성능 테스트 개론

Overview of Performance Test

Jan. 2013

JENNIFER

Overview of Performance Test
Table Of Contents

1 성능 테스트 개요 3

2 성능 테스트 범위 3

3 성능 테스트 접근방안 5

3.1 아키텍처 검증 7

3.2 시스템 성능 검증 7

4 성능 테스트 목표 8

4.1 업무량 Throughput 8

4.1.1 단위 성능 테스트 목표 10

4.1.2 통합 성능 테스트 목표 10

4.2 응답시간 Response Time 10

4.3 시스템 사용률 System Resource Usage 11

5 성능 테스트 프로세스 12

6 업무 성능 모니터링 Application Performance Monitoring 15

7 마치며 16

8 저자에 대하여 16

Overview of Performance Test

오늘날 많은 기업들은 자유 시장 체제로의 전환에 따른 무한 경쟁 시대에 직면하고 있다. 기존의 비즈니스가 주로 자국 내의 소비자들을 위주로 진행되었던 형태라면, 글로벌 경쟁체제에서는 전세계 불특정 소비자들을 대상으로 비즈니스가 이루어지게 된다.

이러한 시장 변화는 비즈니스를 지원하는 정보 시스템에도 많은 혁신을 요구하고 있다. 여러 산업계를 막론하고 이미 완료되었거나 현재 진행되고 있는 차세대 정보 시스템 구축 프로젝트들의 추진 전략을 요약해 보면, 각 기업들이 당면하고 있는 현안들이 어떤 것인지 가히 짐작해 볼 수 있다.

"시장의 변화에 능동적으로 대응 가능한 유연한 시스템 구현"

"시장의 급속한 성장에 대응 가능한 고 확장 시스템 구현"

"경쟁사 대비 서비스 차별화가 가능한 고 품질 시스템 구현"

하지만 이러한 시대적 사명을 가지고 기업들의 전폭적인 지원 속에 출발한 대형 프로젝트 들은 진행도중 많은 난관에 봉착하여 큰 어려움을 겪는데 그중에서 가장 치명적인 것은 다름아닌 시스템 성능문제라고 볼 수 있다.

일반적으로 어플리케이션 기능문제들은 오픈 막바지에는 코드 변경에 따른 부작용(Side Effect)을 없애기 위해서 문제 해결을 보류하고 있다가 오픈 이후 안정화 기간 동안에 반영하거나, 완벽한 해결책은 아니지만 차선책으로 대응하더라도 큰 문제는 없다.

반면 오픈 직전까지 시스템 성능을 보장 못하는 이슈가 발생하게 되면, 프로젝트 주요 의사결정자들은 시스템 오픈 자체를 재고 해야 하는 상황에 놓이게 된다. 수백 내지 수천 명에 달하는 인력들이 참여하는 대형 프로젝트에서 시스템 오픈 일정을 미루게 될 경우, 해당 기업은 추가로 지불해야 하는 천문학적인 비용을 넘어서 기업 이미지에도 상당한 타격을 입게 된다.

“그럼 정보 시스템 성능 문제를 최소화하고 올바르게 대처하기 위해서는 어떤 노력들이 필요할까?”

현실적으로 각 프로젝트마다 처해 있는 상황들이 모두 상이하고, 시스템 구현에 활용되는 기술요소, 솔루션 패키지 그리고 프로젝트를 이끄는 주사업자들의 유사 프로젝트 경험 등에 의해서 상호 영향을 받을 수 있기 때문에 핵심을 꼬집어서 한마디로 정의하기란 쉽지 않다. 하지만 모든 사람들이 공감할 수 있는 성능 문제에 대한 핵심적인 대응책은 시스템 성능에 대한 체계적인 이해와 더불어, 성능 검증을 위한 성능 테스트에 대한 올바른 이해라고 단언할 수 있다.

이러한 판단의 배경에는 현재 업계에서 통용되고 있는 테스트 표준화 및 시장 동향들이 주로 기능 테스트 영역에 치중하고 있고, 성능 테스트 영역을 제대로 이해하고 관련 기술력을 보유하고 있는 전문가들이 매우 드문 현실이 있다.

본 기고에서는 앞서 소개한 시스템 성능 분석 방안을 기반으로 성능 테스트 분야에 지침서로 활용할 수 있도록 성능 테스트 방안에 대해서 소개하고자 한다.

1 | 성능 테스트 개요

현실세계에서 일어나는 복잡 다변한 상황들을 정확하게 예측하는 것은 모든 사람들의 꿈이다. 그러나 그것은 매우 어렵다. 아니 불가능하다고 보는 것이 맞을지도 모른다. 에드워드 로렌츠의 유명한 나비 효과 이론 주장처럼, 특정 상황에 영향을 미칠 수 있는 여러 유형의 변수들은 항상 존재하기 마련이고, 최종 상황은 특정 변수의 아주 작은 변화에도 초기 예측과는 전혀 다른 양상으로 바뀔 수가 있는 것이다.

정보 시스템의 경우도 마찬가지다. 아키텍트는 시스템 구축을 위한 아키텍처를 설계하기 위해 비 기능 요구사항을 참조하는데, 여기에는 성능, 확장성, 가용성, 기타 운영 관리(백업/복구, 모니터링)항목 등이 포함된다. 이들 중 시스템 성능 부문은 오픈 이후에 발생 가능한 예상 업무량, 업무 증가율, 업무 유형, 구축 기술 여러 변수들을 고려하여 시스템 용량을 계획하기 때문에, 예측 결과에 대한 신뢰도를 높이기에는 현실적인 어려움이 따른다.

여기서 더해 시스템 성능 요구사항을 하드웨어 요구사항으로 변환시키는 시스템 용량 산정은 그 신뢰도를 더욱 낮추게 한다. 최근까지도 시스템 규모 산정을 위한 명확한 규모 산정 가이드라인 부재로 인해서, 용량 산정은 각 기업들과 하드웨어 벤더의 주관적인 방법에 따라 시행되고 있다. 이로 인한 폐단으로 각 기업들은 프로젝트 기간 동안 시스템 규모의 과다 또는 부족으로 인해서 많은 고충을 겪고 있는 실정이다.

따라서 시스템 용량 계획 및 용량 산정 결과에 대한 낮은 신뢰도를 보완하기 위한 과학적인 접근 방법이 요구된다.

판단컨대 성능 테스트는 이러한 목적에 부합하는 유일한 대응방안으로써, 용량 계획에서 정의된 주요 성능 모델 변수 즉 최대 동시 사용자, 예상 최대 업무량, 업무 유형 등을 이용하여 시스템 성능과 용량 산정에 대한 적합성 여부를 검증해준다.

