 정도의

성능을 발휘하는가를 검증하는 것이다. 따라서 모든 종류의 유닉스와 리눅스 플랫폼에서 검증을 하지 않고, 요구되는 트랜잭션 처리 성능 만족 여부에 대해 검증을 하는 것으로 그 범위를 제한하였다. 또한, 검증을 위한 애플리케이션과 데이터는 현재의 NEIS 애플리케이션과 샘플 데이터를 활용하였고, 측정 도구는 Mercury사의 Loadrunner를 사용하였다.

3) 공개소프트웨어 검증 수행

⊙ 성능검증 범위

공개 소프트웨어의 검증은 본 연구에서 제시한 절차를 적용하여 기술

검증을 통한 적용 가능성을 검토하고, 성능 검증 및 최적화 작업을 수행하였다. 우선, 검증 범위의 대상 공개소프트웨어는 운영체제로는 리눅스를, 시스템소프트웨어로는 Apache,MySQL 등을 선정하였다.

<표68> 검토 대상 공개소프트웨어

구 분	검증대상 공개소프트웨어
운영체제	▪ 리눅스
시스템 소프트웨어	▪ 웹서버용(Apache) ▪ 웹어플리케이션용(Tomcat) ▪ DBMS용 (MySQL)

1차적으로 자료 조사를 통해 검토한 결과 리눅스 운영체제는 시장동향, 기능 및 성능분석, 지원체계 등에서 활용 가능하다고 판단되어 성능 검증 대상에 포함하였으나, 시스템 소프트웨어(Apache,Tomcat, MySQL)는 요구기능 미흡 및 시스템 컴포넌트 간의 연계성이 미흡하여 교무업무시스템 구축을 위한 공개소프트웨어 활용 대상에서 제외하였다. 다음으로, NEIS 교무업무시스템 구축과 관련된 아키텍처 설계를 위하여 리눅스 및 유닉스 시스템을 대상으로 핵심 업무 중심의 서버 성능검증(서버용량 산정을 위한 기초자료 확보,리눅스 타당성 검증)을 실시하였다.

⊙ 성능검증 항목 및 시나리오

기존 NEIS의 교무/학사 업무 중 가장 시스템 부하가 많은 학교생활기록부 업무(조회, 저장, 마감)와 월 출결 업무(조회, 저장, 마감)를 선택하였다. 성능검증은 사용자 수 증가에 따른 시스템 응답시간과 CPU 사용률을 측정하고, 개인 식별 자료 중심으로 최소한의 항목(주민번호, 이름, 석차,번호)을 암호화하여 암호화에 따른 성능 차이를 검증하였다. 필요 이상의 모

든 항목을 암호화할 경우 시스템에 많은 부담이 되기 때문에 누구의 자료인지 식별할 수 없도록 학생 개인의 식별정보를 DB저장 시에 암호화함으로써 학생 자료에 대한 신뢰성과 무결성, 보안성 등을 확보할 수 있다.

<표69> 성능검증 시나리오

단독서버 (Web/Was/DB)	그룹서버		암호화 적용
	물리적 2 Tier (Web/Was-DB)	물리적 3 Tier (Web-Was-DB)	
LX UX	UX - UX LX - LX LX - UX	UX - UX - UX LX - LX - LX LX - UX - UX	UX - UX LX - LX LX - UX

UX : UNIX, LX : LINUX

<표69>의 성능검증 시나리오는 단독서버, 그룹 서버 그리고 암호화 적용 여부로 크게 구분하고, 그룹서버는 2 Tier와 3 Tier 구조로 나누어 구성한다. 이 4가지 경우별 시나리오는 UNIX와 LINUX의 경우의 수 조합이다.

⊙ 성능검증 기준, 내용 및 결과

교무업무시스템 시범사업을 위해 시험 시스템을 구축한 후 진행한 시험운영 환경, 성능 기준, 시험내용 및 시험결과를 정리하면 <표70>과 같다. 리눅스를 기반으로 한 시범학교의 NEIS 교무업무시스템에 보안 시스템을 탑재하고, 학생 개인 식별정보를 암호화 후 적용한 시험운영에서 업무처리 속도 및 자원 사용률 기준을 모두 통과함에 따라 교무업무시스템의 전국단위 물리적 기반 구축 사업에서 단독DB 서버 2,300여 대에 최종적으로 리눅스가 채택되었다. 교무업무시스템의 단독DB 서버에 리눅스 도입 이 후에도 성능검증 및 최적화 작업(2회)을 통하여 시스템 개통 이전에 최상의 성능을 발휘 할 수 있도록 최적화 작업을 수행하였다.

<표70> 공개 S/W 성능검증 요약

구분	기준
검증 환경	▪ 실제 운영환경과 동일하게 구성하여 성능시험 실시 ▪ 대상 : 단독 및 그룹서버, WEB/WAS 서버 ▪ H/W : 단독서버(CPU 2개, 메모리 4GB), 그룹서버(CPU 4개, 메모리 8GB) ▪ 주요 시스템S/W : O/S(단독DB서버 : 리눅스, 그룹 DB서버 : 유닉스), DBMS (UniSql), WAS(JEUS) 등 ▪ 응용 S/W : 실제 새롭게 개발 완료된 응용S/W 중 최대 부하를 유발하는 학교생활기록부 조회,반영 부분을 시험 대상항목으로 선정
성능 기준	▪ 1개 고등학교의 최대 사용자 수 : 120명(동시 사용자 수 14명) ▪ 15개 초/중학교의 최대 사용자 수 : 800명(동시사용자 수 96명) ▪ 응답시간 : 평균 3초 이내(배치성 업무 제외) ▪ 자원(CPU, 메모리) 사용율 : 75% 이하
시험 내용	▪ 학교생활기록부 관련 개발 응용S/W 적용 시험 ▪ 동시 사용자 수(단독서버 : 14명, 그룹서버 : 96명)를 50%, 100%, 200%로 조정하면서 학생목록 및 해당 학생의 학교생활기록부 조회, 반영 ▪ 등 시험
시험 결과	▪ 시험결과 응답시간, 자원 사용률 모두 성능기준통과 ▪ 업무처리 속도 : 응답시간은 평균 3초 이내로 기준 통과 ▪ 자원 사용율 : CPU 및 메모리 사용율 모두 70%이내로 기준 통과 ▪ 시스템S/W : DBMS, Web/Was 등 모두 성능 및 안정성 통과

4) 평가 및 결론

본 연구에서는 공공기관의 중요 시스템에 공개소프트웨어를 도입하기 위한 검증 절차를 검증범위와 방법 및 기준의 정의, 자료조사에 의한 검증, 성능검증 등 3단계로 제안하였다. 또한, 각 단계별 활동에 대하여 구체적인 내역을 제시함으로써 공개소프트웨어 검증 절차에 직접 적용이 가능하게 되었다. 특히, 공개소프트웨어 검증 절차의 핵심 활동인 성능검증은 5개 세부 활동으로 나누어 제안하였다. 공개소프트웨어 제반 위험요소를 일반 벤

치마크 테스트절차에 반영하여 자료조사에 의한 검증 단계가 추가되었다. 본 논문의 벤치마크 테스트에 의한 성능검증은 구매자가 제품 선정에 더욱 객관적이고 신뢰할 수 있는 데이터를 확보할 수 있다 데 장점이 있다.

또한, 벤치마크 테스트에 의한 성능검증은 교육 행정정보시스템(NEIS)의 교무업무시스템 구축에 공개소프트웨어 도입 사례를 중심으로 본 연구의 유효성을 검증하였다. NEIS 교무업무시스템에 공개소프트웨어 적용을 위한 성능검증 절차는 성능 검증 환경 구축, 성능검증 기준 및 시나리오 작성, 성능 시험 및 결과 측정 등의 순으로 진행하였다.

이렇게 공개소프트웨어 도입 이전에 사전 성능검증 등 충분한 검토 절차를 거쳐 교무업무시스템의 시범사업 중 단독서버에 공개소프트웨어인 리눅스가 도입되었다. 시범사업은 2개 시도교육청 관내 28개 시범학교 및 108개 학교를 대상으로 진행되었다. 시범사업 결과도 벤치마킹 테스트에 의한 성능 기준이 충족됨에 따라 본 사업의 단독서버 2,300여 대에 리눅스가 최종 적용되었다.

NEIS 사례와 같이 시스템에 공개소프트웨어가 도입되고 운영하기까지의 과정은 검증범위 및 기준 정의, 자료조사에 의한 간접 검증, 벤치마크 테스트에 의한 성능검증 등 다단계에 걸쳐 진행되었다. 그 결과 시스템이 정상 가동되고 있다. 이는 공공기관의 중요 업무시스템 구축 시 공개소프트웨어를 도입하기 위한 가장 적절한 업무 적용 모델이라 평가된다. 향후 전자정부 사업을 비롯한 중요사업에서도 본 논문에서 제시한 벤치마크 테스트에 의한 성능검증 등 검증 프레임워크를 활용한다면, 성공적인 공개소프트웨어 적용 시스템 구축과 함께 비용절감, 신뢰성 확보 등의 성과도 얻을 수 있다.

그러나 본 연구에서 제시한 검증 프레임워크는 대규모 사업에 적용한 사례이므로 향후 다양한 시스템 규모에 적용해 보고 이에 따른 보완과 적

용업무, 시스템 규모, 사업 규모 등에 따른 세분화된 모델 연구가 필요하다.

나. BMT를 공개소프트웨어의 성능 검증 프로세스[김두연06a]

국내 소프트웨어 산업이 점차 성숙해감에 따라 상용소프트웨어도 다양한 동종 제품 출시되고 있다. 구매자는 이 중에 최적의 제품을 어떻게 선택할 것인지가 새로운 과제로 등장했다. 이와 같은 경우에 소프트웨어의 비교 및 선정을 위한 방법으로 점차 일반화되고 있는 것이 소프트웨어 벤치마크테스트(BMT)이다. BMT는 구매자가 객관적으로 각 제품의 장단점을 파악할 수 있게 도와줌으로써 구매 목적에 더욱 근접한 제품을 선택하는 데 유용한 지침이 된다.

공개소프트웨어의 도입을 위한 의사 결정의 과정에서도 이러한 벤치마크테스트의 절차와 기법이 적용될 수 있으며, 이를 통해 보다 객관적이고 신뢰할 수 있는 데이터를 확보할 수 있다.

벤치마크테스트란 성능을 측정하려고 사용되는 절차, 도구 또는 프로그램을 나타내며, 주로 비교 평가를 위한 기준으로 사용되고 있다. 그러나 하드웨어 및 네트워크 제품과는 달리 소프트웨어는 객관적인 평가가 어려워 벤치마크테스트가 활발히 진행되지 못하고 있다. 소프트웨어 벤치마크테스트는 소비자에게는 우수 제품을 선택할 수 있는 비교 정보를 제공하고, 소프트웨어 개발 업체에는 자사 제품의 강점을 파악하고, 취약점 보안 및 개선방안을 제시해 줌으로써 경쟁력을 제고 하는데 크게 도움을 줄 수 있다

소프트웨어 벤치마크테스트 초창기에는 테스트 결과를 악용하여 자사의 강점은 부각시키고, 약점은 숨기는 수단으로 사용됐으나, 1973년, 미국 시카고에 Neal Nelson Associate가 설립되어, Unix-based 벤치마크를 수행하기 시작하였으며, 80년대 후반, TPC(Transaction Processing

Performance Council, http://www.tpc.com), SPEC(System Performance Evaluation Cooperative, http://www.spec.com) 등과 같은 컨소시엄이 구성되어 각종 벤치마크 모델을 정립하고 벤치마크의 악용을 상당 부분 감소시켰다

각국의 소프트웨어 벤치마크테스트 현황을 보면 다음과 같다.

