2001년 한국정보처리학회 추계 학술발표논문집 제8권 제2호

Wrapping 방법을 통한 EJB 컴포넌트 성능 측정 도구 설계

임병진, 황길승, 이동진, 이긍해
한국항공대학교 컴퓨터공학과
e-mail:nomja@mail.hangkong.ac.kr, avi94@mail.hangkong.ac.kr
bol022@mail.hangkong.ac.kr, khlee@mail.hangkong.ac.kr

The design of a wrapping based performance testing tool for EJB Component

Byungjin Lim, Gilseung Hwang, Dongjin Lee, Keung Hae Lee
Dept of Computer Engineering, Hankuk Aviation University

요 약

본 논문에서는 EJB 컴포넌트 성능 측정 도구 설계에 대하여 설명한다. 이전 연구에서는 EJB 컴포넌트를 역컴파일하는 방법을 사용하였으나, 역컴파일 방지 기술이 사용된 컴포넌트에 대해서는 성능 측정 할 수 없다는 문제를 가졌다. 또 성능 측정을 하기 위한 테스트 프로그램을 작성해야 하는 2차적인 노력이 필요하였다. 이러한 문제를 해결하기 위해 본 논문에서는 black box 성능 측정 기법을 사용하고, 테스트 프로그램을 자동 생성하여 성능 측정을 자동화하는 방법을 제시한다.

1. 서 론

최근 소프트웨어 개발에 있어 컴포넌트 형태로 개발하는 기술이 빠르게 발전하고 있다. 컴포넌트 형태로의 소프트웨어 개발은 재사용성을 높이고 개발 시간과 비용을 감소시키는 효과를 가질 수 있다. 또 개발 시간의 단축은 소프트웨어 시장을 선점할 수 있도록 빨리 상품화 할 수 있게 한다. 한국 소프트웨어 컴포넌트 컨소시엄에서 발표한 2000년 12월 자료에 의하면, CBD(Component Base Development) 기반 소프트웨어 개발 경험이 있는 업체는 조사 대상 중 75.7%를 차지하고 있다[1].

컴포넌트를 사용하여 개발한 응용프로그램은 컴포넌트의 성능에 영향을 받는다. 최적화되지 않은 컴포넌트는 응용프로그램의 성능을 떨어뜨리는 원인이 된다. 따라서 응용프로그램 개발자는 컴포넌트를 선택하기 전에 그 컴포넌트의 성능을 알아야 한다. 이때 컴포넌트 성능에 대한 자료는 객관성을 유지해야 한다[2].

본 논문에서는 분산환경에서 EJB 컴포넌트 성능을 측정하는 방법을 제안하고 이러한 측정을 수행하기 위한 도구를 설계한다. 2장에서는 기존의 연구와 문제점에 대해 설명한다. 3장에서는 본 논문에서 제안하는 컴포넌트 성능 측정 방법에 대해 설명한다. 4장에서는 결론 및 향후 연구에 대해 논한다.

2. 기존의 연구 및 문제점

2.1 기존 연구

2.2.1 컴포넌트 기반 소프트웨어 테스팅 방법

모듈화된 컴포넌트를 이용한 응용 프로그램은 컴포넌트 영향을 받는다. 따라서 이에 컴포넌트를 테스트하는 연구가 여러 연구실에서 진행중이다[2][4][5]. 테스트 요소로는 행동(behavior), 기능(function), 자원(resource), 커스터마이징(Customizing)등이다. 테스트 방법으로는 컴포넌트 안에 트레이스(trace)하는 모듈을 넣고 각각의 테스트 요소들을 검사한다. 커스터마이징 테스트는 배치

서술자(Deployment Descriptor)에 의해 화이트박스 부분을 테스트한다. 배치서술자는 응용 프로그램에 컴포넌트들이 어떻게 배치되었는지에 대한 조립정보를 가지고 있다. 이런 조립정보를 통해서 컴포넌트 사용자가 컴포넌트 속성(property)을 수정해서 컴포넌트 상태(state) 변경시킬 수 있다. 이때 조립정보를 변경했을 때 발생하는 오류여부를 테스트한다.