2 | 성능 테스트 범위

차세대 정보 시스템 구축 시에 생성되는 테스트 케이스 건수는 산업계 별로 약간의 차이는 있지만 대략적으로 5,000개에서 10,000개 정도로 매우 많은 편이다. 성능 테스트는 전체 기능 테스트 케이스를 대상으로 하기에는 시간과 비용측면에서 효용성이 떨어지기 때문에, 적정 수준의 테스트 대상을 선별하기 위한 가이드라인은 반드시 필요하다.

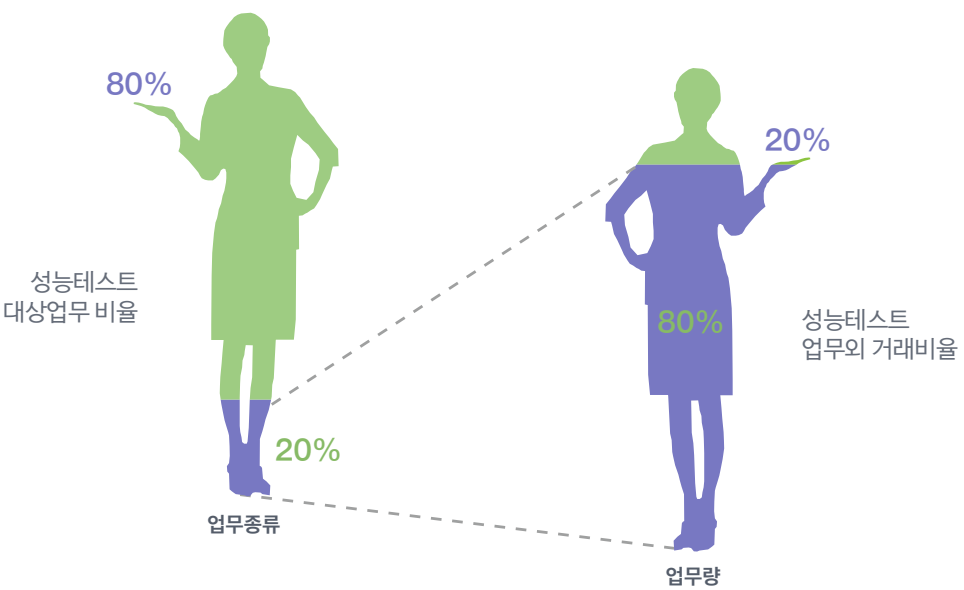
최근 기능 테스트 분야에서 많은 관심을 한 몸에 받고 있는 리스크 기반 테스팅은 이러한 상황하에서 매우 효과적인 대안 책이 될 수 있다. 즉 테스트 대상에 비해서 테스트 시간,

테스트 인력 등의 테스트 자원이 턱없이 부족할 경우, 대상 시스템의 리스크 분석을 통해 집중적으로 테스트 할 부분과 테스트 우선순위를 정하여 테스트를 효과적이고 효율적으로 수행할 수 있게 된다.

리스크 기반 테스팅에서는 리스크 비용이 높은 기능들을 위주로 높은 커버리지 수준의 테스트와 회귀테스트를 진행하고 리스크 비용이 높지 않은 기능들의 경우 상대적으로 낮은 커버리지 수준의 테스트를 진행하게 된다. 여기서 리스크 비용이 높은 항목을 식별하기 위해서는 실패 발생 확률과 실패 비용에 대한 사전 분석이 요구되는데, 성능 테스트의 경우에는 실패 발생 확률과 관련된 사용 빈도와 비즈니스 영향도를 위주로 분석을 진행하면 된다.

통상적으로 사용 빈도 관점에서는 피크 시간 동안 사용자들이 요청한 업무 발생 건수를 기준으로 상위 80%에 해당하는 업무들을 선별하는데, 이는 통계학적으로 널리 사용되는 파레토 법칙에 기인하고 있다. 80대 20법칙으로도 알려져 있는 이 법칙은 '전체 결과의 80%가 전체 원인의 20%에서 일어나는 현상'을 나타내는 것으로, 사용자들이 가장 많이 사용하는 업무 종류 기준 20%이내의 업무들이 시스템 전체 성능을 대변할 수 있다는 논리적 근거를 제공해 준다.

하지만, 업무 종류 관점에서 따져 볼 때는 전혀 다른 해석이 나올 수도 있다. 실제로 업무 발생 건수 기준 상위 80%에 해당하는 업무들을 선별해 보면 업무 종류로 따져볼 때는 전체 대비 10%에도 미치지 못하는 경우가 빈번하기 때문에, 90%에 해당되는 나머지 업무들은 성능 모니터링, 성능 튜닝에 기반한 성능 확보 방안도 함께 고민해야 한다.



3 | 성능 테스트 접근방안

프로젝트 각 개발 단계별로 요구되는 성능 테스트는 검증 목적에 따라서 다음과 같이 두 가지 형태로 구분해 볼 수 있다.

3.1. 아키텍처 검증

설계 단계에서는 신규로 도입된 솔루션 패키지과 개발 프레임워크에 대한 아키텍처 디자인을 확정하기 위한 사전 검증이 요구된다. 통상적으로 솔루션 패키지와 개발 프레임워크 벤더들은 가급적 기본 기능을 그대로 사용하는 것을 추천하지만, 기업 현실을 고려할 때 일정 부분에 대한 커스터마이징과 추가 개발은 불가피하다.

따라서 패키지 커스터마이징 및 추가 개발에 따른 아키텍처 변경 영향도를 사전에 체크하는 아키텍처 검증테스트는 반드시 고려되어야 하며, 아직 운영 환경을 이용할 수 없기 때문에 운영 환경 대비 10% 정도의 적정 규모의 별도 테스트 환경을 준비해야 한다.

아키텍처 검증 테스트 유형은 성능 검증 외에도 다양한 관점에서 검증이 필요하기 때문에 본 기고에서 모든 사항을 나열할 수는 없지만, 주로 많이 사용되는 아키텍처 디자인 검증, 아키텍처 성능 검증, 아키텍처 확장성 검증, 아키텍처 가용성 검증 등으로 구분해서 소개하고자 한다.

먼저 아키텍처 디자인 검증테스트는 아키텍트가 설계한 시스템 아키텍처 상에 여러 옵션들이 존재할 경우 각 옵션별 장단점 분석을 통해 최적의 방안을 선택하기 위해서 수행된다. 통상적으로 솔루션 패키지의 경우 솔루션 벤더에서 제시하는 베스트 프랙티스를 기반으로 설계가 이루어 지기 때문에 큰 이슈는 발생하지 않지만, 개발 프레임워크의 경우에는 프로젝트 상황에 맞게끔 많은 영역에 대한 보강이 이루어지기 때문에 프로토타입을 기반으로 하는 아키텍처 디자인 검증은 반드시 필요하다.