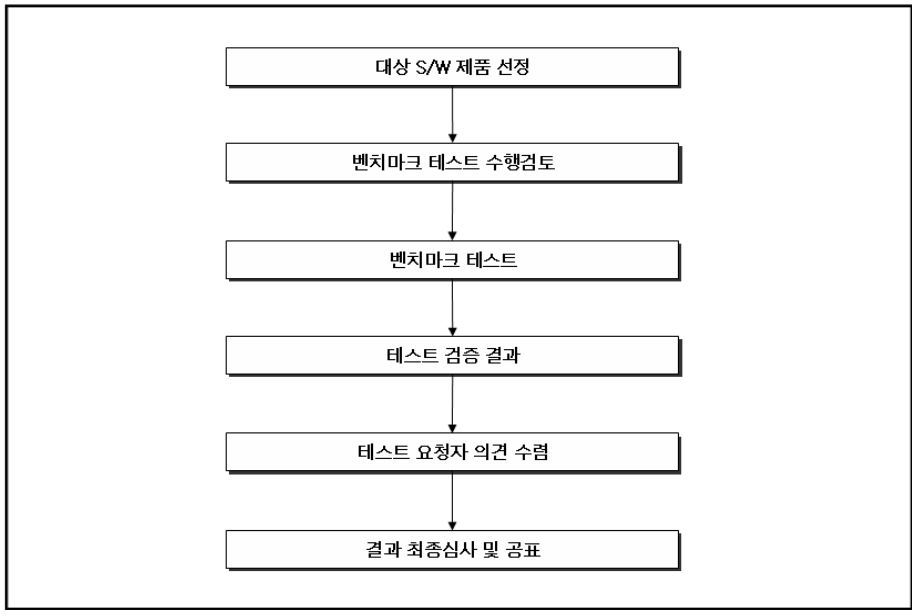
먼저 미국의 경우, 주요 벤치마크테스트 기관으로 NSTL (National Software Testing Labs, http:// www.nstl.com)을 들 수 있는데, 이 기관은 1983년에 NIST(미국 기술표준원)에서 분리되어, 1990년대부터 캐나다 정부의 IT조달 벤치마크 서비스를 대행하며, 북미 지역의 독보적인 시험기관으로 성장하였다. 그리고 VeriTest (http://www.veritest.com)는 2002년 7월 세계 최고의 벤치마크테스트 기관인 Ziff Davis의 eTesting Lab을 합병하여, 세계최고의 벤치마크테스트 도구 개발회사가 되었다. 또, TPC (Transaction Processing Performance Council, http://www.tpc.com)는 트랜잭션처리와 데이터베이스 분야에서 벤치마크를 정의하는 업체들의 컨소시엄으로, 대형 트랜잭션베이스 시스템의 성능을 측정키 위한 사실상(de facto) 표준기관이다. SPEC (System Performance Evaluation Cooperative, http://www.spec.com)은 1988년에 설립되어 60개 이상의 회원사를 가진 성능 표준화 기구로서, 28종의 벤치마크테스트 모델을 개발하여 활용 중이다

한편, 우리나라는 한국정보통신기술협회(TTA, http://www.tta.or.kr) 소프트웨어 시험인증센터를 제외하고는 대다수 하드웨어 벤치마크테스트만 수행할 뿐 소프트웨어 벤치마크테스트는 거의 중단된 상태라고 볼 수 있다. 예컨대 K-Bench(http://www.kbench.com), 보물섬(http://www.bomul.com), PcBee(http://www.pcbee.co.kr) 등의 벤처회사들이 한때 벤치마크테스트를 수행했지만 지금은 모두 중단된 상태이다.

정보보호학회(http://www.kiisc.or.kr)는 2002년도에 7종의 백신소프트웨어에 대한 벤치마크테스트를 수행했지만, 공정성 논란을 일으킨 적이 있으며, 한국정보통신기술협회 소프트웨어 시험인증센터에서는 2002년부터 개발자 및 구매자로부터 의뢰를 받아 벤치마크테스트 결과를 당사자에게만 통보하는 형태로 벤치마크테스트를 수행하고 있다. 현재, TTA 소프트웨어 시험인증센터를 제외하고는 벤치마크테스트 전문기관도 없고 수요도 활발하지 못한 상황이다. 또한, 벤치마크테스트 전문기관을 확대하고자 할 경우 초기 투자비용이 많이 요구되고, 전문 인력도 부족하며, 소프트웨어 분야별 벤치마크테스트 모델이 부족 하는 등 많은 문제점이 있다

TTA 소프트웨어 시험인증센터와 한국소프트웨어진흥원(KIPA)에서는 시장에서 임의 거둬들인 제품에 대해 소프트웨어 벤치마크테스트를 수행하고, 그 결과를 공표하는 프로세스를 개발하였으며, 소프트웨어 벤치마크테스트가 공정하고 투명하게 수행되고, 객관성을 유지할 수 있도록 하고자 외부 전문가로 이루어진 BMT 협의회를 구성하였다. BMT 협의회는 소프트웨어 벤치마크테스트 및 품질평가에 필요한 전문 지식을 갖춘 전문가와 결과 공표에 대한 법적 대응을 위한 변호사 등 15인 이내의 민간위원으로 구성되어 있다

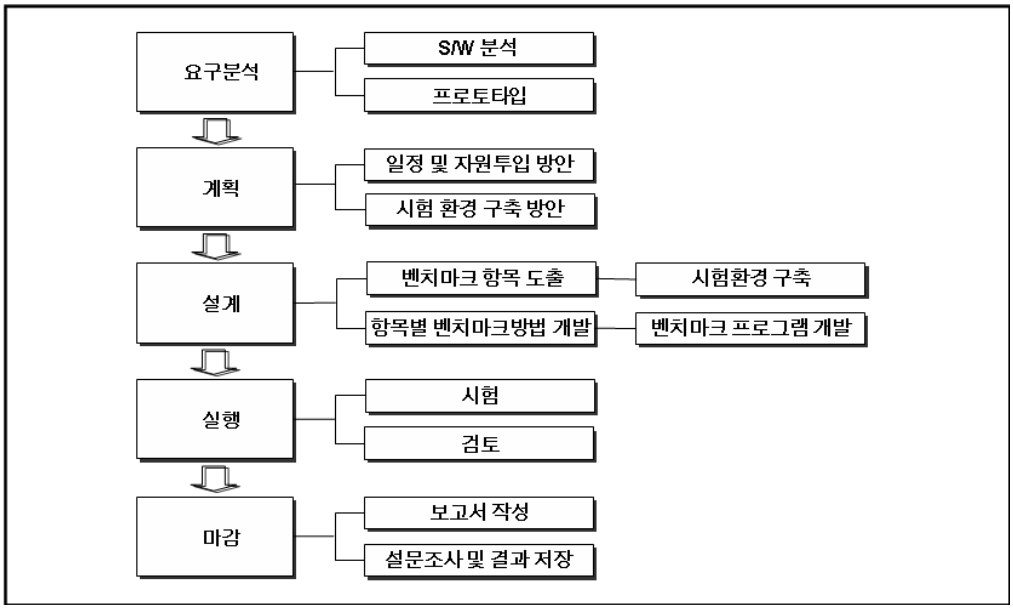
한국소프트웨어진흥원 (KIPA)의 BMT 협의회 BMT 결과 공표 프로세스는 다음의 <그림61>과 같다.



<그림61> BMT 협의회 절차

한국정보통신 기술협회의 BMT 평가 모델은 소프트웨어 벤치마크테스트 절차를 <그림62>와 같이 5단계로 제시하고 단계별 활동과 각 활동에 대한 비교 분석 작업 기준을 제시하고 있으며, 평가 모델 개발 이슈로 가중치, 배점 기준, 과락 항목을 제시하고 있다.

이 모델은 가중치 등 평가 모델 개발 이슈 적용을 객관화하기 어렵고, 시스템 활용 목적에 맞는 벤치마크테스트 평가 모델로 활용하는데 한계가 있으며, 벤치마크 수행에 대한 공개소프트웨어 특성을 반영한 평가 기준수립 등이 필요하다.



<그림62> BMT 평가 모델

다. S/W 벤치마크테스트 평가모델[정통기05a]

1) 서 론

국내 S/W 산업이 점차 성숙해감에 따라 유사한 제품의 개발이 많아지고 점점 군을 형성해가고 있는 추세이다. 다양한 동종 제품의 개발로 인해 사용자는 자신의 사용 목적에 보다.

적합한 제품을 선택할 수 있게 되었다. 그러나 많은 동종제품 가운데 사용자 자신이 다양한 제품을 직접 사용해보지 않고 용도에 맞는 최적의 제품을 선택하는 것은 쉬운 일이 아니다.

이때 객관적이고 공정한 S/W 벤치마크테스트 결과를 확보할 수 있다면 보다 쉽게 좋은 제품을 선택할 수 있을 것이다.

구매자 입장에서 S/W 벤치마크테스트는 객관적인 테스트 결과를 통해

각 제품의 장단점을 올바로 파악할 수 있게 하여 구매 목적에 보다 근접한 제품을 선택하는데 유용한 지침이 될 수 있다.반면 개발사 입장에서S/W 벤치마크테스트는 동종 제품들 간의 객관적인 비료 평가를 통해 각 제품의 강점을 파악하고, 취약점 보완 및 개선방안을 제시할 수 있으므로 S/W 품질 향상에 큰 도움을 줄 수 있다.

또 다른 목적의 벤치마크테스트는 유사 제품들 간의 기능 및 성능의 우선순위를 결정하는데 목적을 둘 수도 있다. 일례로 공공기관 및 대규모 조직에서 도입할 제품을 선정하고자 할 때, 중립의 위치에 있는 전문 기관에 제품의 우선순위를 결정하기 위한 벤치마크 테스트를 의뢰하는 경우도 최근 들어 많은 수요가 발생하고 있다.

기존의 비교 분석을 위한 벤치마크 테스트는 산출물로 각 평가 요소별 평가 값과 비교 분석서를 산출하지만, 요구에 의한 우선순위를 결정하는 벤치마크테스트는 부가적으로 목적 평가 기준에 기반 한 제품별 우선순위를 산출하게 된다. 이러한 순위 산출을 위한 벤치마크 테스트는 무엇보다 기본 목적이 명확해야 하며, 이런 벤치마크테스트는 시험에 응하는 제품을 대부분 수용할 수 있어야 한다.

비교분석을 위한 벤치마크테스트는 시험제품을 선정함에 있어 LINUX OS들에 대한 벤치마크테스트, 백신 S/W들 간의 벤치마크테스트 등과 같이 동종의 국한된 S/W를 대상으로 시험을 진행하는 반면, 순위를 결정하는 벤치마크 테스트는 벤치마크테스트의 목적에 부합하는 어떠한 제품도 시험에 응할 수 있다.

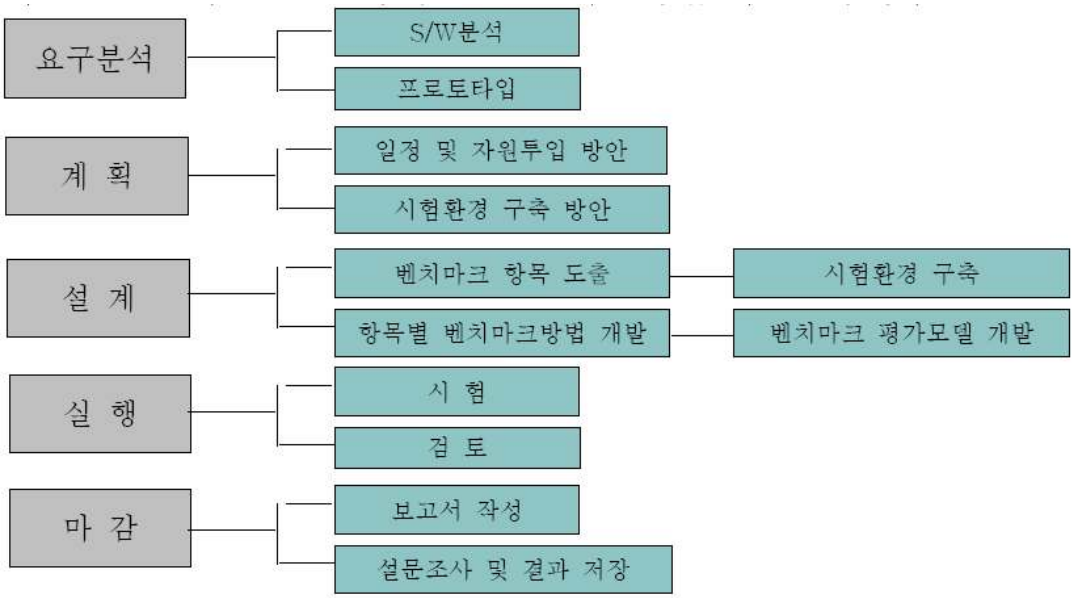
이러한 소비자의 새로운 요구사항을 반영하기 위해서는 기존의 벤치마크테스트 평가모델과 구별되는 특수 요소가 반영된 새로운 S/W 벤치마크테스트 평가모델의 개발이 절실히 필요하다.

본 연구에서는 이러한 요구사항을 만족시킬 수 있는 벤치마크테스트

평가모델 개발 및 프로세스에 대해 연구하고 있으며. 향후 산업계에서 벤치마크테스트 수행 시 지침으로 활용 할 수 있도록 하고자 한다.

2) 프로세스 정의

효율적인 벤치마크테스트 수행을 위해서는 벤치마크테스트 프로세스 정의가 선행되어야 한다. <그림63>은 일반적인 벤치마크테스트 수행을 위한 프로세스를 단계, 활동, 작업으로 구분하여 도식화한 것이다.



<그림63> S/W 벤치마크테스트 프로세스

⊙ 벤치마크테스트 프로세스

▪ 요구분석 단계

시험 대상 항목과 벤치마크테스트 항목 리스트를 도출하기 위해 대상 S/W에 대한 철저한 분석을 실시한다. 분석 이후 항목 도출 시에는 벤치마크테스트에 참여하는 제품의 업체 개발자 및 담당자들과의 미팅을 통해 도

출 항목에 대해 합의하고, 실제 벤치마크테스트 가능 여부를 검증하기 위해 프로토타입 단계를 포함아여 설계단계에 앞서 준비해야 할 조건을 파악한다.