컴포넌트 성능측정방법도 컴포넌트 기반 소프트웨어 개발을 위한 테스트 방법처럼 다르게 성능측정방법을 적용해야 한다. 기존 성능 측정 방식은 바이너리 형태의 독립된 모듈 단위인 컴포넌트 단위로 성능측정을 하기보다는 응용프로그램 레벨에서 응용프로그램의 기능 및 제어들을 테스트하였다[2][3][6][7].

2.2.2 역컴파일을 통한 J2EE 컴포넌트 성능 측정 방법

AUTOPEC(AUTO tool of the Performance measuring for the j2Ee Component) 컴포넌트 자동 측정도구는 측정하는 컴포넌트의 소스코드 없이 바이너리 형태만으로도 컴포넌트의 성능을 측정할 수 있다.

AUTOPEC은 성능 측정하려는 J2EE 컴포넌트를 자동 측정한다. 측정하려는 컴포넌트를 선택했을 때 자동적으로 성능측정 요소가 삽입된다. 측정하는 컴포넌트를 수행하는 클라이언트 프로그램을 수행하면 자동적으로 컴포넌트 성능측정이 내부적으로 처리된다. 성능 측정 결과는 풀 크기에 각 메소드 처리에 따른 CPU 사용률, 측정시간에 따른 메모리 전체 크기와 사용하고 있는 메모리 크기, 남은 메모리 크기를 나타낸다. 또한 트랜잭션 처리 시간과 각 빈들의 처리시간을 자동적으로 측정한다. 그리고 각각 결과를 그래프로 자동 출력한다[8][9].

2.2 문제점

AUTOPEC의 성능 측정 방법은 컴포넌트의 성능 측정을 위하여 컴포넌트를 사용한 소프트웨어를 개발해야만 하는 2차적인 노력을 요구하게 된다. 또 최근에는 java class 파일에 대한 역컴파일 방지 기술이 연구되고 있다[10]. Commercial 컴포넌트들은 이러한 역컴파일 방지 기술을 통해 자신들의 바이너리 컴포넌트를 역컴파일하지 못하도록 할 것이다. 이러한 역컴파일이 방지 기술을 택한 컴포넌트에 대해서는 컴포넌트의 소스 파일을 얻어 낼 수 없다. 따라서 성능 측정 코드를 삽입 할 수 없고, 컴포넌트의 성능을 측정할 수 없게 된다.

3. Wrapping 방법을 통한 컴포넌트 성능 측정 도구 설계

3.1 성능 측정 요소

EJB에서 컴포넌트의 성능 측정 요소로 빈 처리 응답 시간, 트랙잭션 처리 응답 시간, 알고리즘 처리 시간, 메모리 사용률, 풀 크기에 따른 메소드 처리시 사용된 CPU 활용률을 들 수 있다.

3.1.2 빈 처리 응답 시간

빈 처리 응답 시간은 클라이언트가 메쏘드의 수행을 요청하고 그 결과를 받을 때까지의 시간을 측정한다. 빈 처리 응답 시간을 통해 하나의 기능을 수행하는데 까지 얼마만큼의 시간을 소모하는지를 측정할 수 있다. 각 메쏘드들의 처리 응답 시간을 측정하여 합산함으로써 컴포넌트의 처리 응답 시간을 측정 할 수 있다.

3.1.2 트랜잭션 처리 응답 시간

엔티티 빈은 데이터의 지속성을 유지하기 위해서 데이터를 데이터베이스에 저장하거나 읽어온다. 데이터베이스에서 데이터를 처리하는 시간은 비즈니스 로직부분에서 처리에 시간이 많이 소요되는 부분이다. 트랜잭션 처리 시간을 측정함으로써 컴포넌트의 성능을 측정한다.