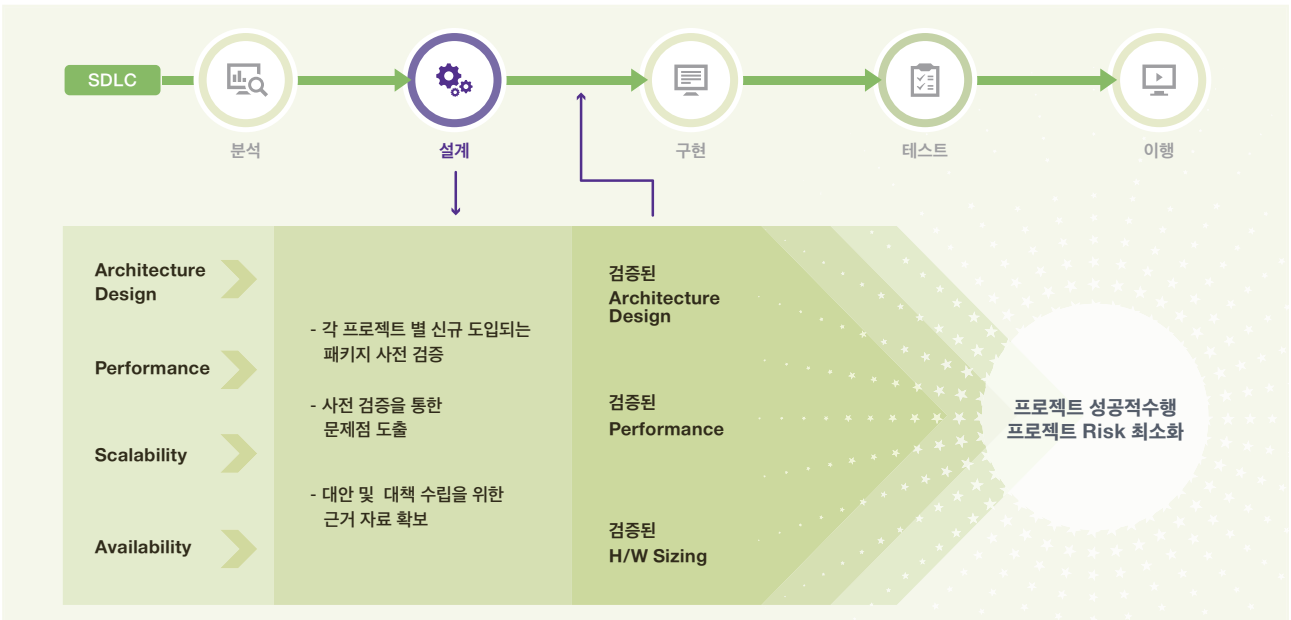
성능 검증 테스트**(Baseline Performance Test)**는 개발이 완료된 업무를 시스템에 적용하기 이전에 도입된 솔루션 패키지와 개발 프레임워크가 프로젝트 성능 목표를 달성하는 지와 더불어 제품 관련 이슈가 없는 지를 검증하기 위해서 수행된다. 만약 솔루션 패키지나 개발 프레임워크가 잠재적인 큰 이슈를 가지고 있을 경우 이를 해결하는데 많은 시간이 걸릴 수 있기 때문에, 사전 검증을 통하여 조기에 식별하기 위한 노력이 필요하다. 설계 단계에서는 아직 업무 개발이 이루어지고 있지 않기 때문에, 솔루션 패키지가 제공하는 기본 기능**(Out of the Box)**을 기반으로 검증할지, 거래량이 많은 대상 업무들을 프로토타입으로 개발해서 검증할지에 대한 선택이 필요하다.

아키텍처 확장성 검증테스트는 향후 시스템 업무량 증가에 따른 시스템 아키텍처의 확장성을 검증하기 위해서 수행된다. 이때 검증 유형은 아키텍처 확장을 위한 컴포넌트 증가 유형에 따라 다음과 같이 두가지로 구분된다.

첫째 수평적 확장성(Horizontal Scalability, Scale Out) 검증은 시스템 증설이 필요할 때 서비스와 관련된 서버 수를 하나씩 늘이고, 개선된 성능 비율을 측정하여 확장계수에 기반한 확장성을 검증한다. 기본적으로 서버 추가 시 병목 지점이 없을 경우에는 확장계수는 100%에 근접하게 측정되지만, 서버 간의 정보 동기화를 비롯한 병목 지점이 있을 경우 확장계수에 영향을 주기 때문에 반드시 측정을 통한 검증이 필요하다.

둘째 수직적 확장성(Vertical Scalability, Scale Up) 검증은 시스템 증설이 필요할 때 기존 서버에 구성된 CPU, Memory과 같은 시스템 리소스 자원을 추가하는 방법이다. 일반적으로 아키텍처 확장할 때, 시스템 리소스 자원을 추가하는 방식이 서버 수를 늘리는 방식에 비해 서버 관리 공수와 운영 리스크 면에서 훨씬 효율적이다. 따라서 신규 서버 도입 시에는 향후 시스템 리소스 자원에 대한 확장이 가능한지 사전 파악이 필요하다.

아키텍처 가용성 검증테스트는 일명 Fail-Over테스트라고 불리며, 주요 컴포넌트 장애에 따른 비즈니스 영향도를 검증하기 위해서 수행된다. 차세대 정보 시스템과 같은 대형 프로젝트들의 경우 아키텍처 설계 시에 서비스 가용률을 정의하게 되어 있는데, 목표값의 수준에 따라 개별 컴포넌트 이중화, 고 가용성 클러스터, 재해 복구 센터와 같은 추가 아키텍처 구성 여부가 결정된다. 여기서 목표 달성 여부는 가용성 테스트 결과를 통해서 수집된 각 장애 유형별 복구 시간을 이용하여 가늠 해 볼 수 있다. 시스템 아키텍처가 매우 복잡할 경우 각 컴포넌트가 가지고 있는 잠재적인 이슈를 조기에 찾기 위한 노력은 프로젝트 리스크를 최소화하는 차원에서 반드시 강구되어야 한다.