<표71> 유형별 벤치마크테스트 프로세스

단계	활동	비교분석 벤치마크테스트 작업	순위도출 벤치마크테스트 작업
요구분석	S/W분석 및 업체 상담	1. 대상 S/W 분석(프로그램, 매뉴얼, 기타 관련자료) 2. 시험 대상항목 및 벤치마크테스트 항목 리스트 도출 3. 해당 업체들과 상담 (벤치마크테스트항목 등) 4. 항목별 시험 절차 및 방법 정의 5. 투입공수 및 수수료 산정	1. 대상 S/W 분석(프로그램, 매뉴얼, 기타 관련자료) 2. 시험 대상항목 및 벤치마크테스트 항목 리스트 도출 3. 벤치마크테스트 위뢰기관과 상담(벤치마크테스트 위뢰기관이 요구하는 항목 및 일반적인 벤치마크테스트 항목 등) 4. 항목별 시험 절차 및 방법정의 5. 투입공수 및 수수료 산정
계획	일정 및 자원 투입 방안	1. 인력 및 자원 투입 계획 2. 일정 계획 3. 교육 계획 4. 전문가 그룹 구성(Optional)	1. 인력 및 자원 투입 계획 2. 일정 계획 3. 교육 계획 4. 전문가 그룹 구성
	시험 환경 구축 방안	1. 시험 환경 설정 2. 시험 도구 및 s/w 선정	1. 시험 환경 설정 2. 시험 도구 및 S/W 선정
설계	벤치마크테스트 항목 도출	1. 전문가 그룹 운영(Optional) 2. 벤치마크 항목 확정 3. 메트릭 작성(항목별 측정기준 및 공식 도출) 4. 해당 업체들과 합의	1. 전문가 그룹 운영 2. 벤치마크 항목 확정 3. 항목별 가중치 부여 4. 항목별 배점 부여 5. 과락 항목 선정 6. 메트릭 작성 (항목별 측정기준 및 공식 도출) 7. 전문가 그룹 및 의뢰기관과 합

			의
	항목별 벤치마크테스트 방법 개발	1. 벤치마크테스트 절차 개발 2. 벤치마크테스트 방법(시나리오, 스크립트, 테스트케이스 등)개발	1. 벤치마크테스트 절차 개발 2. 벤치마크테스트 방법(시나리오, 스크립트, 테스트케이스 등) 개발
	시험환경 구축	1. 시험 환경 구축 2. 시험 도구 및 S/W 설치	1. 시험 환경 구축 2. 시험 도구 및 S/W 설치
실행	벤치마크테스트 수행	1. 벤치마크테스트 항목별 시험 진행	1. 벤치마크테스트 항목별 시험 진행
	검토	1. 문제점 정리 / 협의 2. 보완 시험 3. 결과 정리	1. 문제점 정리/협의 2. 보완 시험 3. 결과 정리
	보고서 작성	벤치마크 절차, 방법 및 비교분석 결과 정리	벤치마크 절차, 방법 정리 및 순위표 산출
마감	검토	벤치마크 결과 검토	벤치마크 결과 검토

▪ 계획단계

요구분석 단계에서 도출한 결과물들을 토대로, 일정 및 자원 투입 계획을 구체적으로 산정한다. 별도의 교육이 필요한 경우 교육계획을 수립하며, 전문가 그룹을 구성하도록 한다. 시험환경 구축을 위한 환경설정과 시험도구 및 S/W 선정도 계획단계에서 이루어진다.

▪ 설계 단계

계획 수립 이후 도메인별 전문가 그룹을 구성하여 시험 대상 주요 항목 및 부가항목을 선정한다. 벤치마크테스트 항목 확정 이후 항목별 측정기준 및 공식을 도출하여 메트릭을 작성하고, 벤치마크 테스트에 참여하는 업체들과 항목에 대한 합의를 진행한 후 벤치마크테스트 평가모델을 완성한다.

▪ 실행 단계

개발된 평가모델의 각 항목 별 벤치마크테스트를 수행한다. 비교 대상 제품별로 동일한 조건에서 시험을 실시하고 시험 결과가 도출되면 결과에 대한 문제점을 정일 검토 후, 필요한 경우 보완 시험을 수행하고 결과를 정리한다.

▪ 마감 단계

벤치마크테스트의 절차, 방법 및 비교분석 결과를 정리 검토하고 결과에 대한 고객 의견을 조사한다. 최종적으로 벤치마크테스트 과정 및 결과를 저장한다.

⊙ 순위도출 형 프로세스 고려사항

<표71>은 비교분석 형 벤치마크테스트와 순위도출 형의 벤치마크테스트의 프로세스를 비교한 도표이다. 도표에서 나타나듯이 일반적인 비교 분석형 벤치마크테스트와 순위도출 형 벤치마크테스트는 동일한 프로세스를 준수하지만 각 단계별로 약간의 차이를 갖는다.

▪ 요구분석 단계

비교분석 형에서 벤치마크테스트 항목 협의 주체는 벤치마크테스트에 응하는 업체, 벤치마크테스트 수행기관, 전문가집단이다. 항목을 도출하고 확정하는 단계에서 특정 제품 또는 특정 기능에 편중되지 않도록 모든 참여 업체와의 협의 조율 과정이 수반된다.

반면 순위도출 형에서의 항목 협의 주체는 벤치마크테스트 의뢰기관, 벤치마크테스트 수행기관, 전문가집단이다. 벤치마크테스트에 응하는 업체는 협의 주체 대상에 포함도리 수 없다. 항목을 도출하고 확정하는 단계에서 의뢰 기관이 요구사항을 정확히 파악하고 전문가 집단 및 수행기관과의 협의 조율 과정이 수반된다.

⊙ 설계 단계

설계단계의 최종 산출물은 벤치마크테스트 평가 모델이다. 최종 확정된 평가 항목들은 요구분석 단계와 마찬가지로 각각 참여 업체들과의 합의(비교분석 형), 의뢰 기관과의 합의(순위도출 형)를 필요로 한다.

<표71>의 순위도출 형 설계단계의 작업 영역을 살펴보면 평가모델 개발 시 부가적으로 고려해야 할 사항으로 항목별 가중치, 항목별 배점, 과락 항목에 대해 언급되어 있다. 벤치마크테스트 목적에 따라 평가모델 개발 요소가 크게 달라짐을 알 수 있으며, 각 사항의 내용에 대해서는 다음장에서 다루고 있다.

⊙ 마감 단계

벤치마크테스트 수행 종료 후 마감단계에서는 각각 벤치마크테스트 항목 별 비교 분석 결과와 순위 표(순위도출 형)를 산출하게 된다. 결과에 대해나 공표는 전자의 경우 각 참여업체들에게 공개되고 일반에 공표 여부는 또 다른 프로세스를 거쳐 공표 여부가 결정되게 된다. 하지만 후자의 경우 벤치마크테스트의 결과가 의뢰 기관에만 공개되며, 참여업체 및 일반에 공개하지 않는다.

3) 평가모델 개발 이슈

순위 도출을 위한 벤치마크테스트는 비교분석을 위한 벤치마크테스트와 근본적으로 목적이 다르기 때문에 의뢰 기관의 요구가 정확히 반영되고 평가모델 개발 시 순위를 도출 할 수 있는 별도의 요소가 추가되어야 한다. [표 1]의 설계단계를 살펴보면 순위도출 벤치마크테스트 작업 영역에 평가모델 개발 시 고려해야 할 요소로 항목별 가중치 부여, 항목별 배점 부여, 과락 항목 설정을 나타내고 있다.

본 장에서는 상기 언급한 순위도출 형 벤치마크테스트 평가모델 개발 시 고려해야 할 요소들에 대해 정의하고 각각의 적용 방업에 대해 자세히 설명하도록 한다.

⊙ 가중치 부여

가중치는 시험 대상 제품의 평가항목 중요도에 따라 차등적으로 부여해야 하며 벤치마크테스트 의뢰기관, 벤치마크테스트 수행기관, 전문위원 3자의 충분한 의견 수렴을 거친 후 결정되어야 한다.

세부항목을 기준으로 벤치마크테스트 항목의 중요도를 2단계(필수항목, 부가항목)로 구분하여 가중치를 부여하는 방법, 3단계(필수항목, 중요항목, 부가항목)로 구분하는 방법 또는 더욱 세밀하게 구분하여 가중치를 부여할 수 도 있다. 하지만 통상적으로 2 단계 또는 3 단계 구분이 일반적인 가중치 부여 방법으로 사용되고 있다.

항목의 중요도를 3단계로 구분할 경우 평가모델이 간결하고 명확해 지는 반면 정밀도는 보다 세밀한 구분방법 보다 덜하게 된다. 반면 3단계로 구분할 경우 보다 정밀한 평가가 가능하지만 제품의 순위에 변동을 줄 만큼의 차이를 발생시키는 일은 드물다. 선정하기 위한 벤치마크 테스트에서는 제품을 도입하려는 벤치마크테스트 의뢰기관에서 중요하게 여기는 필수항목에 통상 50% 이상의 배점을 할당하기 때문에 더욱 그러하다.

[표 2]와 [표 3]은 TTA에서 기 수행한 순위 선정을 위한 벤치마크테스트 사례 중 일부 데이터이며, 가중치 단계와 항목 수를 나타내고 있다.해당 벤치마크테스트에서는 총 40개의 세부항목에 대해2단계 가중치를4:1 비율로 부여하여 적용하였다.

<표72> 2단계 가중치 및 항목 수

	기능성	사용성	효율성	이식성	합계
필수항목	16	5	3	5	29
부가항목	5	2	2	2	11
합계	21	7	5	7	40

<표73> 3단계 가중치 및 항목 수

	기능성	사용성	효율성	이식성	합계
필수항목	13	3	3	3	22
중요항목	5	2	1	2	10
부가항목	3	2	1	2	8
합계	21	7	5	7	40

⊙ 배점 기준

▪ 하향식 및 상향식 배점 부여

품질특성, 분류, 평가항목, 세부항목, 시험항목, 테스트케이스의 6단계의 레벨로 구성된 평가모델을 예로 들면 품질특성별로 큰 점수를 부여한 후 하향식으로 점수를 배분하는 Top-Down 방식과, 테스트케이스마다 작은 점수를 부여하여 상향식으로 점수를 합산하는 Bottom-Up 방식이 존재한다. 언급한 두 가지 방식이 S/W 벤치마크테스트의 일반적인 배점방식이다. 상기 가중치 부여 결과를 그대로 점수화하여 다음 하위 레벨에 배점하는 경우가 Top-Down 방식에 해당할 수 있다. 이렇게 Top-Down 방식을 적용할 경우 최하위레벨의 테스트케이스들 간의 형평성이 결여될 수 있는 문제점이 발생한다. 이러한 문제점은 Bottom-Up 방식의 장점을 반영함으로써 어느 정도 해소할 수 있다.

Bottom-Up 방식을 적용할 경우에는 가중치 부여 시 구분했던 필수항목, 중요항목, 부가항목들 각각에 대해서는 항목 내부에서 동일한 배점을 적용해야 한다. 이렇게 함으로써 Top-Down 방식을 적용함으로써 발생하는 형평성 결여 문제점을 일부 해소 할 수 있다.

⊙ 기능구현여부와 기능구현 정확성

순위를 선정하는 벤치마크테스트는 기존 벤치마크테스트 보다 다양한 제품들이 벤치마크테스트에 참여한다. 이때 기능 평가 항목이 모든 제품을

수용하기는 현실적으로 불가능하다.

어떠한 항목에 대해 해당 제품의 모호한 구현은 Pass/Fail 을 판단하는데 어려움이 있다.

따라서 명확한 기준을 마련하기 위해 기능성에 속한 벤치마크테스트 항목은 실제 해당 기능이 구현되어 있는지를 시험하는 “기능구현”과 구현된 기능이 의도한데로 정상 동작 하는지 여부를 시험하는“정상 동작 하는지 여부를 시험하는”정확성“으로 구분하여 배점한다.

이렇게 구분된 배점을 통해 모호하게 구현된 기능에 해해 부분점수를 부여할 수 있으며, 부분 점수의 적용을 통해 벤치마크테스트를 수행하면서 발생할 수도 있는 오 측정의 역효과를 부분적으로 감소시킬 수 있다.

⊙ 과락 항목 설정

구매를 위한 우선순위 도출을 목적으로 하는 벤치마크테스트 수행 시 대상 제품은 시험수행 기관 또는 의뢰기관에서 선정하는 것이 아니라 신청에 의해 접수된 제품을 대상으로 벤치마크테스트를 수행하게 된다. 이러한 경우 목적에 부합하지 않는 제품들도 벤치마크테스트에 신청하는 경우도 있다.

과락 항목의 범위는 벤치마크테스트 의뢰자가 ‘반드시 구현되어야 하며 정상적인 동작을 보장해야 한’다고 여기는 항목과,간단한 예로 “제품은 정상적으로 설치 가능해야한다” 와 같은 시험 진행을 위한 필수 기본 항목으로 구성한다.

과락 항목 채택 여부는 실질적으로 벤치마크테스트 의뢰자와 협의 하에 채택 하여야 하며, 이러한 과락 항목에 대해서는 모집공고 시 명확하게 공시하여야만 한다.

과락 항목에 대한 모집공고를 명확하게 공시하여도 해당 요구사항을 만족하지 못하는 제품들이 실제 벤치마크테스트에 응하는 사례가 다수 존재하며 현재까지 TTA에서 수행한 순위 선정을 위한 벤치마크테스트 참여 업체 중 37.5% 가량이 해당 케이스로써 실격한 전례를 보인다.일반적으로 과락 항목은 배점을 부여하지 않으며Pass/Fail 형태의 특수 항목으로 분류한다.

4) 결론 및 향후 연구

본 논문에서는 각 제품의 기능별 성능별 비교분석을 통해 제품의 장단점을 파악하는 것을 목적으로 하는 기존의 벤치마트테스트 외에 최근 수효가 늘고 있는 순위 도출을 목적으로 하는 벤치마크테스트에 대해 연구하고, 각 단계별로 주요 점검 요소를 도표로 정리하여 나타내었으며, 세부적으로 상이한 요소에 대해서 자세한 설명을 통해 해당 벤치마크테스트 프로세스 및 평가 모델 개발과 관련한 지침을 제시하였다.