빈에서 데이터베이스를 액세스하는 트랜잭션 처리는 엔티티 빈에서 수행한다. 따라서 트랜잭션 처리는 엔티티 빈에서 수행 처리시간을 측정하면 된다. 엔티티 빈에서 메소드 처리 응답시간은 각 빈들 처리응답시간과 같은 방법을 적용한다. 엔티티 빈에서 수행하는 메소드들의 처리응답시간을 합하여 엔티티 빈의 처리응답시간을 측정한다.

3.1.3 메모리 사용률

해당 컴포넌트를 처리하기 위해 시스템은 메모리

힙(heap)을 할당한다. 컴포넌트를 활용하기 위해서는 사용 시스템에서 최소 메모리 용량을 지원하여야 한다. 따라서 컴포넌트를 실행하기 위해 필요한 전체 메모리 크기를 알아야 한다. 또한 체크시간에 따른 빈 사용 메모리와 사용 가능한 메모리 크기를 측정한다. 이것은 체크하는 시간에 따른 메모리 자원 사용을 보기 위해서이다.

3.1.4 CPU 사용률

컴포넌트의 메쏘드가 실행되는 동안 CPU의 사용량을 측정함으로써 컴포넌트의 실행이 얼마만큼의 CPU를 점유하는지 알 수 있다. 이는 같은 기능을 수행하는 컴포넌트들 중 CPU의 부하를 최소화하는 것을 선택하는데 중요한 척도가 될 수 있다.

3.2 성능 측정 시스템 설계

3.2.1 성능 측정 시스템의 시나리오

사용자의 입력이나 Reflection을 사용하여 얻어낸 컴포넌트의 Spec에 따라 컴포넌트를 테스트하는데 사용될 Input Data Set을 자동 생성한다. 사용자가 새로 정의한 타입의 Class나 자료 구조에 대해서는 사용자가 추가적으로 Input Data를 입력할 필요가 있다. 이렇게 입력한 Input Data는 같은 기능을 수행하는 다른 컴포넌트에서 테스트할 수 있는 데이터로 만들기 위해 파일로 저장한다. 컴포넌트의 Spec 입력이 완료되면 컴포넌트의 Method를 사용하여 Test Program을 생성한다. 이 Test Program에는 컴포넌트의 성능을 측정할 수 있는 성능 측정 코드가 자동으로 삽입된다. 컴포넌트의 성능을 측정할 수 있는 Test Program이 생성되면 이를 컴파일한 후 실행하여 컴포넌트의 성능을 측정하고, 이 측정된 결과를 GUI(Graphic User Interface)를 통해 사용자에게 보인다.

그림 1은 이를 도식화한 것이다.

3.2.2 Performance Test Program Generator

사용자로부터 컴포넌트의 Spec을 입력받는다. 컴포넌트의 이름과 컴포넌트에서 사용하는 Method, 각 Method의 Parameter, Return Type등을 입력받는다. 여기서 Java의 Reflection 기능을 사용할 수 있는데, 이는 컴포넌트의 클래스 파일을 클라이언트 측에서 가지고 있어야 한다. 컴포넌트의 Spec 입력이 완료되면 Test Program을 구성하는 기본 구조를 가진 Test Program Template을 로드하여 컴포넌트의 Method와 성능 측정 코드를 삽입하여 Test Program을 생성한다.

그림 2는 Test Program을 생성하는 방법을 도식화 한 것이다.

