Black Box 방식의 EJB 컴포넌트 성능 측정

황길승*0 이긍해* 권오천** 신규상**

*한국항공대학교 컴퓨터공학과
**한국전자통신연구원

{avi94^0, khlee}@mail.hangkong.ac.kr {ockwon,gsshin}@etri.re.kr

A Black Box Based Performance Testing of EJB Components

Kil-Seung Hwang* Keung Hae Lee* Oh-Cheon** Kwon Gyu-Sang Shin**

*Department of Computer Engineering, Hankuk Aviation University, Koyang-si, Kyunggi-Do
** ETRI(Electronics and Telecommunications Research Institute), 161 Gajeong-dong, Yuseong-gu Daejon

요 약

컴포넌트 기반 소프트웨어 개발에서 구현에 사용되는 각 컴포넌트의 성능을 이해하는 것은 중요하다. 본 논문은 Black Box의 관점에서 EJB 컴포넌트의 성능을 측정하는 방법과 이 측정 방법을 지원하는 성능 측정 도구의 설계 구현에 대하여 설명한다. 본 도구는 대상 컴포넌트에 대한 영향을 최소화하기 위하여 클라이언트 시스템에서 동작하여 성능을 측정할 수 있도록 설계 되었다. 또한 사용자가 적은 노력으로 새로운 컴포넌트의 성능을 측정할 수 있도록 테스트 프로그램을 자동으로 생성한다. 본 연구의 결과는 상용 컴포넌트의 시장에 등장하는 여러 컴포넌트의 성능을 객관적인 방법으로 비교 측정 할 수 있는 효과적인 수단이 될 것으로 기대된다.

1 서론

특정 기능별로 모듈화된 소프트웨어 컴포넌트를 조립하여 소프트웨어를 개발하는 CBD(Component Based Development) 모델은 현 소프트웨어 공학의 개발방법론 측면에서의 하나의 커다란 흐름이다. 이와 더불어 개발된 컴포넌트 기반 소프트웨어를 효과적으로 테스팅할 수 있는 방법은 좀 더 질 높은 소프트웨어를 개발하기 위한 노력의 하나로 그 중요성이 증가하고있고 많은 곳에서 이에 대한 연구가 활발히 진행되고 있다.

컴포넌트 기반 소프트웨어 개발방법론이 보다 보편화 되기 위해서는 이를 객관적으로 검증하고 시험할 수 있는 방법들이 확립되어야 한다. 컴포넌트를 시험한다는 것은 컴포넌트 개발자에게는 개발된 컴포넌트에 대한 신뢰성, 유효성을 검증하는 수단이 되고 컴포넌트를 사용하는 사용자의 입장에서는 보다 효율적인 성능을 가지는 컴포넌트를 선택하기 위한 객관적인 지표를 제시할 수 있기 때문에 중요하다.

본 논문에서는 컴포넌트 기반 개발 모델 중 EJB(Enterprise Java Beans) 컴포넌트의 성능측정 방법에 대해서 설명한다. EJB는 Sun 사에서 발표한 Java를 기반으로 한 컴포넌트 기반 개발모델의 하나이다.

본 논문에서는 Black Box 모델에 기초한EJB 컴포넌트 성능측정도구에 대하여 설명한다. 그리고 기존 성능측정 도구와의 차이점을 설명하고 도구의 설계 및 구현에 대해 설명한다. 1장 서론에서는 EJB 컴포넌트 성능측정 도구의 배경에 대해서 설명하고 2장 관련연구에서는 이전의 연구에 대하여 설명하며 우리연구와의 차이에 대하여 설명한다.. 3장 EJB 컴포넌트 성능측정도구의 설계 및 구현에서는 실제로 EJB 컴포넌트의 성능측정을 위한 도구의 접근방법과 구조에 대해서 설명하고 4장 결론에서는 향후 연구에 대해서 기술한다.