향후 보다 다양한 가중치 부여 방법 개발에 관한 연구와, 보다 공정한 배점기준에 환한 연구를 지속적으로 진행하여 평가모델에 반영하여야 하며, 이러한 연구를 통해 벤치마트테스트 목적에 부합하는 적합한 평가모델 개발이 가능할 것이라 예상된다.

1.2.2. 일본 IPA의 공개소프트웨어 성능 평가

가. 일본의 IPA 소개[IPAWEB]

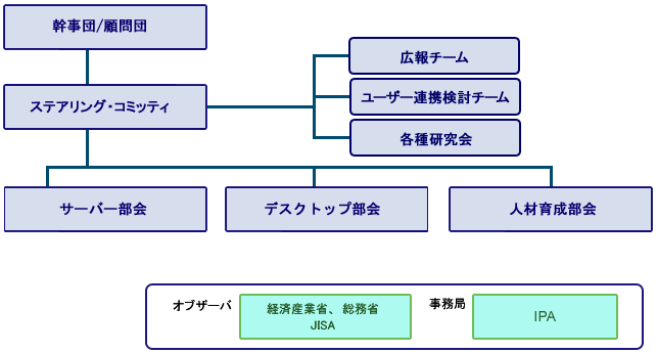
Linux(을)를 시작으로 하는 오픈 소스 소프트웨어(OSS)에 대해, 기업·대학 등의 정보시스템으로의 도입이나 편성시스템으로의 활용이 진전되고 있지만, 사용자가 안심하고 사용하기 위한 기술상·제도상의 과제가

문제가 되고 있다. 일본OSS추진 포럼(대표 간사 쿠와바라 히로시 : ㈜히타치 제작소 특별 고문)으로써 새로운 소프트웨어 패러다임(paradigm)를 활용하는 것에 의해서 「독점에 의한 폐해의 배제와 선택사항의 확대」, 「기술 혁신의 촉진」, 「인재육성」을 실현해, 일본의 경쟁력 강화를 도모하는 것이 설립 목적이다. 기업간의 경쟁과 협조를 본연의 자세에 유의하면서, 커뮤니티와의 제휴등을 통해 OSS보급에 공헌하고 있다.

또, 본 포럼은 중국 OSS 추진 연맹, 한국 OSS 추진 포럼과 함께, 북동아시아 OSS 추진 포럼을 구성해, 중국·한국의 민간기업·연구 교육 기관과 제휴 체제를 쌓아 올리고 있다. 한.중.일의 정부 차원의 제휴로 상호 협조하고 있다.

일본 OSS추진 포럼에서는, 일본의 정보 시스템의 사용자, 벤더, 학문적인경험이 있는자와 실무에서의 유식자가 뭉쳐, OSS의 활용상의 문제에 대해서, 자유로운 입장에서의 논의, 과제 해결을 하고 있다.

본 포럼에서는 각 기업의 최고 권위자로 구성되는 간사단 및 고문단이 중심이 되고, OSS의 검토를 계속하고 있다. 또, 간사단 및 고문단을 지원하는 스테어 링코밋티가 설치되어 있다. 또한 서버, 데스크탑, 인재육성의 3개의 부회로 구성하여, 분야별 활동을 진행하고 있다.



<그림64> 일본OSS추진 포럼 체제도

1) 서버 부회

본부회는, 일본의 IT산업의 발전과 IT시장에 있어서의 공정한 경쟁, 및 IT사용자의 요구를 만족하는 복수의 선택사항을 가질 수 있는 것을 목표로 해, OSS를 엔터프라이즈 분야의 서버로서 활용하는 경우에 필요한 여러가지 문제의 해결을 도모하는 것으로, 일본 국내에서의 OSS의 보급, 확대를 목표로 하는 것이다.

본부회는, 상기의 목적을 달성하기 위해서, 이하의 활동을 실시한다.

⊙ 엔터프라이즈 분야에 있어서의 OSS 비즈니스에 대해서, 연구조사·문제 분석을 실시해, 비즈니스 모델을 제언한다. 또, 비즈니스에 부흥하기 위한 장기 지원에 대해서, OSS의 특징을 근거로 한 문제 분석을 실시해, 모델을 제언한다.

⊙ 엔터프라이즈 분야에 있어서의 OSS의 성능·신뢰성 평가에 대해서, 그 기술면의 상황 조사, 사용면으로부터의 문제 분석을 실시해, 기술개발의 방향성을 제언하는 것과 동시에, 필요한 툴 개발을 기획한다.

⊙ 엔터프라이즈 분야에 있어서의 OSS 시스템의 이익을 위해서, 미션 크리티컬 용도에의 새로운 전개를 예측한 Road Map을 제안하는 것과 동시에,OA서버 용도에의 적용을 검토한다.

⊙ 정부, 지방공공단체로의 OSS 시스템 이용에 대해서, 연구조사·문제분석을 실시해, 정부 조달에 대해 OSS 시스템이 원활히 유지될 수 있도록 관계 기관에 제언을 실시해, OSS의 보급, 확대를 도모한다.

⊙ 북동 아시아 OSS 추진 포럼의 WG1에 대응해, 서버에 관한 일본측 조직으로서 활동한다.

⊙ 3년 후에는 시장의 톱 쉐어를 목표로 한다

⊙ 미션 크리티컬 시스템 분야에의 적용 영역 확대하여 서버 부회를 설립해 구체적인 기술 이정표를 설정하고 회원 기업이 목표실현을

하도록 지원한다.

2) 데스크탑 부회

본부회는, 일본의 IT산업의 발전과 IT시장에 있어서의 공정한 경쟁, 및 IT이용자의 요구를 만족하는 것을 가질 수 있는 것을 목표로 해, 일본에 있어서의 OSS 데스크 탑 환경의 보급의 촉진, 및 보급을 방해하는 여러가지 문제의 해결을 도모하는 것으로, 일본 국내에 OSS의 보급, 확대를 목표로 하는 것이다.

⊙ 데스크탑 분야에서의 OSS의 보급은, 서버 분야와 비교해서 늦다. 그 요인을 분석해, OSS 데스크탑이 도입시의 선택사항이라고 인정되어 OSS 데스크탑에의 이행이 촉진되는 것을 목표로 한다.
⊙ 연구적인 과제에 대해서는, 해결을 향해 교육기관과도 제휴해, 산학관으로 임하기 위한 구조를 검토한다.
⊙ 데스크탑 영역에 있어서의 OSS 이용 촉진을 목적으로 해, 그에 필요한 기능이나 장해를 밝혀내, 그 해결책을 모색하는 TF를 설립해, 실시단계에서 TF를 해체 한다.
⊙ 금년도의 최대의 목표는, 작년도 과제에서 추출된 테스크 포스의 조사의 결과, 판명된 OSS데스크탑의 보급 저해 요인에 대해서의 해결책을 모색해, 저해 요인을 해결하고 대처한다.

3) 북동 아시아 OSS 추진 포럼

일본 OSS 추진 포럼은, 중국 OSS 추진 연맹, 한국 OSS 추진 포럼과 함께, 북동 아시아 OSS 추진 포럼을 구성하여, 중국·한국의 민간기업·연구 교육 기관과 제휴 체제를 만들고 있다.

일·중·한국의 정부 차원의 제휴로 협조하고 있다.

<그림65> 북동 아시아 OSS 추진 포럼 조직도

⊙ WG1

WG1(은)는, 일중한국으로의OSS에 관한 기술개발, 및 평가를 목적으로 한 워킹 그룹. 북동 아시아OSS추진 포럼으로서OSS에 관한 여러가지 활동을 추진해 가려면 , 우선 각국간에서, 기술적인 과제에 대한 공통 인식을 얻음.

각국에서의 OSS에 관한 기술적인 평가 활동이나 개발 프로젝트의 내용을 공유해, 장래적으로는, 그러한 과제를 공동으로 해결하는 것을 목표로 하고 있다.

⊙ WG2

2004년7달에 삿포로에서 행해진 제2회 북동 아시아 OSS추진 포럼에 대하고, 동포럼에 인재육성에 관한 WG2의 설치가 결정되었다.

또,2004년12달에 서울에서 행해진 제3회 북동 아시아 OSS추진 포럼에 대하고,WG2(인재육성)에 붙어 이하의 의장 성명이 발표되었다.

▪ 테스크 포스1 : 각국에 있어서의 OSS인재육성/교육의 현상이나 과제를

조사/공개해, 북동 아시아 레벨로의 과제를 논의한다.

- 테스크 포스2 : 제4회 북동 아시아OSS추진 포럼에의 준비로서OSS콘테스트의 계획을 한다.

⊙ WG3 : Study on Standardization & Certificarion

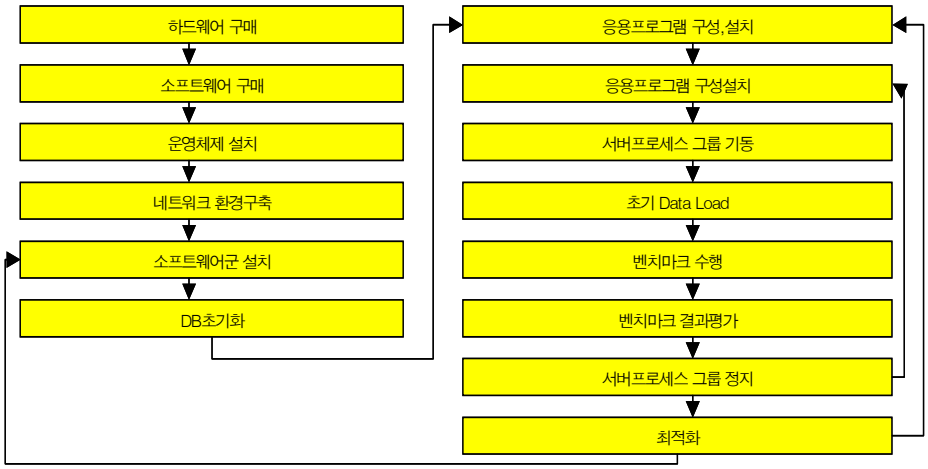
- 조직: 본 WG은,2004년7월28일에 삿포로에서 개최된 제2회북동 아시아 OSS추진 포럼에 의해서 조직 되어 일중한국IT국장 회의에 의해서 승인되었다.
- 목적: 일중한국 및 세계 각국에 있어서의OSS와 관련한다. IT 시스템의 호환성 유지 때문에, 이 WG은, OSS에 관련하는 표준과 인증에 관한 일중한국의 요청을 연구해, 필요하게 응해 북동 아시아OSS 추진 포럼과/ 또는 관련하는 단체에 그러한 활동을 위해서 제언과 제안, 및 기술 보고서를 작성한다.
- 활동 분야: 호환성 유지, 공공IT 시스템이 「OSS」의 선택을 확대하기 위해서, 기술 요구와 인증의 방법에 대해 논의한다.
- 일중한국의 공통 요구 사항과 기존의 국제 표준 및 국제 디펙트 표준과의 차이를 명확하게 한다.
- ISO/IEC JTC1 등의 국제 표준화 단체나 FSG 등의 국제 디펙트 표준화 단체와의 관계 구축과 관계 강화를 행한다.
- 다양한 OSS 관련의 표준 활동, 예를 들어 교육 활동, 홍보 광고, 공개 발표등을 활성화 한다.

나. 일본의 IPA의 공개소프트웨어 성능 평가[김두연06a]

일본의 IPA(INFORMATION-TECHNOLOGY PROMOTION AGENCY)에서 공개소프트웨어 대한 성능과 신뢰성 평가하여 공개하고 있다. 주요 평가 내용을 살펴보면 여러 Linux 버전들 간, 공개소프트웨어 기반 위에서 다수의 AP 서버를 이용한 분산 처리 환경, 상용AP 서버와 공개소프트웨어 AP 서버 간 등의 성능 및 신뢰성을 평가하여 그 절차와

평가 결과를 공개하고 있다. <그림66>은 일본 IPA BMT 절차를 나타낸 것이다. 하드웨어 구매부터 최적화까지 모두 14개 절차를 제시하고 있다.

BMT를 위한 사전준비로 각종 테스트 시스템 환경구축과 벤치마크를 수행하고 그 결과를 평가하여 만족스러운 결과를 얻을 때까지 최적화를 반복하는 과정을 나타내고 있어 본 연구에 시사하는 바가 크다.



<그림66> 일본IPA BMT 절차

그러나 공개소프트웨어 전후 단계의 과정이 없고 단순히 특정 시스템 소프트웨어 환경에서의 성능 평가 결과만을 저장하여 공개하고 있다. 공개소프트웨어 기반의 응용시스템은 시스템마다 특성이 있다. 따라서 시스템마다 요구하는 성능조건이 다르다. IPA의 공개소프트웨어의 성능평가는 다양한 응용시스템을 고려하지 않았다는데 한계가 있다.

다. 일본의 IPA의 OSS 성능·신뢰성 평가사례 [IPAWEB]

일본의 IPA는 2005 년도 상반기 오픈 소스 소프트웨어 활용 기반 정비 사업으로 OS단계의 OSS의 성능·신뢰성 평가를 하였는데 평가 아래와 같은 평가 방법에 의해 실시를 하였다. 개발 기반 WG의 활동 목적은 「서버Linux, OSS의 한층 더 보급·확대를 위한 벤더 사이의 과제 해결」이다.