3.2. 시스템 성능 검증

프로젝트가 구현 단계를 거쳐서 업무 개발 단계 막바지에 이르게 되면, 프로그램 개발이 완료된 업무들을 대상으로 시스템 성능 테스트를 고려하게 된다. 일반적으로 업무 성능 검증테스트는 테스트 라이프사이클 중 시스템 테스트의 일환으로 진행되며, 기능 관점의 통합 테스트를 통과한 기능상 하자가 없는 프로그램 코드를 기반으로 실제 운영 환경에서 수행된다.

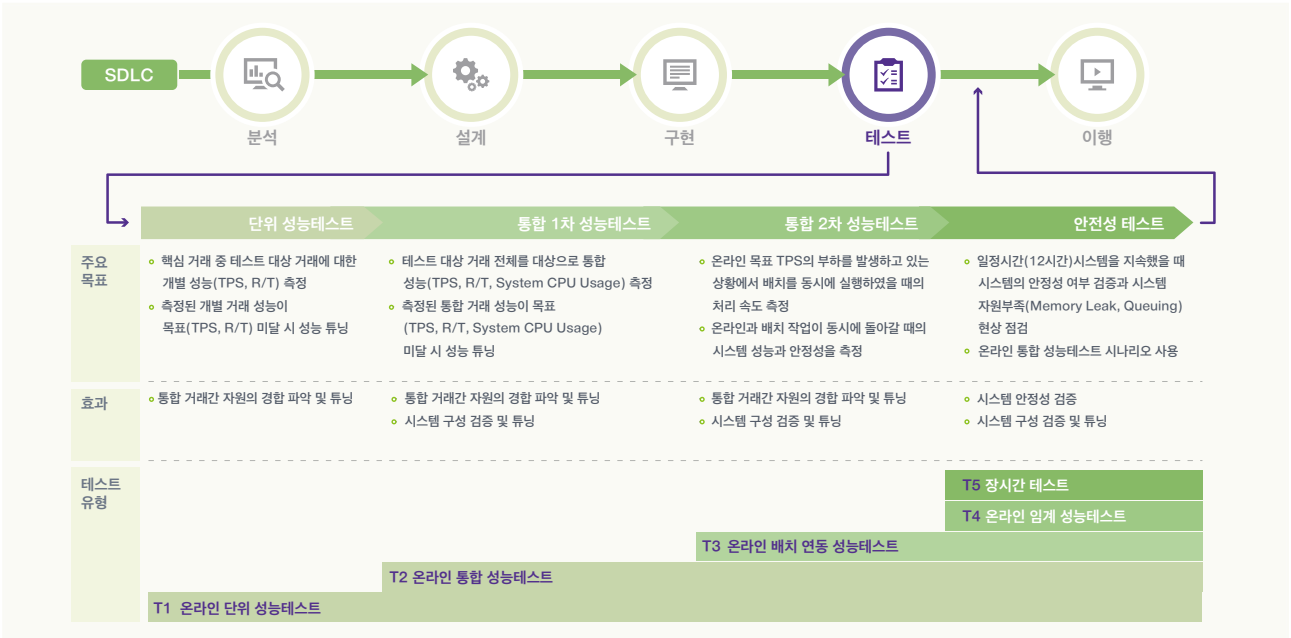
시스템 성능 검증 테스트는 검증 범위에 따라 단위 성능테스트, 통합 성능테스트, 임계 성능테스트, 장시간 테스트, 검증 대상에 따라서 온라인 테스트, 배치 테스트, 인터페이스 테스트 등으로 구분된다.

단위 성능테스트는 테스트 대상 단위 업무(온라인/배치/인터페이스)에 대한 성능 목표치를 달성할 수 있는지 여부를 검증하기 위해서 수행된다. 단위 업무에 대한 최대 업무량을 측정하기 위해서 스트레스 테스트 형태로 진행되며, 측정된 최대 업무량과 단위 성능테스트 목표와의 비교하여 목표에 미달할 경우 튜닝 대상 업무로 식별된다. 단위 성능테스트는 본격적인 통합 성능테스트를 수행하기 위한 사전 단계로 수행되기 때문에, 설사 목표 달성을 하지 못한다고 하더라도 너무 오랜 기간 동안 성능 튜닝에 매달릴 필요는 없다.

통합 성능테스트는 테스트 대상 전체 업무들에 대한 성능 목표치를 달성할 수 있는 지 여부를 검증하기 위해서 수행된다. 실제 운영 상황에 가까운 부하 상황을 재현하기 위해서 부하 테스트 형태로 진행 되며, 본 테스트를 통해 구축된 시스템에 대한 최종 성능을 검증하게 된다. 실제 운영 환경에서는 온라인 업무와 더불어 배치, 인터페이스 업무들이 함께 처리되기 때문에, 총체적인 관점에서 접근하는 자세가 반드시 필요하다.

임계 성능테스트는 테스트 대상 시스템이 SLA(service level agreement)를 만족하는 수준에서 최대로 처리할 수 있는 임계 성능을 산출하기 위해서 수행된다. 시스템 자원을 임계 상황(일반적으로 70%)전까지 사용하게끔 부하를 주기 위해서 스트레스 테스트 형태로 진행되며 이때 임계 성능 수치를 구하게 된다. 만약 시스템 성능 병목 현상을 야기하는 원인이 시스템 자원이 아닌 경우에는 지속적인 튜닝을 통해서 시스템 자원을 최대한 사용하게끔 튜닝 한 이후에 재 검증을 해야 한다.

장시간 테스트는 시스템 운영 중에 발생 가능한 메모리 누수나 예기치 못한 오류들을 사전에 검증하기 위해서 수행된다. 운영 상황을 시뮬레이션 하기 위해서 통합 성능테스트 부하 모델의 약 70%에 해당하는 부하 환경을 기준으로 24시간 정도 수행한다.



4 | 성능 테스트 목표

프로그램 개발이 완료된 업무에 대한 기능상에 문제 여부를 검증하는 기능 테스트와는 달리, 성능 테스트는 프로젝트 성능 목표에 대한 도달 여부를 검증하기 위해서 수행된다. 일반적으로 프로젝트 성능 목표는 다음과 같이 업무량, 응답시간, 시스템 사용률 등으로 구성된다.

4.1. 업무량 throughput

성능 테스트 목표 중에서 가장 중요하게 여겨지는 항목은 다름아닌 업무량이다. 사용자가 가장 많이 몰리는 피크 시간 동안에 시스템이 처리해야 하는 단위 시간 당 최대 업무 처리 건수를 의미하는 것으로써, 시스템 운영시의 최악의 부하 조건을 고려하여 부하량 프로파일(**Workload Profile**)을 정의 하게 된다.