2 관련연구

현재 개발된 EJB 컴포넌트 성능측정도구로는 Flashline사의 Flashline QA Lab, TestMyBean사의 TestMyBean등의 도구들이 있다. *Flashline*사의 *Flashline QA Lab*은 소스코드 구조의 분석을 통해서 측정한 메모리 사용률과 실행시간을 측정하여 성능과 유효성을 검증한다. 그리고 TestMyBean사의 TestMyBean은 Response time per Method, Number of Exception등의 효과적인 성능측정요소들을 가지고 있다. 이러한 도구들은 공통적으로 소스레벨의 정보를 필요로 하는 White Box Testing방법을 사용한다. 컴포넌트의 성능측정에 White Box Testing 방법을 사용하면 바이너리형태의 컴포넌트를 역컴파일 또는 그에 준하는 서버에 영향을 주는 동작을 취하게 된다는 문제점을 가지고 있다.[8][9]

이미 배치되어 있는 컴포넌트의 성능측정은 컴포넌트의 동작과는 완전히 독립된 형태로 수행되어야 하며 성능측정이 컴포넌트의 동작 및 상태에 어떠한 영향이라도 미쳐서는 안된다.

본 논문의 EJB 컴포넌트 성능측정도구는 Black Box Testing 방법을 사용하여 기존의 제품들이 가지는 이러한 문제점들을 해결하였다. 성능측정은 일반적인 EJB 클라이언트의 형태를 가지는 성능측정 테스트 프로그램을 생성하여 실행함으로서 수행된다. 이는 기존의 컴포넌트에 어떠한 영향도 주지않고 단지 컴포넌트에

접근하여 결과를 얻어내는 동작만으로 성능을 측정함으로써 컴포넌트와 서버에 주는 부하를 최소화 하는 효과를 얻고 독립적인 성능측정작업을 수행할 수 있도록 하였다.

3 EJB 컴포넌트 성능측정도구의 설계 및 구현

3.1 EJB컴포넌트 성능측정도구의 접근방법

EJB 컴포넌트의 성능을 측정하기위한 사용자 시나리오는 다음과 같다.

1. 사용하고자 하는 컴포넌트가 배치되어 있는 서버의 정보, 즉 서버의 IP Address 와 프로토콜, 그리고 포트번호를 입력한다.
2. 컴포넌트에 대한 리모트, 홈 인터페이스 이름, JNDI 이름을 입력한다.
3. 테스트에 사용할 비즈니스 메소드를 선택하고 반복실행횟수, 메소드 파라미터등의 정보를 입력한다.
4. 테스트를 시작한다.
5. 결과를 그래프로 확인한다.
6. 여러 컴포넌트의 결과를 서로 비교하여 보다 효율적인 컴포넌트를 선택한다.

EJB 컴포넌트 성능측정 시나리오는 일반적인 컴포넌트 성능측정 시나리오와 유사하다. 사용자는 컴포넌트의 성능을 측정하기 위한 기본적인 정보를 입력하여야 한다. EJB 컴포넌트 성능측정 도구는 성능측정과정에 사용자의 주관적인 요소를 더하기 위해서 메소드별 실행반복횟수나 파라미터 값을 임의로 조정할 수 있도록 하였다.

성능측정은 네가지 성능측정요소를 중심으로 수행된다. 네가지 성능측정 요소는 다음과 같다.

- 메소드별 응답시간
- 컴포넌트 응답시간
- CPU 사용률
- 메모리 사용률

메소드별 응답시간은 컴포넌트의 비즈니스 메소드를 실행하였을 때의 응답시간을 측정한다. 컴포넌트 응답시간은 컴포넌트의 모든 비즈니스 메소드들의 응답시간의 합으로 구성된다. 이는 동일한 기능을 하는 컴포넌트가 얼마나 신속하게 결과를 도출하는지를 비교할 수 있는 요소가 될 수 있다. CPU사용률과 메모리 사용률은 서버측의 측정과 클
라이언트 측의 측정으로 나눌 수 있다.

서버측의 CPU및 메모리 사용률은 서버에서 구동되는 별도의 Agent형태의 데몬 프로그램과 통신을 통해서 이루어진다. 이는 컴포넌트가 실행됨에 따라 시스템 리소스를 얼마나 차지하는지에 대한 평가요소로 활용될 수 있다. 이 네가지 성능측정 요소는 컴포넌트의 성능을 객관적이고 효율적으로 측정하고 결과를 비교하여 우수한 컴포넌트를 선택할 수 있는 기준을 제시하는데 도움을 준다.

3.2 EJB 컴포넌트 성능측정도구의 구조

EJB 컴포넌트 성능측정도구는 크게 세부분으로 나뉘어진다. Input Dataset Generator와 Test Program Generator, 그리고 Result Reporter가 그것이다.