OSS에 관한 노하우를 오픈화해, 안전한 OSS를 사용할 수 있는 환경 만들기를 목표로 하고 있다. OSS의 실시스템에의 적용이, Linux뿐만이 아니고, 미들웨어에까지 확대하고 있는 것으로, OSS(을)를 적용한 시스템이 복잡화되고 있다. 그럼에도 불구하고,OSS그리고 비즈니스를 전개하는 벤더 사이에서는, 성능·신뢰성 등의 시스템 설계·구축에 필요한 데이터가 부족하고 있어, 결과적으로, 각자가 같은 평가를 실시하고 있다. 장해 해석 툴이 부족하고 있어, 원인 규명에 시간이 걸린다 하고 있다.

- ⊙ 벤더 공동의 OSS의 성능·신뢰성 평가에 의해, 시스템 설계·구축 노하우를 공유한다. 성능 평가에서는, 결과뿐만이 아니라, 순서나 툴·데이터도 공유해, 공통화를 도모한다. 순서나 툴은 넓게 커뮤니티에 공개해,OSS의 보급에 공헌한다. 벤더에 있어서의 평가 코스트를 절감 한다. 최종적으로는, 다양한 노하우를 베이스로 한 시스템 구축에 의해, 시스템 전체의 신뢰성 향상에 연결한다.
- ⊙ 장해 해석 툴을 개발해, 노하우를 공유한다. 상용OS의 장해 해석으로 당연한 듯이 이용되는 덤프나 트레이스라고 하는 툴은, Linux의 현장에서 대부분 사용되지 않지만, 이를 이용하는 노하우를 벤더간에 공유한다. 필요한 기능(부족한 기능)은, WG에서 개발해서 공개한다. 이러한 대응에 의해 장해 해석에 걸리는 시간을 단축해, 미션 크리티컬 시스템에의 Linux적용 요구에 대응한다.

1) OS단계의 OSS 성능·신뢰성 평가사례 [IPA05a]

⊙ 퍼포먼스 감시 툴의 신뢰성 평가

OS의 퍼포먼스 감시 툴로서 sar, iostat, mpstat 라고 하는 3 개의 커멘드의 신뢰성 평가를 실시하고 특히 고부하시에 있어서의 이러한 툴의 거동을 조사해 데이터가 지정한 간격으로 취득할 수 없다고 하는 기존의 문제에 대해서 몇 개의 회피책을 제시한다.

▪ 툴의 개요

sar, iostat, mpstat 는 일정 마다 커넬이 제공하는 여러가지 시스템 통계 정보를 취득 것에 따라, 시스템의 부하 상황을 모니터링 할 수 있는 커멘드이다.이런 커멘드는 Linux 시스템에서는 sysstat 라고 하는 패키지에 포함되어 많은 Linux 디스트리뷰션에서는 표준으로 제공되고 있는 일반적인 툴이다.

- sar 는 CPU 이용률, 네트워크, 디스크 I/O 및 페이징 등의 다방면에 걸쳐 시스템 통계 정보를 일정 간격 마다 취득하는 커멘드이다. 취득한 통계 정보는 화면에 출력되고,"-o"옵션을 사용하는 것으로써 지정한 파일에 바이너리 형식으로 보존할 수도 있다. 바이너리 형식으로 보존했을 경우, 통계 정보를 확인하기 위해 텍스트 형식으로 변환·출력되어진 것으로부터, 필요한 정보를 유연하게 취득할 수 있다.

-iostat는 일정기간 마다의 디스크의 전송 회수나 전송량, 입출력 리퀘스트 큐의 길이 등, 디바이스나 파티션마다의 입출력의 정보를 취득하는 커멘드이다(sar에서도 입출력의 정보는 얻을 수 있지만, 시스템 전체의 정보 밖에 얻을 수 없다).

-mpstat은 일정기간마다 CPU에 관한 통계 정보(CPU 이용률, 우선순위 변동 발생 회수 등)를 취득하는 툴이다. iostat 커멘드와 같이, 취득된 결과는 표준 출력에 텍스트 형식으로 출력되어 데이터를 파일에 보존하기 위해서는 파일 리디렉트를 사용할

필요가 있다.

⊙ 평가 항목

퍼포먼스 감시 툴의 평가로서 3 종류의 대표적인 부하 타입(디스크 I/O, CPU, 메모리)에 대해서 지정한 시간 간격으로 시스템의 통계 정보를 어느 정도 취득할 수 있는지를 평가한다.

⊙ 평가 순서

퍼포먼스 툴의 평가 순서는 아래와 같다.

- 퍼포먼스 감시 툴을 기동
- (1)의 10 초 후에 부하 생성을 개시
- (2)의 부하 생성 종료의 10 초 후에 퍼포먼스 감시 툴을 종료

측정에서는 이상의 순서를 자동화한 스크립트를 사용했다. 또, 실제의 측정에 대하고는 같은 부하량, 모니터링 툴에 대해서 3 회 측정을 실시하고 데이터 취득 간격의 분포를 그래프화하고 3회의 측정에 부하가 발생하고 있는 기간에 있어서의 데이터 취득 간격을 집계한다.

⊙ 평가 결과

우선, 부하가 비교적 높지 않은 상황으로 sar, iostat, mpstat 로 모니터링을 실시해, 지정한 시간 간격으로 데이터를 취득할 수 있는지, 부하 정보를 올바르게 취득할 수 있는지를 확인하고 stress 프로세스로 CPU 부하를 걸쳤을 때의 sar 의 거동에 대해서, 지정한 간격으로 데이터를 취득할 수 있는지, 부하 상황을 올바르게 취득할 수 있는지를 확인한다.

다음에 디스크 I/O 부하를 걸었을 때의 iostat 의 거동에 대해서 지정 간격으로 데이터를 취득할 수 있는지, 부하 상황을 올바르게 취득할 수 있는지를 확인한다.

다음에, stress 프로세스로 메모리 부하를 걸었을 때의 mpstat 의 거동에 대해서, 지정 사이 간격으로 데이터를 취득할 수 있는지, 올바른 부하 상황을 취득할 수 있는지 확인한다.

2) DB단계의 OSS 성능·신뢰성 평가사례[IPA05b]

오픈 소스 RDBMS를 기업 시스템 등으로 이용하는데 있어서 중요한 평가 방법의 정비를 실시하고 데이타베이스는 기업 시스템 중에서도 데이터를 통괄적으로 관리하는 것이 중요시 되며, 많은 기업용 시스템에서도 데이타베이스가 일반적으로 이용되고 있다.

오픈 소스 제품에 대해도 OS레벨로부터 미들웨어 레벨에 주목이 옮기고 있고, 특히 오픈 소스 RDBMS에의 관심이 높아지고 있다.

이러한 중, 시스템의 개발 국면에 있고, RDBMS를 선택하는 경우의 판단 기준으로서의 성능 평가 수법의 요구가 높아지고 있어 오픈 소스 RDBMS의 적정한 이용 범위를 판별하는데 있어서, 표준적인 평가 순서를 가지는 의의는 크다고 생각할 수 있다.

이것에 응하기 위해서 본 평가에서는, 보다 범용적인 툴로서 활용할 수 있도록 , 시스템 개발에 대해 필요한 RDBMS의 성능 측정을 위한 환경 구축·평가 순서를 책정해, RDBMS의 sizing나 튜닝에 있어서의 표준적인 순서로서 이용할 수 있다는 것을 검증한다. 또, 최근 주목을 받고 있는, IA32 아키텍쳐의 64 bit 확장판인 CPU를 사용해, 32 bit와 64 bit의 차이도 맞추어 검증한다. 또, 이 순서를 이용해 주요한 오픈 소스 RDBMS에 관한 성능 측정을 실시해, 참고가격으로서 결과를 나타낸다.

본 평가에서는 데이타베이스의 작업의 부담량 툴로서 OSDL의 DBT 시리즈를 이용했다. DBT는 오픈 소스 라이센스로 제공되며 염가로 간편하게 RDBMS의 평가 툴로서 이용할 수 있다. 툴의 선정에 관해서, 주요한

조건은 이하와 같다.

⊙ DB벤치마크에 이용할 수 있는 툴로서의 기능(실환경에 가까운 성능 평가 툴인 것)
⊙ 환경 구축의 손쉬움(저비용, 환경 구축에 시간이 들지 않는 것)
⊙ 이용시의 제약(이용시의 제한이 적은 것, 코스트가 들지 않는 것)
⊙ 결과의 공개에 대한 제약(공개의 제한이 적은 것)
⊙ 범용성(오픈 소스 RDBMS에의 대응, 다양한 규모로 이용할 수 있는 것)

DBT는 TPC의 간이판 성능 평가를 위한 작업의 부담량 툴로서 오픈 소스로 제공되고 있어 무상으로 이용할 수 있어 결과의 공개 제한이 적고, 본 평가로 이용하는 것으로서 타당이라고 판단한다.

덧붙여 본 평가에서는 성능 평가 순서를 작성해, 이 순서를 이용하는데 있어서의 유의점을 나타내는 일에 의해, 오픈 소스 RDBMS를 보다 유효하게 활용하기 위한 지침이 되는 일을 목적으로 하고 있어 개개의 하드웨어, 소프트웨어의 개별 제품의 성능 비교를 실시하는 일을 목적으로 하지 않았다.

3) OSS Middleware의 성능·신뢰성 평가사례[IPAc05]

일본의 IPA에서는 JBoss의 성능·신뢰성 평가 순서를 확립확하고 JBoss를 어디까지 사용할 수 있는지를 판별하고 성능 한계 부근에서의 JBoss의 거동을 관찰하고 하여 커넬 2.6은 2.4에 비해 얼마나 좋아졌는지와 동일 하드웨어·소프트웨어 환경상에서, OS만을 변화시키면서 성능 한계 부근에서의 상태를 측정하였다.

벤치마크 측정 후에 얻을 수 있는 결과는 아래와 같다.

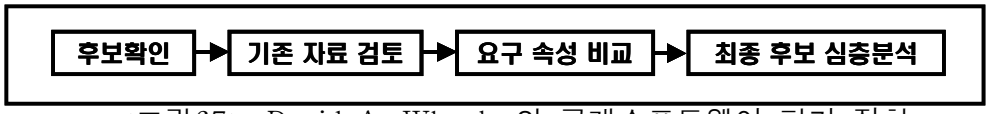
⊙ 처리 성능(종합, 트랜잭션(transaction) 종별마다)

⊙ 단위시간의 트랜잭션(transaction) 처리 성능(JOPS)
JOPS = jAppServerOperation Per Second
⊙ 응답 시간(트랜잭션(transaction) 종별마다)
⊙ 평균 응답 시간, 최대 응답 시간,90% 응답 시간등
⊙ 여러 가지의 테스트 결과(약 30의 테스트 항목)
⊙ 트랜잭션(transaction) 정합성, 응답 시간이 요구 범위내인지
⊙ 에러 로그, 트랜잭션(transaction) 상세 데이터
⊙ 측정 결과의 공개에 대한 제한
⊙ IR 및 JOPS치의 공개에는, SPEC.org의 리뷰가 필요
IR = Injection Rate(서버에의 부하 및 DB의 규모를 나타낸다)

http://www.spec.org/jAppServer2004/results/ 에 순서·결과를 공개하여 사용자들이 판단할수 있는 자료를 제공을 하였다. 다만 현재 상태로서는 상용 AP서버에만 적용하였다.

1.2.3. David A. Wheeler의 공개소프트웨어 평가 방법 [DAV06]

David A. Wheeler는 공개소프트웨어를 평가하는 절차로 공개소프트웨어 후보 확인, 기존 자료 검토, 요구 기본 속성 비교, 최종 후보 심층 분석 등 4단계를 제시하고 있다. 이 중 3단계인 요구 속성 비교를 위하여 기능, 비용, 신뢰성, 성능, 저작권 등 13가지 속성을 제시하였다.



<그림67> David A. Wheeler의 공개소프트웨어 평가 절차

이 연구는 공개소프트웨어 선정을 위한 절차와 평가 방법을 제시하고 있으나 일반적인 방향성을 제시하는 정도에 그치고 있다.

한편, 메사츄세츠 기술연구소의 Michael P. Voightman은 공개소프트웨어의 특징인 1) 광범위한 전문가 평가, 2) 빠르고 빈번한 코드배포, 3) 우수한 교육 기반 등을 평가 및 활용하여 소프트웨어를 개선하고자 하였다. 그는 공개소프트웨어의 특징을 살려 민감한 핵심 업무도 적용할 수 있다고 보았다. 그러나 장애 발생 시 이에 대한 책임 문제, 신속한 유지보수 문제 등에 대해서 언급 없이 적용한다는 것은 모든 책임과 유지보수 부담이 업무 운영자에 전적으로 전가되어 도입을 꺼린다는 것을 간과하였다.

가. OSS/FS 평가 방법 소개

Open Source Software / Free Software (OSS/FS) 는 커다란 이슈로 성장하였다. 명백히, OSS/FS 프로그램은 저작권을 어떠한 목적의 프로그램 실행하기 위한 자유를 유저에게 제공한 것이다. 이것은 연구, 수정, 재배포에 이전 개발자에게 어떠한 저작료도 지불하지 않는다. OSS/FS 프로그램을 사용하는 많은 양질의 연구는 저작권을 가진 프로그램과 비교하여도 충분한 가능성과 이유를 가짐을 보이고 있다. OSS/FS 프로그램은 FLOSS, FOSS, libre 프로그램이라고도 불린다. OSS/FS 프로그램은 비상용적이지만, 많은 사용적인 곳에 사용되고 있지만, 이것 역시 OSS/FS의 비상용성에 영향을 미치지 않는다.