차세대 정보 시스템 구축과 같은 대형 프로젝트에서는 업무량을 계산할 때 많은 노력이 요구되는데, 아래 그림과 같이 교체 대상이 되는 현행 시스템에 대한 기존 업무량 조사, 신규 시스템과의 업무 매핑 그리고 년간 업무 증가율에 대한 조사 등이 필요하다. 특히 업무 매핑의 경우 기존 업무에 대한 통폐합, 신규 업무 추가, 업무비율 조정 등도 고려되어야 하기 때문에 현행 시스템을 자세히 이해하는 업무 담당자의 지원이 절대적으로 필요하다.

한편 현행 시스템 혹은 신규 시스템 중 하나가 솔루션 패키지를 사용할 경우에는 업무량 계산이 더욱 어려워진다. 사용자 지정(커스텀) 개발의 경우 업무 담당자의 지원을 받을 경우 업무량 계산을 위한 트랜잭션 분석이 가능한 반면, 솔루션 패키지의 경우에는 업무 담당자 또는 솔루션 컨설턴트가 지원한다고 해도 트랜잭션 분석 자체가 힘들다. 이런 경우에는 트랜잭션 분석 건수를 기반으로 업무량을 계산하기 보다는, 업무 담당자와의 인터뷰를 통해서 다년간의 경험치에 의한 각 업무별 처리 건수를 정의하는 방법도 고려해 볼만 하다.

업무량 목표는 테스트 대상에 따라 단위 성능테스트 목표와 통합 성능테스트 목표로 구분된다.

4.1.1. 단위 성능테스트 목표

단위 성능테스트 목표는 각 단위 업무에 대한 성능 목표를 의미하는 것으로, 테스트 수행을 통하여 측정된 최대 업무량과 비교하는 기준이 된다. 여기서 단위 성능테스트는 단위 업무에 대한 최대 업무량을 측정하는 스트레스 테스트로써, 업무량만이 테스트 목표 기준이 된다.

단위 성능테스트 목표치는 테스트 대상 시스템에서 단위 업무가 단독으로 수행된다는 가정하에 정의되기 때문에, 현행 시스템에서 측정된 실제 업무량 보다는 훨씬 크게 설정되어야 한다.수치적으로 따져 볼 때 각 대상 업무에 대한 단위 성능테스트 목표는 측정된 실제 업무량보다는 크고 전체 업무들에 대한 업무량 보다는 작은 특성을 가지고 있는데, 이 범위 내에서 어떤 값으로 정할지에 대해서는 많은 고민이 필요하다. 현재까지 가장 널리 이용되는 단위 성능테스트 목표 설정 방법은 실제 업무량이 가장 많은 단위 업무와 전체 업무들에 대한 업무량과의 비율을 구하여, 이 비율을 모든 업무들에 동일하게 적용하여 목표를 수립하는 것이다.

4.1.2. 통합 성능테스트 목표

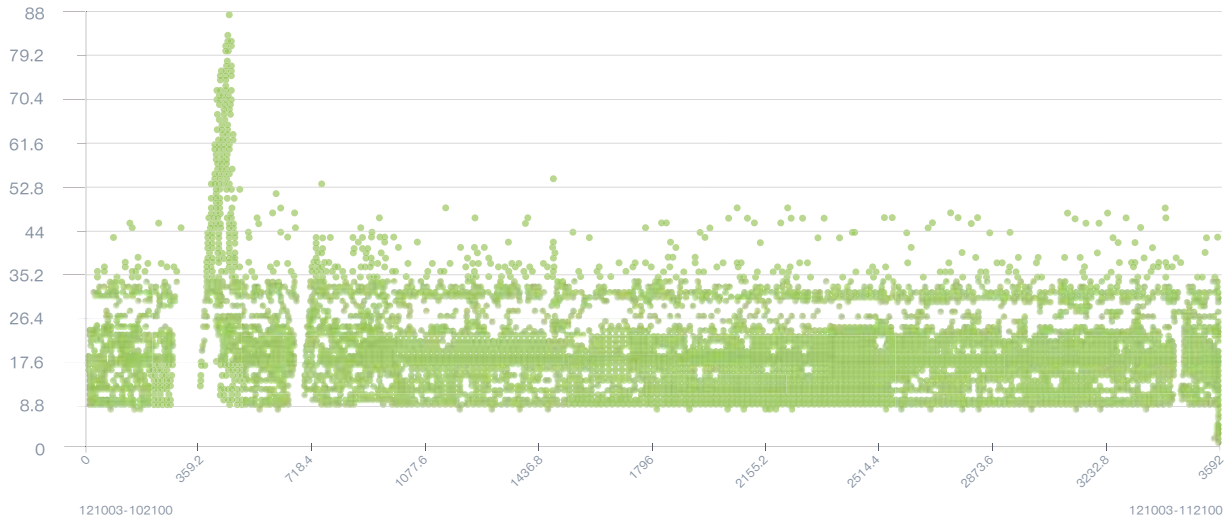
통합 성능테스트 목표는 테스트 대상 전체 업무들에 대한 통합 성능 목표를 의미하는 것으로, 테스트 수행을 통하여 측정된 최대 업무량과 비교하는 기준이 된다. 여기서 통합 성능테스트는 실제 운영 상황에 가까운 부하 상황을 재현하기 위한 부하 테스트로써 업무량뿐만 아니라 응답시간, 시스템 사용률에 대한 목표 달성 여부도 중요하다.

여기서 교체 대상이 되는 현행 시스템이 여러 개 존재할 경우, 각 개별 시스템의 피크 일자에 대한 업무량 분석을 통해서 여러 개의 통합 성능테스트 목표를 정의할 수도 있다. 하지만 아무리 피크 일자라고 하더라도 매번 처리해야 하는 업무량은 조금씩 틀릴 수가 있기 때문에, 최악의 조건을 고려하여 각 개별 시스템 별 최대 업무량을 기준으로 하나의 통합 성능테스트 목표를 설정하면 된다.

4.2. 응답시간 Response Time

앞서 소개한 업무량의 경우 시스템 측면에서의 성능 목표가 되는 반면, 사용자 측면에서는 응답시간이 성능 목표 기준이 된다. 응답시간은 업무 처리에 소요되는 총 시간을 말하는 것으로써, 사용자가 요청을 보낸 시점부터 처리 결과가 사용자 PC의 화면에 보여질 때까지 걸린 시간을 의미한다. 최근 들어서는 정보 기술의 발달과 더불어 인터넷의 폭발적인 확산으로 인해서 더욱 빠른 응답시간이 요구되는 추세인데, 과거 8초가 사람들이 최대한 인내할 수 있는 시간으로 여겨졌다면, 최근 들어서는 1-2초로 짧아지고 있다.