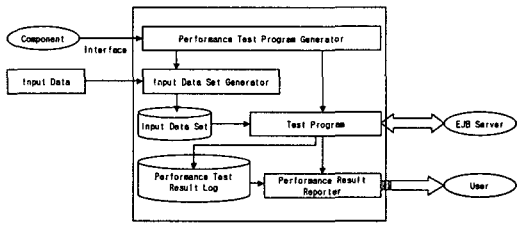

그림 1. 컴포넌트 성능 측정 시나리오

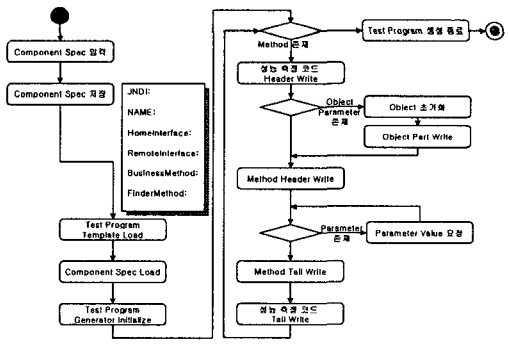

그림 2. Test Program 생성 방법

3.2.3 Input Data Set Generator

Input Data Set Generator는 컴포넌트의 비즈니스 메소드를 호출함에 있어서 필요한 Parameter를 생성한다. 분석된 클래스 파일에서 얻을 수 있는 Parameter 정보는 Parameter type의 이름 정도에 불과하다. Input Data Set Generator는 Parameter type을 입력받아 해당 타입의 랜덤한 실제 값을 생성시켜 그 값을 리턴한다. 리턴된 Parameter 값은 실행할 메쏘드의 Parameter로 사용한다.

Input DataSet Generator는 크게 Primitive type과 Object type을 처리해 주는 루틴으로 나눌 수 있다. Primitive type을 처리해주기 위한 부분은 다음과 같

은 알고리즘을 가진다.

Input Data Set Generator의 객체가 생성되면 데이터 파일을 생성한다. 데이터 파일은 Primitive type에 대한 값들을 저장하고 있는 파일이다. Parameter type이 주어지면 이 데이터 파일에서 Parameter에 해당하는 값을 검색하여 하나의 값을 리턴한다. 해당 타입에 대한 랜덤한 값들은 데이터 파일에서 클래스의 버퍼로 로딩되고, 그 버퍼에서 랜덤한 인덱스에 해당하는 값이 리턴된다.

그림 3은 이를 도식화 한 것이다.

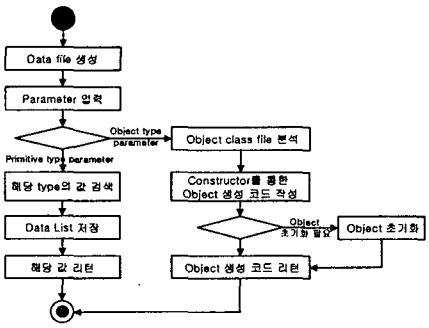

그림 3. Input Data Set Generator의 실행

3.2.4 Test Program

Performance Test Program Generator에 의해 생성된 Test Program은 기본적으로 EJB Client Program의 형태를 가지게 된다. Test Program은 EJB Server에 Deploy되어 있는 컴포넌트와 서로 통신하게 되며 컴포넌트의 메쏘드를 호출함으로써 메쏘드에 대한 성능 측정 코드를 통해 컴포넌트의 성능을 측정하고 그 결과를 Performance Test Result Log로 기록한다.

3.2.5 Performance Analysis Reporter

Test Program의 실행이 종료되어 Performance Test Result Log가 생성되면 이를 Performance Analysis Reporter가 분석하여 사용자에게 각 성능 측정 요소에 대해 Graph로 보인다.

4. 결론

본 연구에서는 컴포넌트의 성능 측정을 자동화하고, Wrapping 방법을 통해 성능 측정하는 방법을 연구하였다. 이 방법은 컴포넌트의 성능을 측정하기 위해 따로 소프트웨어를 구현해야 하는 문제점을 해결하고, 역컴파일이 불가능하도록 한 상업 컴포넌트에 대해서도 성능을 측정 할 수 있다. 또 같은 기능을 수행하는 컴포넌트에 대해 같은 데이터를 사용하여 테스트 할 수 있으므로 둘 이상의 컴포넌트에 대해 성능을 비교할 수 있다. 컴포넌트의 성능을 비교하여 더 나은 컴포넌트를 선택하여 소프트웨어를 개발함으로써 소프트웨어의 성능을 향상 시킬 수 있을 것이다.

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 Test방법", 소프트웨어공학기술 워크샵 2000 발표집, 1권 1호, pp.83-92, 8.2000

[5] 최병주, "EJB컴포넌트의 맞춤테스트기법", 소프트웨어공학기술 워크샵 2000 발표집, 1권1호, 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, 2000년 10월

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

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