이글은 상용 프로그램에서는 이미 알고 있을 수도 있지만 어떻게

OSS/FS 프로그램을 포함한 프로그램을 평가하는 가를 설명하고 있다. 소프트웨어 개발에 관련된 중요한 기술은 여기에 설명된 대부분의 프로세스에 필요하지 않다. 그러나 몇몇 단계에서는 소프트웨어 개발에 대한 지식을 필요로 한다.

OSS/FS를 평가하는 많은 다른 접근 방법이 있다. "Open Source Maturity Model"이라고 불리는 커다란 2가지 평가 절차가 있는데, 하나는 Navica/Golden Open Source Maturity Model이고 남은 하나는 CapGemini Open Source Maturity Model이다. Business Readiness Rating(BRR) 모델은 OSS/FS 성숙도를 rating하기 위한 모델이다. 이 모델에 대한 것은 참고문헌에서 찾길 바란다. Top Tips for Selecting Open Source Software은 선택을 위한 팁이다. Karin van den Berg's "Finding Open options: An Open Source software evaluation model with a case study on Course Management System"도 36개의 시스템을 평가하는 예시로써 OSS/FS 프로그램을 평가하는 접근법을 제공한다. 흥미로운 것은 van dem Berg는 5개의 다른 평가 방법을 조사하고, 평가 프로세스를 van dem Berg가 발견한 주요 기준에 부합되는 것은 1~2개였다. 특정한 영역에 대해 제시된 어떠한 문서는 그들의 작업 대부분이 Choosing Open source, A Guide to Civil Society Organizations by Mark Surman and Jason Diceman(2004/1/6)등의 것과 비슷하다. 물론 특정한 어플리케이션 영역에 대한 조사와 추적을 한 사이트도 있다. InfoWorld's Special Report on "Build your business with open source"는 비즈니스를 위한 후보 어클리케이션을 묶어 놓기도 하였다.

모든 프로그램을 평가하기 위한 기본 단계는 동일하다. 다음과 같은 단계를 제시한다. 후보 식별(identity), 존재하는 리뷰를 읽음, 당신이 필요한 프로그램에 속성에 비교, 그리고 상위 후보의 철저한 분석이다.

그러나 평가를 위한 이러한 단계를 수행하는 방법은 상용 프로그램과

OSS/FS 프로그램이 다르다. 주요한 다른 점은 OSS/FS 프로그램을 위한 정보의 가능성(information available)과 상용 프로그램의 정보의 가능성이 다르다는 점이다. 대부분의 OSS/FS 프로그램은 정보를 공유하는 것을 중시 여기나 상용프로그램은 아니기 때문이다, 프로그램의 소스코드, 프로그램의 설계, 분석, 개발자들 사이에 미래의 방향과 디자인에 대한 논의, 유저와 개발사 사이에 어떻게 하면 잘 사용할 수 있는 지에 대한 논의 등이 정보에 포함된다. 그러나 가장 중요한 것은 OSS/FS 프로그램은 사용자가 변경, 재배포가 가능하다는 것이다. 이러한 다른 점은 많은 점(유연성/customizability, 옵션의 지원, 비용)에 영향을 미친다.

만약 상용프로그램을 OSS/FS 프로그램과 비교한다면, 당신은 여기에 설명된 일반적인 접근법을 사용할 수 있다. 상용프로그램을 평가하는 이미 알고 있는 방법을 세부 평가 절차에 사용할 수 있다. 그리고 여기에 설명된 OSS/FS 방법을 세부 평가 절차에 사용할 수도 있다.

상용프로그램은 보기, 수정, 재배포에 대한 권리를 주지 않는다. 어떠한 조직에서는 OSS/FS 프로그램을 사용하기로 결정할 수 있다. 심지어 이러한 경우에도 여전히 OSS/FS 프로그램을 평가할 수 있을 필요가 있다. 왜냐하면 수많은 OSS/FS 프로그램 중에서 요구에 적합한 프로그램을 찾는 방법을 알아야 하기 때문이다.

소프트웨어를 평가하는데 보낸 노력의 많은 부분은 소프트웨어에 얼마나 중요하고 얼마나 영향을 주는지를 판단하는 것이다. 전체 평가 프로세스는 작은 프로그램의 경우 5분 정도이지만, 커다란 프로그램의 경우 수개월이 걸릴 수 있다. 그러나 일반적인 절차는 동일하다.

제품을 비교하는 것은 유연해 지도록 노력해야한다. 제품은 80% 정도는 찾는 것과 부합하지만 100% 부합하기는 힘들다. 그러나 만약 원하는 것이 무엇인지도 모른다면, 이러한 과정들을 이해할 수 없을 것이다.

나. 후보 식별

첫 번째 단계는 선택한 것을 찾는 것이다. 다음과 같은 방법을 사용할 수 있다.

명백한 방법은 친구나 동료에게 물어보는 것이다. 물론 찾는 것을 가지거나 사용하고 있는 자에게 물어보는 것이 가장 좋다. 만약 그들이 그것에 대한 경험을 가지고 있다면 추가적으로 질문을 하라, 다음 단계인 리뷰에서 유용하게 사용될 것이다.

"generally recognized as mature"(GRAM), "generally recognized as safe" (GRAS) 리스트를 보면 OSS/FS 치명적인 실수가 없는 또는 적은 잘 알려진 것들이다, 예를 들어, 누군가 웹서버가 필요한 누구도 Apache 정도의 실수를 가지고 치명적인 실수라고 생각하진 않을 것이다.

리스트에는 다음과 같을 것이다.

1) GRAM 리스트
2) IDA Open Source Migration Guidelines
3) The Table of equivalents / replacements / analogs of Window software in Linux

검색을 사용한 방법도 있다.

1) Freshmeat, Icewalkers, "Free Software Directory", 윈도우의 환경에서 작동하는 OSS/FS는 OSSwin, OpenCD, Koders.com
2) SourceForge, Savannah
3) Alltheweb, AltaVista, Teoma
4) Linux BSD

5) Debian

컴포넌트나 프로그램을 찾는 다음과 같은 팁도 있다.

1) 구글과 같은 검색 엔진에서 찾는 방법을 피하라
2) 특이하고 다양한 방법으로 검색을 하라
3) 이미 알려진 제품이 있는 경우 compete, competitor, compatible 이라는 호환 용어를 사용해라
4) 다양한 제품을 알고 있는 경우 이들을 조합해서 사용해라.
5) to->2와 같은 것을 고려하라

위의 모든 방법이 실패한다면 다른 사람에게 물어봐라, 유사한 관련된 프로그램을 찾기 위해 그들의 mailing list를 보는 것도 좋다. 검색엔진에 물어보는 방법도 있다. 물론 다른 사람을 고용해서 찾는 방법도 있다.

다. 존재하는 리뷰 읽기

식별의 단계가 끝나면, 존재하는 리뷰를 읽어야 한다. 이것은 프로그램의 장점과 단점을 빠르게 배우는 방법이다.

리뷰를 찾는 단순한 방법은 검색엔진을 사용하는 것이다. 찾은 후보들의 이름이 들어간 기사를 찾는 것이다. 물론 찾은 곳에서 찾아도 된다. 또는 이러한 것들을 발행하는 발간물을 찾아도 된다. 이 절차는 초기에 후보를 거르는 역할을 해줄 것이다.

모든 리뷰의 리스트를 찾을 수 없다. 이것은 끝없는 일이기 때문이다. 그러나 OSS/FS 프로그램의 경우 다음과 같은 방법으로 찾을 수 있다.

1) OSS/FS에는 많은 Content Management System(CMSs)가 있다.

"Content management problems and Open Source Solutions"라는 곳에는 수많은 리뷰가 있다.

2) OSS/FS Software Configuration Management(SCM) 프로그램도 많이 존재한다. Comments on Open Source Software/Free Software (OSS/FS) Software Configuration Management (SCM) System이 그 예이다.

이 두 영역은 많은 사람들의 협력을 통해 만들어진 것이다. OSS/FS 프로젝트는 많은 사람의 협력이 필요하며, 이것은 OSS/FS의 커다란 자산인 것이다.

많은 평가 방법에도 치우침 현상이 있으며, 이것은 환경에 특화된 것이 아님을 기억해라. 대부분의 잡지는 광고를 기반으로 한다. 예를 들어 comment의 경우 많은 사람이 comment를 달았다고 좋은 것도 아니며, 특정 사람만이 comment를 단 경우도 있다는 것이다. 또한 좋다고 여기어지는 OSS/FS도 일부 사람에게는 비판적일 수도 있다. 중요한 것은 정보를 얻는 것 이외에 다른 것이 더 많다는 것을 기억해야 한다.

리뷰의 중요한 점은 제품의 인기(공유)이다. 일반적으로 많은 인기 있는 제품은 사용하면 좋다. 커다란 시장을 가진 제품은 일반적으로 많은 요구를 충족시킬 것이며, 지원과 협동이 쉬운 편이다. OSS/FS 프로젝트는 많은 사람들(유저, 개발자, 사용자 등등)을 거치게 된다. 개발자 누구도 자신이 만든 것이 버려지길 원하지 않기 때문에 커다란 시장을 가진 것이 성공할 확률이 높은 것이다. 이와 반대로 빠르게 버려지는 시장은 커다란 리스크를 가지고 있다. 버려지는 데에는 이유가 있는 것이다.

시장 공유는 대부분의 OSS/FS 제품의 측정을 극도로 어렵게 한다. 누군가는 다운 받고 설치는 하지만 등록은 하지 않기 때문이다. 그러나 시장

공유 데이터는 어떠한 공용 제품(운영체제나 웹 브라운저) 같은 곳에 유용하다. 특히 인터넷 서비스의 보급으로 이러한 것이 가능해지고 있다, 프로그램이 작동하는 것을 인터넷으로 볼 수 있기 때문이다. 다운로드 횟수와 “인기” 수치는 시장 공유에 힌트가 된다. 그러나 이것은 편향적이기 쉽다. 웹서비스 엔진에서 얼마나 많은 링크를 가졌는지 알 수 있는 (Google, Advanced search and then use") 서비스를 이용한다면 적어도 다른 사람에게 얼마나 있는 있는지 알 수 있을 수도 있다. 중요한 것은 많이 링크가 되어 있는 것은 많은 사람이 흥미를 가질 뿐 아니라 이미 많은 사람이 사용해봤다는 뜻이 되기도 한다.

제품의 간접적인 측정은 선택한 Linux distribution에 포함되었는지 아닌지 판단하는 것이다. 예를 들어, Red hat Linux에 당신이 찾는 것이 포함되어 있다면 우선적으로 고려해볼만 하다는 것이다. 이러한 것은 이미 수많은 사람들의 의견이 반영되어 포함되기로 결정되었다는 뜻이기 때문이다.

라. 필요로 하는 프로그램의 속성 명백하게 비교하기

당신이 다른 리뷰를 읽고, 식별하고 난 후에, 당신은 실제로 당신이 원하는 최상의 것을 찾기 위한 비교를 해야 한다. 목표는 “가장 유사한” 몇 개의 후보 중에 실질적인 선택을 하는 것이다. 이 과정은 리뷰가 중요한 것이 존재하지 않거나 잊어먹었을 지도 모르는 중요한 속성을 식별해 주기 때문에, 적어도 몇 개의 리뷰를 읽은 후 해야 한다. 이 과정은 긴 절차를 요구하지 않는다. 당신은 모든 후보를 빠르게 제거해 나갈 수 있을 것이다.

첫 번째 단계는 OSS/FS 프로젝트 웹사이트를 찾는 것이다. 실제로 모든 OSS/FS 프로젝트는 프로젝트 웹 사이트를 가진다. 주소를 모른다면 검색 엔진에서 쉽게 찾을 수 있을 것이다. OSS/FS 프로젝트의 웹사이트

는 OSS/FS 프로그램을 제공하는 것만은 아니다. 이곳에선 선택을 위한 다양한 정보를 제공한다. 예를 들어 프로젝트 웹사이트는 일반적으로 FAQ, 프로젝트 문서 관련된 링크, 개발자 메일링 리스트, 프로젝트에 대한 사용자의 논의 같은 것을 포함한다. Software Release Practice HOWTO는 프로젝트 웹사이트를 어떻게 만들지에 대한 개발자 지침을 포함한다. Free Software Project Management HOWTO에는 각 프로젝트를 관리하는 자에 대한 지침을 제공한다.

드문 경우에 "fork" 할 수도 있다. 이 뜻은 단일 오리지널 프로그램을 기반으로 하는 프로그램이다. 이러한 경우가 생기면 기술과 프로젝트 지시를 받기 힘들 수도 있다. 이러한 경우에는 오리지널 프로그램의 웹사이트와 "fork"한 웹사이트를 모두 보고 평가해야할 것이다.

다음으로 당신은 중요한 속성을 기반으로 가진 프로젝트, 프로그램을 평가할 수 있다. 중요한 속성은 functionality, cost, market store, support, maintenance, reliability, performance, scaleability, useability, security, flexibility/customizability, interoperability, legal/license issues가 있다. 이 속성들은 프로그램을 사용할 때의 이득, 복구, 위험 등 결정에 도움을 주는 요인 들이다. 속성은 상용 소프트웨어의 것과 비슷하지만, OSS/FS의 경우 약간 다르다. 특히 프로젝트 코드가 공개되어 있기 때문에 당신은 평가 동안 이 정보의 장점을 취할 수 있다.