응답시간은 통계학에서 사용되는 대표 값 유형에 따라서 평균, 중앙값, 백분위수 등으로 표시될 수 있다. 이들 중 가장 많이 사용되는 형태는 평균, 백분위수이며, 최근 들어서는 이상치outlier에 따른 응답시간 편차가 큰 경우를 대비해서 가급적 90 백분위수를 권장하고 있다. 예를 들어 아래 그래프와 나와 있는 응답시간 분포와 같이 측정 도중에 이상치가 발생하여 응답시간의 편차가 커질 경우, 대표 값으로 평균을 사용하게 되면 왜곡된 수치가 나올 수 있다.



4.3. 시스템 사용률 System Resource Usage

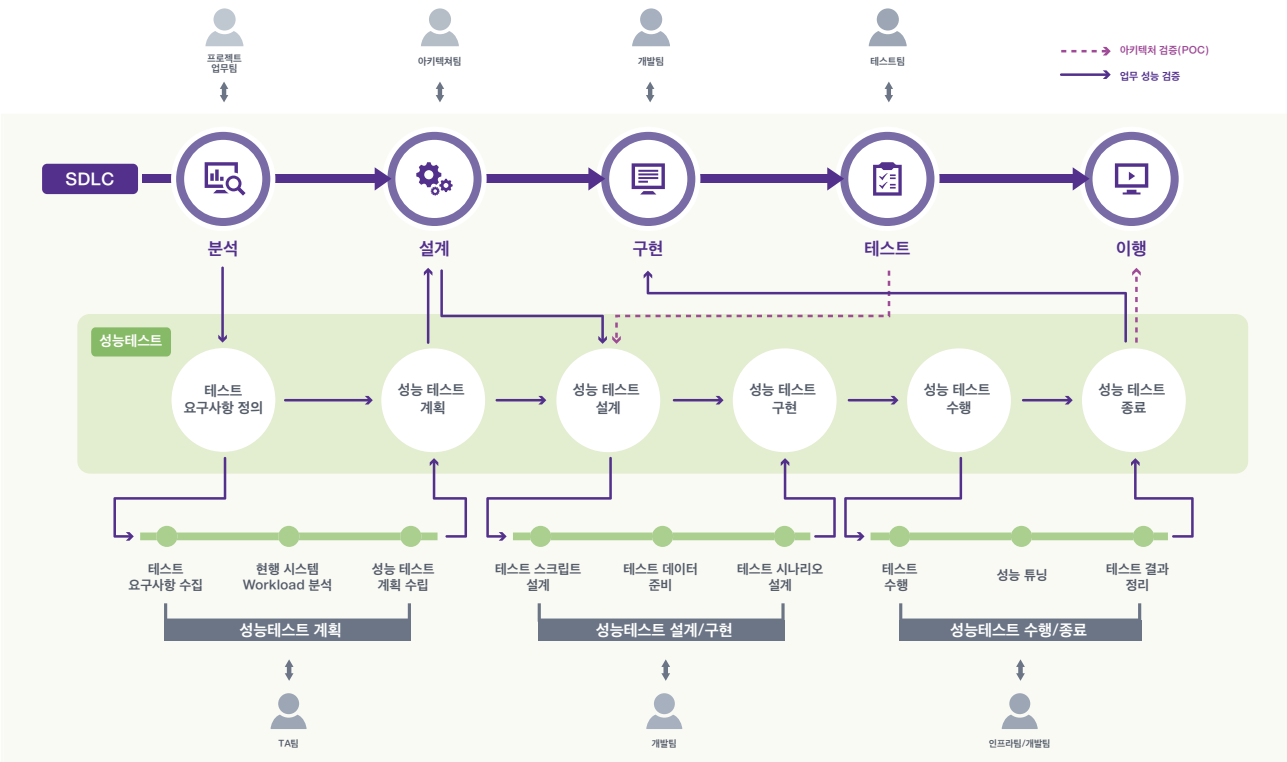
일반적으로 각 기업들은 정보 시스템의 하드웨어 용량 산정 시에 시스템 증설 여부를 결정짓는 임계 시스템 자원 사용률을 정하고 있다. 이는 시스템 자원 특히 CPU 사용률이 어느 임계 지점을 넘어 설 경우 응답시간이 기하급수적으로 높아지는 현상에 기인하고 있으며, 앞서 기고한 성능 분석 모델에 그 원인을 자세히 설명하고 있다.

각 기업들마다 조금씩의 차이는 보이고 있지만, 대부분의 경우 대부분의 경우 CPU 사용률에 대한 목표치는 60~70% 내외로 정의된다. 반면 ORACLE RAC의 경우 한쪽 노드의 장애발생시 다른 쪽 노드를 이용한 서비스가 가능하게끔 구성하기 위해서, CPU사용률 목표치는 RAC 노드 수에 따라서 상이하게**(2**대는 **30%~35%, 3**대는 **45%~50%)** 설정되어야 한다.

5 | 성능 테스트 프로세스

최근 들어 소프트웨어 품질에 대한 관심이 커지면서 많은 기업들이 소프트웨어 품질 평가모델(ISO 25000 SQuaRE)을 비롯한 테스팅 표준화 동향에 적극 동참하고 있다. 하지만 기업들의 적극적인 노력에도 불구하고, 성능 테스트 영역의 경우 성능 테스트 자동화 솔루션과의 긴밀한 의존성에 기인하는 특수성으로 인해서 아직까지도 일반인들의 이해도가 많이 떨어지는 실정이다. 하지만 성능 테스트 진행을 위해서는 테스트 프로세스에 대한 기본적인 이해는 반드시 필요하기 때문에 여기서는 간략하게나마 상위 레벨에 대한 프로세스를 소개하려고 한다.

성능 테스트 프로세스는 아래 그림과 같이 3가지 단계로 구성되어 있다.



첫번째 성능 테스트 계획 단계는 테스트 관련 요구사항 수집, 테스트 대상 시스템에 대한 부하량 분석, 테스트 계획 수립 등의 태스크로 구성되어 있다. 요구사항 수집 단계에서는 프로젝트 분석 단계에서 정의된 비 기능 요구사항을 기준으로 테스트 범위, 테스트 목표에 대한 상위 레벨 기대치를 수집한다. 부하량 분석 단계에서는 테스트 대상 업무, 업무량, 동시 사용자 수, 호출 간격 등을 포함하는 부하량 프로파일workload profile을 정의하기 위해서 테스트 대상 시스템 또는 레거시 시스템에 대한 사용자가 가장 많이 몰리는 피크 일자의 부하량을 분석한다.