우선 Input Dataset Generator는 컴포넌트와 서버에 대한 입력된 정보를 저장하고 인터페이스 파일을 분석하여 비즈니스 메소드의 정보를 추출해내는 역할을 한다. 그리고 추출된 비즈니스 메소드의 파라미터 정보를 분석하여 임의로 정의된 데이터 집합중 임의의 값을 디폴트로 적용한다. 이는 사용자의 수정이 가능하도록 구성된다. 요약하면Input Dataset Generator는 컴포넌트의 성능을 측정하기 위한 모든 입력값을 관리하기 위한 역할을 제공한다.

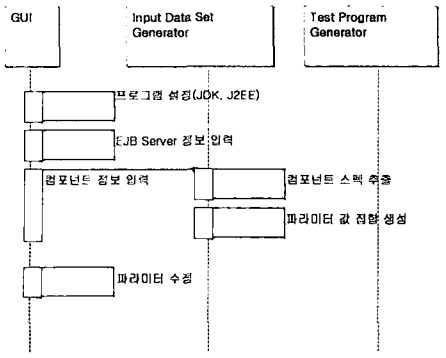

그림 1. Input Dataset Generator의 순차도

입력된 모든 정보들은 Test Program Generator에서 Test Program을 생성하기 위한 값으로 사용된다. Test Program Generator는 Input Dataset Generator에서 처리한 데이터들과 EJB 클라이언트 프로그램 템플릿을 사용하여 Test Program을 생성한다. 생성된 Test Program은 네가지 성능측정요소를 위한 코드를 가지고 있으며 실행되면서 네개의 성능측정결과가 기록되는 로그파일을 생성하게 된다.

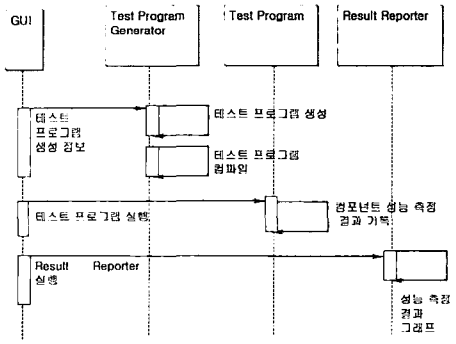

그림 2. Test Program Generator와 Result Reporter의 순차도

생성된 결과는 Result Reporter에서 사용자가 이해하기 쉬운 그래프의 형태로 재가공되어 보여지게 된다.

성능측정 시나리오에서 또하나 중요한 요소중 하나는 예전에 측정했던 성능을 다시 확인하는 기능이다. 이를 위해 성능측정시 자동으로 프로젝트 파일을 생성하여 이후의 성능측정 결과확인이 가능하도록 하였다. 프로젝트 파일에는 성능측정시 사용되었던 서버와 컴포넌트 관련 데이터들과 각종 관련 정보들이 포함되어 있고 이를 적용함으로서 간단하게 예전의 작업들을 확인할 수 있도록 하였다.

3.3 EJB 컴포넌트 성능측정도구를 이용한 성능측정

EJB 컴포넌트 성능측정도구를 이용하여 배치되어 있는 EJB 컴포넌트의 성능을 측정하기 위한 단계는 다음과 같다.

1. 클래스패스 설정 및 환경설정
2. 사용자의 서버정보 및 컴포넌트 정보 입력
3. Test Program 생성 및 실행
4. 결과 도출
5. 두개 이상의 컴포넌트의 성능측정값 비교

EJB 컴포넌트 성능측정도구를 실행하기 위해서는 j2ee의 라이브러리 파일을 클래스패스로 등록하는 간단한 설정을 필요로한다. 설정이 끝나면 EJB 컴포넌트 성능측정도구를 실행한다. 실행한 후 사용자는 EJB가 배치되어있는 서버에 대한 정보, 즉 서버접근 프로토콜과 서버의 IP Address, 그리고 포트번호를 입력하고 컴포넌트에 대한 정보, 즉 JNDI Name, 리모트, 홈 인터페이스 이름 등을 입력한다. 입력이 완료되고 저장되면 자동으로 리모트 인터페이스의 비즈니스 메소드의 리스트와 메소드 반복횟수, 해당하는 파라미터 데이터를 임의로 자동 생성하여 화면에 표시한다. 사용자는 이 정보를 사용자의 요구에 맞게 수정하여 저장할 수 있다.