1) Functionality(기능성)

중요한 질문 중 하나는 단순성이다. 프로그램이 당신이 원하는 것을 하는가?, 이것은 이미 이전의 단계에서 필요한 것을 추렸기 때문에 원하는 것을 포함하고 있을 것이다. 많은 프로젝트 웹사이트에는 프로그램이 현제 가지는 기능의 짧고 쉬운 접근법에 대한 설명을 제공한다. 더욱 많은 정보

는 FAQ나 프로그램 문서를 참조하라.

당신에게 요구하는 특정한 기능은 당신의 특정한 요구와 프로그램의 종류에 따라 다르다. 그러나 모든 프로그램에 적용 가능한 일반적인 기능적 이유도 있다.

특히 당신은 당신이 기존에 가진 컴포넌트와 얼마나 잘 통합되고 호환되는지 고려해야 한다. 만약 관련된 표준 있다면, 프로그램이 그것을 지원하는가? 사용하는 것들 사이에 데이터 교환이 있다면, 얼마나 잘 교환이 되는가? 예를 들어, MOXIE가 Microsoft Office 포맷을 다운로드한 것을 얼마나 잘 표현하는가를 비교하고, 그 다음 얼마나 잘 다른 프로그램을 다루는 가를 비교하면 된다.

당신은 필요한 하드웨어와 운영체제, 관련된 프로그램을 고려해야 한다. 운영체제 요구사항 중 오직 Microsoft Windows에만 적용가능한 것들이 있다. 많은 OSS/FS 프로그램이 일반적으로 GNU/Linux 또는 Unix에서만 돌아가기 때문에 문제가 된다. 이러한 경우 프로그램을 빠르게 Windows 환경으로 포팅을 해야 한다. 이것은 시간이 필요한 작업이며, 또한 잘 작동되리라는 보장도 없다. 일반적으로 윈도우는 Linux와 Unix의 몇몇 중요한 특징(Low-Level "fork")을 지원하지 않고, 윈도우는 GUI 환경이기 때문이다. 때때로 서버 어플리케이션의 경우 값싼 컴퓨터에 GNU/Linux, FreeBSD, OpenBSD에 OSS/FS를 많이 사용한다. 오늘날 컴퓨터와 OSS/FS 운영체제가 특별한 목적의 일을 위해 점점 비용이 싸지고 있다. 특별한 목적의 운영체제는 범용적인 목적의 운영체제보다 보안이 더욱 쉽고 강력해지기 때문이다. 웹 브라우저나 원격 터미널에서 서버 어플리케이션을 통제할 수 있다. 그래서 서버와 데스크 답의 운용체제가 달라도 된다.

프로그램(상용, OSS/FS)은 당신이 원하는 모든 기능을 가진 것은 거의 없을 것이다. 그래서 종종 없는 것 지원하지 않는 기능들이 있을 것이고 그들끼리의 조합을 원할 것이다. 또한 프로그램은 당신의 목적에 효율적이지 안는 방법을 가질 수도 있다. 유연성과 customizability를 위해 부분적인 메커니즘을 가진 프로그램도 있다.

추가적으로 없는 부분은 코드를 변환함으로써 기능을 추가시킬 수도 있다. 내부적으로 또는 누군가를 사거나, 프로젝트를 통해서 기능을 추가할 것이다. 많은 사람들이 OSS/FS 프로젝트의 가치를 위해서 기능추가를 한다. 게다가 OSS/FS 프로젝트는 많은 사람들이 공통 프로젝트를 수정하는 협력관계(operating as consortias)를 통해 성공하게 된다. 기존 프로그램에 기능을 추가한는 것은 처음부터 만드는 것에 비해 유지보수, 관리, 수정에 대한 비용이 덜 든다. 그러나 기능 추가는 OSS/FS 프로그램의 비용을 증가시킨다. 새로운 기능에 대한 리스트, 계획, 기존 것의 변화에 대한 리스크 등 많은 고려사항이 있다. 만약 추가할 기능이 중요하다면 프로젝트 개발자들과 충분히 논의를 하고 비용과 시간을 수정해야할 것이다. 물론 기능 추가를 무시할 수도 있다.

2) 비용(Costs)

명백한 비용은 누구에게나 중요한 것이다. 대부분의 OSS/FS 프로그램은 얻는데 비용이 들지 않는다라고 생각한다. 그러나 free software에서 free라는 용어는 가격적인 면이 아닌 "freedom"을 기본으로 한다. OSS/FS 프로그램도 배포에 돈이 필요할 수도 있다.

그러므로 비용을 고려할 때, 프로그램의 배포까지 소요되는 비용을 고려해야 한다. 일반적으로 개발, 소유권, 기간에 따른 비용을 고려하면 될 것이다. 그러므로 초기 저작권 비용, 설치비용, 라이센스 업그레이드 비용,

인건비용, 지원/유지비용, 간접비용, 트렌지션 비용, 필요 하드웨어 비용 등 다양한 비용을 모두 교려하여야 한다.

많은 상용 벤더는 현재 초기 비용만을 보이며, 숨겨진 다른 비용들이 있다. 이것은 OSS/FS에도 마찬가지 이다. 확실하게 모든 비용에 대해 고려해야 하며, 초기 비용만을 따져서는 안 된다. 비싼 하드웨어 비용과, 팔기 위해 필요한 것, 지원 비용 업그레이드 비용들도 포함되어야 한다. 또 다른 위험 요소는 프로토콜과 포맷에 대한 확장이다. 프로토콜과 형식이 문서화되지 않고, 표준화되지 않고, 구현에 비용이 든다면, 이것도 차후 숨겨진 커다란 리스크가 될 것이다.

훈련비용도 있다. 훈련비용은 중요하며, 유사 기능에 대한 훈련비용을 참조하면 된다. 훈련비용은 재품 비용에 포함되지 않는 경우가 많으며, 취순 사용으로 훈련비용이 낮아지는 경향이 있다. 트렌지션 비용은 기존의 것에 익숙한 사람이 다른 것을 사용할 때 나타나는 비용이다. 쉽게 접할 수 있는 것이라면 이 역시 낮아질 것이다. 지원 비용의 경우 보통 OSS/FS의 경우 self-supported, competed 임으로 상용 제품에 비해 저렴한 편이다. 다시 말하지만, 당신은 당신의 특수한 환경에 필요한 모든 것을 고려해야 할 것이다.

3) 시장 공유(Market Share)

처음에 이야기 한 것처럼, OSS/FS 프로그램이 얼마나 인기 있는지도 중요하다. 이전 section에서 이야기 한 것처럼, 시장 공유는 많은 정보를 포함하고 있다.

4) 지원

이 글의 목적을 위해, “support" 라는 용어는 많은 영역에 영향(cover)

을 준다. 이것은 재품의 설명, 많은 정보 소스에 포함된다. fixing th project, 새로운 기능의 추가에 관한 자세한 정보는 maintenance section 을 참고해라

OSS/FS와 상용 프로그램의 중요한 차이 중 하나는 어떠한 support 하는 방법이다. 기본적으로 OSS/FS 프로그램 유저는 다음과 같은 선택을 할 수 있다.

⊙ 전통적인 상용 support 모델
⊙ 내부적인 support
⊙ 개발자와 유저 커뮤니티에 의한 support

선택은 일에 따라 범위가 다르고 또한 대답도 다르다

전통적인 상용 support 모델의 support를 제공하는 다른 회사 또는 사람을 사용하는 것이다. 이때에는 비용, 평판 등 많은 기준으로 평가해야 한다. 게다가 프로젝트 소프트웨어를 개발한 곳에 직접 support를 요청할 수도 있다.

OSS/FS 프로젝트 페이지를 보라. 프로젝트 기여자의 email address를 얻어서 기여자와 접촉을 시도해 보라. 많은 경우 상용 벤더와 유사한 방식으로 주요 support를 제공하는 누군가가 있을 것이다. 조직에는 이러한 일을 담당하는 컨설턴트나 supplier가 있을 것이다. 심지어 개발자 커뮤니티 또는 유저 커뮤니티에 의존하는 방법도 있을 것이다. 1997년에 Linux User Community는 InforWorld에서 “Best Technical Support" 를 수상하였다. 그러나 모든 OSS/FS 제품이 이러한 방식으로 잘 support 되는 것은 아니다. 메일링 리스트의 actives를 보고, 유저의 질문에 얼마나 잘 대답하였는지를 보고 유저의 만족도를 체크하면 된다. 모든 질문에 대답하는 것은 불가능하다. 그러나 이것이 최고의 방법일 수 있다. 만약 질문자가 충분한 정보를 주고, 그것이 자신의 상황과 맞다면 그에 대한 대답이

충분한 support가 될 것이다.

5) 유지(Maintenance)/Longevity

유용한 프로그램이 완전하게 지속적인 경우는 드물다. 요구가 변하고, 새로운 사용자가 요구를 만들기 때문에 완벽한 프로그램이란 존재하지 않는다. 프로그램을 유지하는 것이 그래서 중요한 것이며, 미래에도 계속 유지될 것이다. 물론 미래를 예상하는 것은 어렵다 그러나 만약 프로그램이 활동적으로 유지된다면, 그렇지 않은 프로그램보다 더 많이 사용될 것이다.

OSS/FS 유지 옵션은 support 정도로 중요한 것이다. 그리고 실질적인 유자와 support는 완전히 나누어 진 것이 아니다. OSS/FS support 조직은 제품의 유지 경험을 가진다면, 요구에 접합한 변경할 수 있을 것이다.

OSS/FS 프로젝트 사이트는 다양한 유저의 개선사항을 수집하는 지점이기도 하다. 그러므로 프로젝트의 인터넷 존재는 프로그램을 유지하는 법에 대한 지침을 줄 것이다. 개발자의 메일링 리스트 기록은 개발자가 개선과 버그 수정에 대한 체크들도 포함된다. Freshmeat의 “vitality" 측정을 가진다. 어떠한 OSS/FS 프로그램은 유지가 없는 경우가 있다. 이것은 큰 리스크가 될 것이다. 일반적으로 “소프트웨어가 개발 중인가?”, “멈추진 않았는가?”를 확인해 보는 것이 좋다.

이미 많은 사용자가 사용을 했으며, 일반적으로 긍정적인 리뷰가 달린 “just works”한 변화가 없는 프로그램도 있다.

물론 법적인 활동으로 인해 프로젝트가 중단되었다면, longevity에 큰 영향을 미칠 것이다. 이러한 경우는 극히 드물다. 자세한 정보는 legal section을 참조하라

상용 소프트웨어는 longevity 동안 자동적으로 진화할 수도 있다. 사실

OSS/FS의 중요한 장점중 하나는 당신이 self-support 또는 재구성를 위해 지속적으로 소스코드를 사용하는 것이다. 이와 대조적으로 상용 프로그램은 계속하여 support 하지 않고 갑자기 사라질 수도 있다. 예를 들어, A라는 회사의 제품을 구매하였고, support를 받고 있다가, A라는 회사가 사라지거나 정책적 문제로 제품의 판매나 support를 중단할 수도 있다.

6) 신뢰성(Reliability)

신뢰성의 어떻게 프로그램이 적절한 답을 내는 가에 대한 측정이다. availability와 유사하다. 그러나 Reliability는 어떻게 사용하는 가에 크게 의존한다. 신뢰성을 측정하는 가장 좋은 방법은 "실제"로 작동시키는 것이다. 이것은 이후에 논의 하겠다.

정보는 보통 신뢰성과 같은 프로그램의 표준치에 도움을 줄 수 있다. 특히 측정 프로그램은 신뢰성에 많은 영향을 미친다. 프로젝트 웹 사이트는 프로그램의 성숙도를 설명하려는 시도를 할 것 같다. 사실 개발자들은 완벽주의자 이여서 자신이 개발한 것은 완벽할 것이라고 생각한다. Freshmeat에 있는 성숙도 측정은 도움을 줄 것이다. SourceForge의 Development Status 측정도 도움이 될 것이다. 나는 GRAM OSS/FS 프로그램을 식별하려고도 했었다. GNU/Linux distribution에 포함된 OSS/FS 프로그램은 이미 성숙한 것이다.

7) 성능(Performance)

많은 프로젝트 웹 사이트는 성능 데이터를 포함한다. 어떠한 OSS/FS 프로젝트는 단지 긍정적인 데이터만 보이고 있다. 메일링 리스트에는 많은 성능 정보가 있다. 성능을 측정하는 최고의 방법은 "실제"로 작동시키는 것이다. 이것은 이후에 논의 하겠다.

8) 확장성(Scaleability)

확장성은 프로그램이 다룰 수 있는 문제나 데이터의 크기를 말한다. 만약 당신이 비정상적으로 큰 데이터를 사용하기 원한다면, 프로그램이 충분히 그를 받쳐줄 수 있어야 한다.

9) 사용성(Useability)

사용성은 인간과 기계 사이의 인터페이스의 품질을 측정하는 것이다. 높은 사용성의 프로그램은 배우기도 쉽고 사용하기도 쉽다.

어떠한 프로그램은 다른 프로그램에 의해서만 사용될 수 있으며, 유저가 직접 사용할 수 없을 수도 있다. 대표적인 경우가 API이며 이러한 경우, 프로그래머가 얼마나 사용하기 쉬운가가 사용성이 된다. 측정하는 방법은 샘플을 보고 판단하는 것이다.

어플리케이션은 사용자의 사용에 의해서 결정된다. 오늘날에 대다수 프로그램이 사용하는 인간-기계 인터페이스는 커멘드 라인 인터페이스와 그래픽 유저 인터페이스(GUI)이다. 상호 배타적인 것이 아니며, 대다수 프로그램은 이 둘을 동시에 사용한다. 커멘드 라인은 프로그램을 사용하기 쉽고 많은 프로그래머, 시스템 관리자가 이 방식을 좋아한다. 그래서 대상 목표나 목적이 이들이라면 커멘드 라인 인터페이스를 사용하는 것이 좋다. 그러나 대다수의 프로그램 사용자의 경우 GUI를 선호한다. 대부분의 경우, 당신은 GUI를 선택할 것이다.