NO	출처	대분류	중분류	프로그램 ID	유형	프로그램 설명	트랜잭션 볼륨	동시 사용자	응답 시간	거래비율	LR사용자	Pacing Time
01	INTERVIEW	고객	고객검색	고객 통합 검색	ONLINE	다양한 조건의 검색	3,000	20	6	0.13	33	40
02	INTERVIEW		고객등록	고객 정보 등록	ONLINE	고객 정보 등록 프로그램	200	20	6	0.01	2	40
03	INTERVIEW		고객수정	고객 정보 수정	ONLINE	고객 정보 수정 프로그램	200	20	6	0.01	2	40
04	INTERVIEW	벤더	벤더등록	면세 가이드 관리	ONLINE	면세 가이드 등록 및 상신	500	10	6	0.02	6	40
05	INTERVIEW		벤더관리	면세 agent관리	ONLINE	면세 agent등록 및 상신	500	10	6	0.02	6	40
06	INTERVIEW		벤더검색	벤더 통합 검색	ONLINE	다양한 조건에 검색	3,000	20	4	0.13	33	40
07	INTERVIEW	면세상품	상품관리	상품등록	ONLINE	신규상품등록	1,500	30	6	0.07	17	40
08	INTERVIEW		상품관리	상품수정	ONLINE	상품수정	1,500	30	6	0.07	17	40
10	INTERVIEW		상품관리	상품검색	ONLINE	다양한 조건에서 상품검색	7,000	50	4	0.31	78	40
11	INTERVIEW	레시피	레시피정보관리	레시피등록	ONLINE	레시피 등록	1,000	20	6	0.04	11	40
12	INTERVIEW		레시피정보관리	레시피수정	ONLINE	레시피 수정	1,000	20	6	0.04	11	40
13	INTERVIEW		레시피정보관리	레시피검색	ONLINE	다양한 조건에서 레시피검색	3,000	30	4	0.13	33	40

성능테스트 계획 수립 단계에서는 테스트 목적, 테스트 범위, 테스트 목표, 테스트 대상 업무, 테스트 일정, 테스트 조직, 테스트 프로세스, 시작/종료 조건 등에 대한 정보를 기반으로 테스트 계획 문서를 작성한다. 성능테스트는 개발팀을 포함한 여러 이해 관계자들의 지원이 필요하기 때문에, 성능테스트 계획서는 설명회를 통해서 반드시 공유되어야 한다.

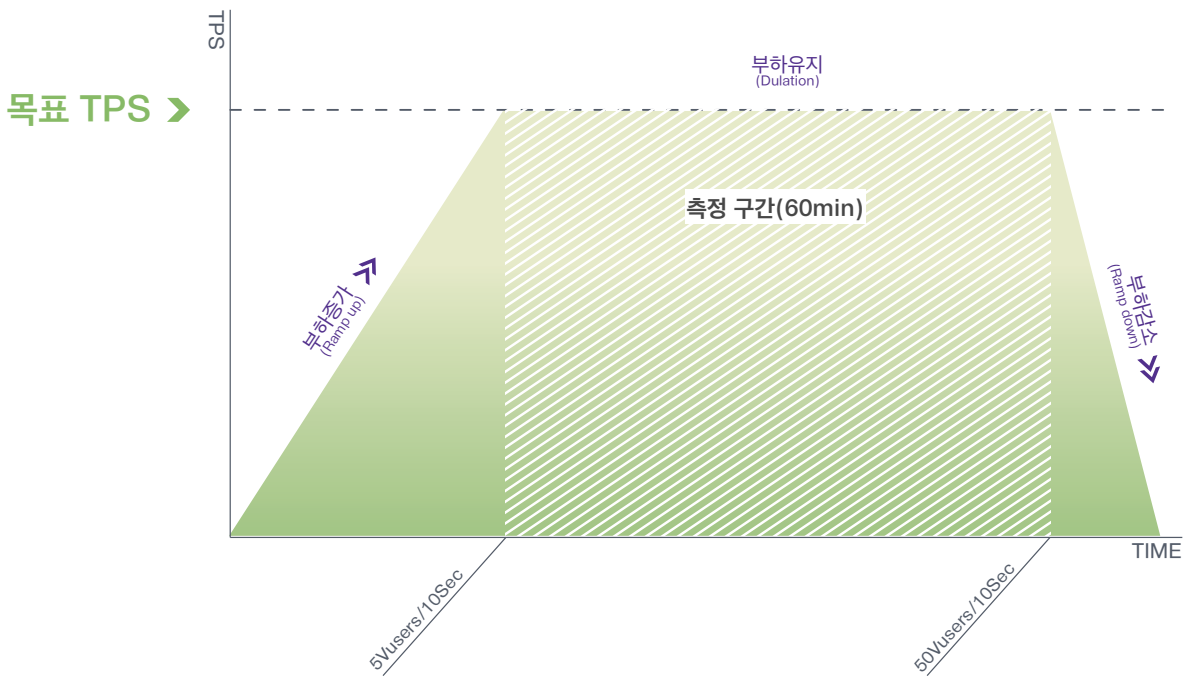
두번째 성능 테스트 설계/구현 단계는 테스트 대상이 되는 업무들에 대한 테스트 스크립트 설계, 테스트 데이터 준비, 테스트 시나리오 설계 등의 태스크로 구성되어 있다. 테스트 스크립트 설계 단계에서는 테스트 대상 업무를 위한 테스트 스크립트가 생성되는데, 일반적으로 테스트 대상 업무별로 별도로 작성하는 것을 원칙으로 한다.

테스트 데이터 준비 단계에서는 테스트 대상 업무에서 필요로 하는 볼륨 데이터를 준비하는데, 원칙적으로 레거시 시스템에서 추출된 데이터를 사용해야 한다. 이때 추출된 데이터 볼륨이 부족하거나 추출된 데이터가 없을 경우에는 테스트 데이터를 별도로 생성하게 된다. 하지만 인위적으로 생성된 테스트 데이터는 실제 상황에서 발생하는 다양한 조건들을 만족할 수 없기 때문에, 테스트 결과에 대한 신뢰도를 낮출 수 있음을 유념해야 한다.

테스트 시나리오 설계 단계에서는 부하량 분석 단계에서 정의된 부하량 프로파일을 기반으로 성능 테스트 솔루션에 시나리오를 생성하게 된다. 성능 테스트 자동화 솔루션에는 실제 상황과 유사하게 부하를 생성할 수 있는 여러 기능(가상 사용자, 호출 간격, 대기 시간)들을 제공하기 때문에, 이들 개념들에 대한 이해는 반드시 필요하다.