데이터의 생성 및 수정이 끝나면 Test Program을 생성한다. Test Program은 자동생성하며 컴파일이 완료되었다는 메시지를 표시한다.사용자는 생성된 Test Program을 볼 수 있고 수정할 수 있고 다시 저장하며 재컴파일할 수 있다. 그리고 마지막으로 Test Program 실행버튼을 통해 Test Program이 실행되면서 성능측정 요소들에 해당하는 로그파일과 프로젝트 파일 등을 생성하게 된다.

이제 사용자는 결과보기를 통해서 결과를 그래프형태로 볼 수 있고 컴포넌트 성능비교기능을 통해서 각각의 성능측정값을 비교해 볼 수 있다.

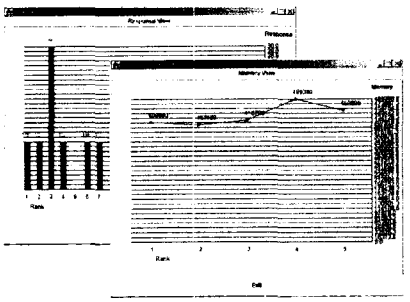

그림 3. 컴포넌트 성능측정 결과 그래프

4 결 론

EJB 컴포넌트의 성능측정은 성능 측정 프로그램이 컴포넌트에 미치는 영향을 최소화하는 것이 바람직하다. 본 논문에서는 원격지에 배치되어 있는 EJB 컴포넌트에 대하여 성능을 측정하는 방법에 대하여 설명하였다. 이 방법은 측정 대상 컴포넌트의 명세로부터 성능 측정 프로그램을 자동으로 생성하고 사용자가 필요한 테스트 데이터를 입력함으로써 측정에 필요한 비용을 줄일 수 있다. 본 논문은 이러한 측정 방법과 구현된 성능 측정 도구에 대하여 설명하였다. 현재는 본 측정 도구를 사용하여 웹에서 성능을 측정할 수 있도록 하기 위한 연구를 추진하고 있다.

참고문헌

[1] 한국 소프트웨어 컴포넌트 컨소시엄, "컴포넌트 산업육성 계획수립을 위한 컴포넌트 산업 및 기술 동향 조사", http://www.component.or.kr/, 12. 2000.

[2] Jerry Ga, "Component testability and component testing challenges", CMU Software Engineering, Carnegie Mellon University, 2000

[3] Jerry Ga, Eugene Y. Zhu, Simon shim, "Testing component-based software ", STARWEST '99, 1999

[4] 전태웅, "객체지향 프레임워크의 Built-in Test 방법", 소프트웨어공학기술 워크샵 2000 발표집, 1 권 1 호, pp.83-92, 8.2000

[5] 최병주, "EJB 컴포넌트의 맞춤테스트기법", 소프트웨어공학기술 워크샵 2000 발표집, 1 권 1 호, pp.93-102, 8. 2000

[6] Jerry Ga, Eugene Y. Zhu, Simon shim,"Monitoring Software components and component based software", San Jose State University, 1999

[7] 권오천, "공용컴포넌트 플랫폼 선정시 고려사항", 공용컴포넌트플랫폼선정을위한 공청회, pp.71-73, 3. 2000

[8] 오창남, 이긍해, Enterprise javaBeans(EJB) 컴포넌트의 성능 측정 방법, 추계학술 발표회, 한국정보과학회, 2000 년 10 월

[9] 오창남, 이긍해, Enterprise javaBeans(EJB) 컴포넌트의 성능 측정 시스템 설계, 2000 년 한국정보처리학회 추계학술 발표회, 한국정보처리학회,2000 년 10 월.

[10] "SourceGuard", URL http://www.4thpass.com/sourceguard/support/index.html, 4thpass Inc.

[11] S. J. Bayer, M. Devarakonda, S. Fink, E. Gluzberg, M. Kalantar, P. Muttineni, E. Barness, R. Arora, R. Dimpsey, and S. J. Munroe. "Java Server Benchmarks", IBM Systems Journal 39, No. 1, 57-81(2000, this issue)

[12] D. Viswanathan and S. Liang, "Java Virtual Machine Profiler Interface", IBM Systems Journal 39, No. 1, 82-95(2000, this issue)

[13] I. H. Kazi, D. P. Jose, B. Ben-Hamida, C. J. Hescott, C. Kwok, J. Konstan, D. J. Lilja, and P. C. Yew, "JaViz : A Client/Server Java Profiling Tool," IBM Systems Journal 39, No. 1, 96-117 (2000, this issue)