OSS/FS 프로그램은 2가지 부분으로 설계된다. 하나는 엔진이며, 하나는 인터페이스에 관한 부분이다. 이렇게 나누면 reliability가 향상된다. 일부 사용자의 경우 실질적으로 인터페이스를 거치지 않고 엔진을 직접 사용하여 일을 하고 싶어 할 수도 있다. 이러할 때에 커멘드 라인 인터페이스

가 사용될 수 있다. 그러나 대부분의 사용자는 쉽게 접할 수 있는 GUI 인터페이스를 원할 것이다. 그래서 보통 OSS/FS 프로젝트의 경우 엔진을 만들고 그에 따른 자메 프로젝트로 GUI를 따로 만드는 경향이 있다.

많은 OSS/FS 유저 인터페이스는 웹브라우저를 사용하여 구현되었다. 이것은 많은 장점을 가진다. 웹 브라운저나 운영체제에 설치하지 않고 사용할 수 있고, 대부분의 사용자는 웹브라우저로 작업하는 것에 익숙하다는 것이다. 그러나 웹 인터페이스의 좋고 나쁨도 역시 인터페이스의 사용성에 대한 평가가 필요하다.

만약 GNOME, KDE 어플리케이션이라면 관련된 유저 인터페이스 지침을 따를 수도 있다. GNOME 어플리케이션의 경우 HIG 라는 것을 제공한다. KDE의 경우 KDE user interface guidelines을 제공한다. 이것은 Freedesktop.org에 논의 되어 있다. 또한 Jim Gettys' Open Source Desktop Technology Road Map은 현제 또는 일반적인 소픈소스의 유용한 지침을 제공한다.

마지막으로 사용성 평가는 hands-on 테스팅을 요구한다.

10) 보안

제품의 보안을 평가하는 것은 복잡하다. 같은 제품일지라도 다른 보안 요구사항과 다른 환경, 다른 사용을 하기 때문이다. 이러한 것을 해결하는 방법은 당신의 보안 요구사항을 명확하게 식별하는 것이다.

어떤 프로젝트는 코드의 잠제적인 결점을 찾기 위해서 리뷰어들의 평가로 알아내기도 한다. OpenBSD가 이러한 방식을 사용한다.

물론 만약 보안이 당신에게 중요한 요소라면 개발 시작부터 이것을 고려해야 한다. 이 글의 discusses evaluation software security in more

detail section을 참조하라.

11) 유연성(Flexibility)/Customizability

이 둘은 높은 내적 속성이다. 유연성은 프로그램이 설계에 포함되지 않은 비정상적인 환경에서 얼마나 작동하는지를 측정하는 것이다. Customizability은 당신이 명시한 환경에서 적합하게 제품을 customize 할 수 있는 가를 측정하는 것이다. OSS/FS 프로그램은 상용 프로그램보다 이 두 가지 속성이 더 좋은 편이다. 그러나 이 두 가지 속성은 개발자에게 소스코드를 이해하고 수정하는 스킬을 요구한다. 물론 OSS/FS 프로그램은 상용 프로그램에 비해 쉽게 확장할 수 있다. 이러한 것을 위해 template, "plug-in", API, command language를 제공한다.

12) Interoperability

OSS/FS 제품은 일반적으로 관련된 표준에 따라 구현한다.

13) 법적/저작권 이슈

법적 이슈는 다른 중요한 속성이다. 이것은 프로그램의 저작권에 정의된 것이다. 그러므로 당신은 저작권 요구사항을 고려하고, 그 나라의 법적 문제도 고려해야 한다.

소프트웨어의 다른 속성과 다르게 이 속성은 상용 프로그램의 평가 관점에 보게 된다. 이것은 잘못된 것이다. 당신이 상용 소프트웨어를 평가할 때, EULA와 같은 저작권 용어를 확인하면 된다. 만약 유사한 EULA를 확인할 지라도 당신의 조직에 적용할 수 있는 EULA를 찾을 수는 없을 것이다. 물론 OSS/FS의 저작권 상태도 확인해야 한다. OSS/FS 프로그램은 상용 소프트웨어와 다른지만, 유저의 권리가 있는 것이다. 개발자마다

개발의 동기가 다르듯이 소프트웨어 저작권도 그들의 동기에 따라 달라진다. 특히 프로그램을 수정한다면 이러한 저작권은 그 상황에 문제가 없는지 확인해야 한다. 이것은 copyleft 라는 용어를 참조하기 바란다.

확실히 OSS/FS 프로젝트와 상용 벤터는 법적 행동이 다르다. 이것이 모든 유저에게 영향을 줄 수 있다. 그러므로 그것을 우선 체크하고 고려하는 것이 좋을 것이다. A Comparison of the GPL and th Microsoft EULA"의 글에는 GPL과 EULA의 비교가 있다. Sydney Morning Herald article에 이에 대한 요약이 있다.

중요한 것은 OSS/FS를 사용할 지라도 모든 고용자들은 이 규정을 따라야 한다는 것이다. Optaros ha published its Free and Open Source Software Policy가 정책에 좋은 예시가 될 것이다. 이곳에는 특정한 법적 장치가 없다. 그러나 도움일 될만한 법적 이슈를 포함하였다.

⊙ Warranty/legal recourse
⊙ License Audits
⊙ License issues unique to OSS/FS
  - Checking if the program is OSS/FS
  - Why different OSS/FS licenses matter
  - Copylefting vs. non-coylefting
  - Computer Libraries
  - Other examples of license impacts
  - Patent defense
  - License summary

14) 다른 이슈

물론 당신에게 중요한 다른 이슈도 있다. 지역적 정책이나 특별한 기술 요구사항 같은 것이다. 명확히 당신의 평가에 이러한 것을 포함할 수도 있

다.

다른 문제는 OSS/FS 프로그램의 shareware 와 freeware 프로그램의 혼동이다. OSS/FS는 shareware와 freeware라는 용어를 사용하지 않는다. 이 용어는 상용 프로그램의 배포 방식에 대한 용어이다.

마. 최종 후보의 더 자세한 분석 수행

초기 평가가 된 후에 최종 후보가 나올 것이며 이에 대해 더욱 자세한 분석을 해야 한다. 특히 이 부분은 많은 작업 부하가 있을 것이다.

이 단계는 OSS/FS 프로그램과 상용 프로그램이 같은 방법을 사용한다. 고려해야할 중요한 속성은 이전 단계와 같다.

더미 데이터를 통해 성능, scaleability, 동작 등을 볼 수 있다. 요즘과 같이 컴퓨터 환경에 좋아지면서 성능은 더 이상 중요한 것이 아닐 수 있지만, 성능에 대해서는 여전히 평가해보아야 한다. 만약 성능이 중용한 요소라면 테스트 환경을 더욱 실질적으로 만드는 것이 좋다.

명백히 당신은 비-비극적(critical) 상황이 아닌 곳에서 새로운 프로그램을 시도할 것이다. 그리고 얼마나 잘 작동하는지 관찰할 것이다. 만약 많은 돈이 들어간다면, 가격에 대한 추가적인 이슈를 확인해 보아라.

프로그램의 많은 버전에 대해 주의 깊게 살펴야 한다. 이전 버전이 올바로 작동한다고 해도 이후 버전이 올바로 작동한다는 보장이 없다. 이것은 OSS/FS에 중요한 부분이다.

1) 기능 추가를 위한 세부 분석

만약 OSS/FS 프로그램이 모든 기능을 제공한다면 당신은 이 부분을

읽을 필요는 없다. 그러나 그런 경우는 극히 드물다. 그래서 추가적으로 기능을 추가해야 한다.

만약 내부적으로 처리하기로 했다면, 프로그램 설계문서와 소스 코드가 함께 필요할 것이다. 잘 디자인 된 프로그램은 쉽게 수정하고 이해할 수 있다. 추가할 기능을 알고, 프로젝트 개발자와 함께 이것을 이야기 해보아라. 그리고 기능을 추가하면 된다.

2) 소프트웨어 보안을 위한 세부 분석

보안에 대한 특별한 가치를 가진 코드가 있을 수도 있다. 당신은 개발 전문가에게 당신의 코드를 보임으로써 가능할 수 있다. 보이는 예시는 다음과 같다.

⊙ it minimizes privileges
⊙ it strives for simplicity
⊙ it carefully checks inputs
⊙ source code scanning tools such as RATS and Flaw finder report few problems

물론 개발 전문가가 한 번에 어떻게 개발할지 알려줄 때도 있다. 그러나 대부분 정보는 충분하지 않을 것이다. 만약 세부적인 사항을 알지 못한다면 그것을 우선 배워야 한다. My book on how to write secure programs에서 공짜로 이에 관한 것을 사용할 수 있다.

다른 접근 방법은 Common Criteria 평가를 할 수 있는 자를 고용하는 것이다. 이것은 시간과 돈이 들어가는 일이지만 확실한 방법 중 하나이다.

바. Wrap-up

OSS/FS와 상용 프로그램을 같은 방법으로 평가할 수 있다. 그러나 OSS/FS 프로그램을 평가하기 위한 정보를 얻는 방법은 상용 프로그램의 그것과는 다르다. 이 프로세스에 따르면 어떠한 guarantee도 없다 물론 다른 프로세스도 마찬가지 이다. 그러나 best answer를 찾는 법을 알려준다. 그러나 나는 이 절차를 통해 이유 있는 답을 얻을 수 있는 확률이 높다고 생각한다. 만약 당신이 이 방법을 사용한다면 당신의 평가 결과가 더욱 빠르게 나올 것이다.

1) 문제
2) 추천 해답, 보통 대답 중 마지막 부분에 있는 대답이 믿을 만하다.
3) 추천을 사용한 프로세스, 특히 4가지 기본 스텝을 이용하여 평가하길 바란다. 자세한 내용은 How to Evaluate Open Source Software / Free Software (OSS/FS) Programs by David A. Wheeler, http://www.dwheeler.com/oss_fs_eval.html을 참고하라
4) brief description 참조한 주요 대안을 식별하라. 각 프로그램의 다양한 버전이 있다. OSS/FS 프로그램은 빠르게 개선되고 있고 이전의 버전과 최신의 버전이 다를 수도 있다. 가능하다면 best, worst를 등급을 매기고 가장 적합한 대안을 선택하는 것이 좋다. 정량적인 수치를 매기는 경우, 이전에 선택한 프로그램이 잘못된 경우 다음 등급의 프로그램을 대안으로 선택하면 된다.
5) 가능하다면 각 대안에 따라 세부적인 논의를 해보아라.
6) final 추천으로 결론을 내려라.

참고문헌

[이도규06] 공개소프트웨어 정책성과 발전방향, 이도규, 2006. 6. 정보과학회지 제24권 제6호

[김두연06a] 공개소프트웨어의 성능 검증 프로세스, 숭실대, 김두연, 2006

[김두연06b] 벤치마크 테스트를 통한 공개소프트웨어 검증 절차에 관한 연구, 김두연, 한국 IT서비스학회, 2006

[정통기05a] S/W 벤치마크테스트 평가모델 개발 이슈, 한국정보통신기술협회 시험인증연구소S/W시험인증 센터, 2005

[정통기05b] 소프트웨어 벤치마크테스트(BMT) 현황, 한국정보통신기술협회, 2005

[박호인97] 소프트웨어 제품을 위한 평가 선정을 위한 모형 비교 및 적용에 관한 연구, 박호인, 정보처리학회 논문지 제4권 제7호 , 1997

[이종무97] 소프트웨어 품질평가 투입요소 결정에 관한 연구, 고려대학교 대학원 경영학과, 이종무, 1997

[정통기03] 소프트웨어 품질보증, 한국정보산업연합회, 2003

[최석진92] 소프트웨어 품질평가를 위한 최적화 모델에 관한 연구, 한양대학교 산업대학원, 최석진, 1992

[진흥원05] SW제품의 품질정보 제공을 위한 BMT 시행, 한국소프트웨어진흥원,2005

[정보진06] 최신 정보시스템 성능평가 모델 도입방안 연구, 정보사회진흥원, 2006

[진흥원03] S/W BMT 기술동향 및 활성화 방안 연구보고서, 한국소프트웨어진흥원, 2003.12

[정통기06] BPM(Business Process Management) SW 벤치마크테스트, 한국정보통신기술협회, 2006

[DAV06] David A Wheeler, How to Evaluate Open Source Software

Programs, http://www.dwheeler.com/oss-fs-eval.html, 2006.
[IPAWEB] http://www.ipa.go.jp/software/open/forum/development
[IPA05a] 2005 년도 상반기 오픈 소스 소프트웨어 활용 기반 정비 사업 「OSS 성능·신뢰성 평가 / 장해 해석 툴 개발」OS단계-장해 해석의 순서·툴 평가편,IPA,2005
[IPA05b] 2005 년도 상반기 오픈 소스 소프트웨어 활용 기반 정비 사업 「OSS 성능·신뢰성 평가 / 장해 해석 툴 개발」DB단계-OSDL DBT-1/3에 의한 DBMS 평가편, IPA,2005
[IPA05c] SPECjAppServer2004에 의한 JBoss의 성능·신뢰성 평가-평가, IPA,2005]