테스트 시나리오 설계 단계에서는 부하량 분석 단계에서 정의된 부하량 프로파일을 기반으로 성능 테스트 솔루션에서 시나리오를 설계하게 된다. 성능 테스트 솔루션에는 실제 상황과 유사하게 부하를 생성할 수 있는 여러 기능(가상 사용자, 호출 간격, 대기 시간)들을 제공하기 때문에, 테스트 담당자는 이들 개념들에 대한 사전 이해를 해야 한다.

테스트 시나리오 설계 단계에서 고려해야 할 또 다른 중요한 포인트는 시나리오 패턴이다. 일반적으로 테스트 대상 시스템에 워밍업 없이 급격한 부하를 가하게 되면, 요청된 업무를 정상적으로 처리하지 못하는 문제가 발생할 수 있다. 성능 테스트 솔루션에서는 이러한 문제를 극복하기 위해서 세가지 부하 구간을 제공하여 실제 상황과 유사한 형태의 부하 생성을 가능하게 한다. 여기서 첫번째 부하 구간은 부하 증가(**ramp up**)구간으로써 지정된 시간마다 지정된 사용자가 점진적으로 증가하는 형태를 띠게 하여, 시스템이 워밍업 할 수 있게끔 해 준다. 두번째 부하 구간은 부하 유지(**duration**)구간으로써 사용자가 더 이상 증가하지 않고 동일 부하가 그대로 유지된 채 진행되게 해 준다. 실제 테스트 결과를 측정하는 구간이기 때문에 각 테스트 유형별로 충분한 시간(통합 성능테스트의 경우 1시간)을 가지게끔 정의가 필요하다. 마지막 부하 구간은 부하 감소(**ramp down**)구간으로써 지정된 시간마다 지정된 사용자가 점진적으로 감소하는 형태를 띠게 하여 부하량을 감소시키게 한다.



세번째 성능 테스트 수행단계에서는 각 테스트 유형 별 테스트 수행, 성능 미달 시 성능 튜닝 수행, 테스트 결과 정리 등을 진행하게 된다.

테스트 수행 단계에서는 테스트 계획상에 나와 있는 각 테스트 유형 별 접근 방안을 가지고 테스트를 진행하게 되는데, 테스트 수행 결과가 테스트 목표를 달성할 때까지 성능 튜닝과 병행하여 반복 수행하게 된다.

성능 튜닝 단계에서는 테스트 수행 결과가 테스트 목표에 도달하지 못할 경우, 병목을 야기하는 근본 원인을 분석하고 제거하는 활동을 진행해야 하는데, 시스템 아키텍처가 복잡하고 대형화 될 수록 튜닝을 담당하는 전문 인력은 반드시 필요하다.

테스트 결과 정리 단계에서는 테스트 계획상의 각 테스트 유형 별 목표와 측정 결과를 비교 분석하여 테스트 결과를 정리하게 된다. 이때 테스트와 관련된 상세 현황(테스트 조건, 테스트 상세 결과 파일, 시스템 현황 정보, 시스템 구성 내역)에 대한 정리도 반드시 필요하다.

6 | 업무 성능 모니터링 Application Performance Monitoring

성능 테스트를 논의할 때 간과해서는 안될 중요한 부분은 성능 모니터링이다. 테스트를 수행하는 동안 테스트 대상 시스템을 구성하는 주요 컴포넌트들(시스템, 네트워크, 웹 어플리케이션 서버, 데이터베이스 서버)에 대한 주요 성능 측정 항목을 주기적으로 수집하고 분석하는 활동은 반드시 필요하다.

전통적으로 성능 모니터링은 엔터프라이즈 IT 관리의 일환으로 시스템, 네트워크를 포함한 인프라 측면에서 주로 진행되어 왔고, 최근 들어서는 어플리케이션을 포함한 웹 어플리케이션 서버, DBMS 서버 영역으로 점차적으로 확대되고 있다.

한편 정보 시스템 환경이 점차적으로 대형화 되고 복잡해 지고 있는 오늘날의 현실을 고려해 볼 때, 업무 성능 모니터링 체계도 이러한 변화에 대응할 수 있는 다양한 컴포넌트들을 연계하는 통합 모니터링 체계를 지향하는 진화가 요구된다.

현재 국내 J2EE기반 APM 시장에서 독보적인 위치를 점유하고 있는 제니퍼의 경우에도 이러한 트랜드에 맞춰서 다음과 같은 최신 기능들을 제공하고 있다.

7 | 마치며

대형 프로젝트들이 떠안고 있는 가장 큰 숙제인 동시에 반드시 극복 해야 하는 난관인 시스템 성능 문제는 시스템 용량 계획/산정, 성능 테스트, 성능 모니터링, 성능 튜닝 등 다양한 각도에서 접근이 필요하다.

먼저 시스템 용량 관점에서는 원활한 비즈니스 지원을 위해서 요구되는 시스템 처리 성능을 여러 제반 여건을 고려하여 올바르게 계획하고 시스템화 하는 노력이 필요하다. 이와 더불어 구축된 시스템에 대해서는 실제 상황에 근거를 둔 워크로드 프로파일을 기반으로 아키텍처 관점과 시스템 성능 관점에서의 성능 검증, 성능 모니터링, 성능 튜닝이 필요하다.

본 고에서 다루고 있는 성능 테스트 개론이 시스템 성능 문제를 고민하고 대안책을 마련하고자 하는 많은 이들에게 도움이 되었으면 한다.

8 | 저자에 대하여

박성훈 컨설턴트는 BTO 컨설팅의 애플리케이션 품질 관리 대표 컨설턴트로 활동하고 있다. 다년간 엔터프라이즈 환경하에서 성능 벤치마크, 성능 테스트, 품질 관리 등의 업무를 담당하였다. BTO 컨설팅 이전에는 한국 HP, 머큐리 인터랙티브 코리아 등에서 프로페셔널 서비스 컨설턴트로 역할을 수행한 바 있다.

박성훈 컨설턴트(Bruce Park)
Consultant, mail : bruce@jennifersoft.com

about JenniferSoft...

JenniferSoft, Inc. is the software vendor company with expertise in application performance monitoring and performance problem resolution, providing Application Performance Management (APM) solutions and services to enterprise companies around the world. Technology is foremost of what's valued in JenniferSoft, as we strive to bring to the market the latest and best technology available. We think that software solution should not be just a mesh of form and functions but it should be designed with its user in mind, each component formed with intent to elevate the experience of its user. JenniferSoft combines latest technology in APM with field-tested expertise and experience to bring to our customers a well-balanced solution that is both advanced in features and practical. With vision of combining technology and experience into one, JenniferSoft pledges to continue focuing on R&D to develope world class solution.